Homework Assignment #7

Due: Dec. 13th, 2018

CMPUT 304
Department of Computing Science
University of Alberta

Note: All logarithms are in base 2 unless specified otherwise.
Exercises are for you and you alone. You need not submit them and they will not be graded.
You may submit as many answers to as many problems as you like.

Problem 1. (1 pts) Give an example of a graph where the greedy matching algorithm finds an optimal vertex cover; and give an example of a graph where the greedy matching algorithm finds a 2-approximation of the optimal vertex cover.

Problem 2. (1 pts) Recall the SET-COVER problem. The input is a grounds set U with n elements, and a collection $\mathcal{S} = \{S_1, S_2, S_3, ..., S_m\}$ where each $S_j \subset U$. Our goal is to find the smallest cover of all elements in U: a collection $S_{i_1}, S_{i_2}, ..., S_{i_t}$ such that $\bigcup_{j=1}^{t} S_{i_j} = U$.

Given a SET-COVER instance $(U, \mathcal{S})$ and some element $x \in U$ we denote the *frequency* of x as $f_x = \#\{S_j \in \mathcal{S} : \ x \in S_j\}$ — the number of sets in $\mathcal{S}$ that contain x. The max-frequency of a SET-COVER input $(U, \mathcal{S})$ is denoted as $F = \max_{x \in U} f_x$.

Give a poly-time algorithm that works by relaxing a suitable ILP into a LP and then rounding the LP solution, and results in a F-approximation for the SET-COVER problem.

Problem 3. (2 pts) In the KNAPSACK problem, the input is a collection of n items, each with weight w_i and value v_i. In addition, you are given an integer C, that denotes the capacity of your knapsack. You goal is to find a set of items S such that its total weight does not exceed C (i.e. $\sum_{x \in S} w(x) \leq C$) and which maximized the total value of all items:

$$\max\{\sum_{x \in S} v(x) : \quad \text{s.t.} \ \sum_{x \in S} w(x) \leq C\}$$

Give a fPTAS for the KNAPSACK problem.

Hint: Start with a dynamic programming solution that is similar to the SUBSETSUM problem. Denote $v_{\max}$ as the largest value of any item with weight $\leq C$, and thus $V = n \cdot v_{\max}$ bounds the largest value of any set in the knapsack. So fill a table of size $O(nV)$ with cell $A[i, u]$ denoting the min-weight of any subset of items $\{1, 2, .., i\}$ whose value sum to u. Then replace true item values with approximations. You can follow the details in `http://www.eecs.yorku.ca/~andy/courses/4101/lecture-notes/LN9.pdf` just remember to write the solution in your own words!