CSE 431/531: Algorithm Analysis and Design — Spring 2018

# Homework 2

*Instructor: Shi Li* — **Deadline: 3/16/2018**

| Problems | 1 | 2 | 3 | 4 | Total Score |
|---|---|---|---|---|---|
| Max. Score | 10 | 10 | 10 | 40 | 70 |
| Your Score | | | | | |

**Collaboration Policy** You are allowed to discuss the homework problems with classmates. However, it is highly recommended that you first think about each problem for enough time before the discussion. You must write your solutions by yourself, in your own words. You need to write down the names of the students you collaborated with. For the programming problem, you must implement the algorithm by yourself.

**Problem 1 (10 points)** (Problem 2 of the Exercises for Greedy Algorithms in the Textbook) (a) Suppose we are given an instance of the Minimum Spanning Tree Problem on a graph $G$, with edge costs that are all positive and distinct. Let $T$ be a minimum spanning tree for this instance. Now suppose we replace each edge cost $c_e$ by its square, $c_e^2$, thereby creating a new instance of the problem with the same graph but different costs. True or false? $T$ must still be a minimum spanning tree for this new instance.

(b) Suppose we are given an instance of the Shortest $s$-$t$ Path Problem on a directed graph $G$, with edge costs that all positive and distinct. Let $P$ be a minimum cost $s$-$t$ path for this instance. Now suppose we replace each edge cost $c_e$ by its square, $c_e^2$, thereby creating a new instance of the problem with the same graph but different costs. True or false? $P$ must still be a minimum cost path for this new instance.

For each of the above two questions, if your answer is "True", you need to give an explanation. If your answer is "False", you need to give a counter-example.

**Problem 2 (10 points)** (Minimum Dominating Set on trees.) Given a graph $G = (V, E)$, a subset $S \subseteq V$ of vertices is called a dominating set of $G$ if for every $v \in V$, we either have $v \in S$, or there exists some $u \in S$ with $(u, v) \in E$. In other words, every vertex is either in $S$ or has a neighbor in $S$. Given a *tree* $T = (V, E)$, design an efficient greedy algorithm to find the minimum-size dominating set of $G$. It suffices for you to use the three-step approach: specify the greedy strategy, prove that the strategy is safe and then show what is/are the remaining instance(s) after you made the decision according the greedy strategy.

**Problem 3 (10 points)** (Problem 13 of the Exercises for Greedy Algorithms in the Textbook). Given a set of $n$ jobs $\{1, 2, 3, \cdots, n\}$, each job $j$ with a processing time $t_j > 0$ and a weight $w_j > 0$, we need to schedule the $n$ jobs on a machine in some order. Let $C_j$

be the completion time of $j$ on in the schedule. Then the goal of the problem is to find a schedule to minimize the weighted sum of the completion times, i.e, $\sum_{j=1}^{n} w_j C_j$.
**Example.** Suppose there are two jobs: the first takes time $t_1 = 1$ and has weight $w_1 = 10$, while the second job takes time $t_2 = 3$ and has weight $w_2 = 2$. Then doing job 1 first would yield a weighted completion time of $10 \cdot 1 + 2 \cdot 4 = 18$, while doing the second job first would yield the larger weighted completion time of $10 \cdot 4 + 2 \cdot 3 = 46$.

Design an efficient greedy algorithm to solve the problem. You may use the three-step approach as in Problem 2, but there is also a more direct approach to solve the problem.

**Problem 4 (40 points)** This is a programming problem. You need to implement the Kruskal's algorithm for the minimum spanning tree problem, using the union-and-find data structure. You can use C++, Java or Python to implement the algorithm.
**Input.** You need to read the input graph from the standard input (the console). In the first line of the file, we have two positive integers $n$ and $m$. $n$ is the number of vertices in the graph and $m$ is the number of edges in the graph. The vertices are indexed from 1 to $n$. You can assume that $1 \leq n \leq 10000$ and $1 \leq m \leq 100000$. In the next $m$ lines, each line contains 3 integers: $u, v$ and $w$, with $1 \leq u < v \leq n$ and $1 \leq w \leq 10^6$. This indicates that there is an edge $(u, v)$ of weight $w$. You can also assume that the graph is connected and there are no parallel edges.
**Output.** You need to output the result to the standard output (the console). The first line of the file is an integer indicating the total weight of the minimum spanning tree. From line 2 to line $n$, you need to output the $n$-1 edges in the minimum spanning tree. Each line contains 2 integers between 1 and $n$, indicating the two end-points of an edge.

| **Sample input (from the course slides)** | **Output for sample input** |
|---|---|
| 9 14 | 42 |
| 1 2 5 | 1 2 |
| 1 8 12 | 2 3 |
| 2 3 8 | 3 6 |
| 2 8 11 | 3 9 |
| 3 4 13 | 4 5 |
| 3 6 4 | 5 6 |
| 3 9 2 | 6 7 |
| 4 5 9 | 7 8 |
| 4 6 14 | |
| 5 6 10 | |
| 6 7 3 | |
| 7 8 1 | |
| 7 9 6 | |
| 8 9 7 | |

**Submission.** Submit your source file via UBlearns. For convenience, please only submit one source file, with the name MST_⟨firstname⟩_⟨lastname⟩_⟨UBID⟩.{cpp|py|java}, where ⟨firstname⟩ is your first name (with initial letter capitalized) and ⟨lastname⟩ is your last name (with initial letter capitalized) and ⟨UBID⟩ is your UB netID (as in your UB email address). The extension of the source file is cpp, py, or java, depending on the programming language you use.