CS 473: Algorithms, Spring 2020
HW 11 (due Wednesday, May 5nd at 8pm)

This homework contains three problems. **Read the instructions for submitting homework on the course webpage**.

Collaboration Policy: For this home work, each student can work in a group with up to three members. Only one solution for each group needs to be submitted. Follow the submission instructions carefully.

For problems that ask for approximation algorithm, a full credit solution requires the following components:

- An algorithm that runs in polynomial time and retuns a valid solution (although sub-optimal).
- Proof of correctness and running time of the algorithm.
- Proof of approximation factor of the algorithm. This typically involve lower bounding OPT, and then obtaining an upper bound on the *value of the solution returned by your algorithm* as a function of lower bounds of OPT.

---

1. Provide a 1/2-factor, polynomial time, approximation algorithm for the Acyclic Subgraph problem:

   Input. An directed graph $G = (V, E)$.

   Output. A maximum-cardinality set of edges $E' \subseteq E$ such that $G[E']$ is acyclic.

   Hint. *Arbitrarily number the vertices from 1 to n. Let E_+ be the edges going in an increasing direction, and E_- be those in a decreasing direction. Pick the biggest of E_+ and E_-.*

2. Recall as discussed in class, that one possible 2-approximation for the Vertex Cover problem involves solving the LP relaxation of the standard integer linear program, and rounding up to 1 every coordinate where the optimal value was at least 1/2. This question asks you to extend this technique to the Set Cover problem:

   Input. A ground set $U = \{1, 2, \ldots, n\}$, and a collection of m subsets $S_1, \ldots, S_m \subseteq U$ with weight $w_i > 0$ for each subset $i \in \{1, \ldots, m\}$.

   Output. The minimum weight collection of these subsets which "covers" U, namely, a collection $I \subseteq \{1, \ldots, m\}$ such that $\bigcup_{i \in I} S_i = U$, and $\sum_{i \in I} w_i$ is minimized.

   (a) For the unit weight case, i.e., $w_i = 1, \forall i \in \{1, \ldots, m\}$, get a factor k polynomial time approximation algorithm for Set Cover, where k is the largest size of a subset, i.e., $k = \max_i |S_i|$.

(b) Extend the VERTEX COVER LP-rounding technique to get a factor f, polynomial time, approximation algorithm for SET COVER, where f is the maximum number of times some element appears in the subsets. Formally, if $f_i := |\{j : S_j \ni i\}|$, then $f = \max_i f_i$. (Note that Vertex Cover is the special case where $f = 2$.)

3. Consider the LP relaxation for Set Cover from the previous problem. Let x_i be the variable in the relaxation for set S_i. Suppose x^* is an optimum solution to the LP relaxation. Define $y_i = \min\{1, 2\ln n \cdot x_i^*\}$ for each set S_i. Pick each set S_i independently with probability y_i.

   - Prove that the expected weight of the sets chosen is at most $2\ln n \cdot OPT$.
   - Prove that the probability that any fixed element in the universe is *not* covered by the chosen sets is at most $1/n^2$.
   - Prove that, with probability at least $1 - 1/n$ all the elements of the universe are covered by the chosen sets. *Hint:* Use union bound.
   - Prove that with probability at least $(1/2 - 1/n)$ the algorithm outputs a set cover for the universe whose weight at most $4\ln n \cdot OPT$ where OPT is the weight of an optimum Set Cover. *Hint:* Use Markov's inequality.

The remaining problems are for self study. Do *NOT* submit for grading.

- In the Metric-TSP problem the goal is to find a minimum cost tour in a metric (V, d) that visits all the vertices. We saw Christofides's heuristic that gives a 3/2-approximation. Now consider the s-t TSP-Path problem in a metric space (V, d). Here the goal is to find an s-t walk of minimum cost that visits all the vertices. This differs from the tour version in that one does not need to come back to s after reaching t.
  - Given an example to show that the TSP tour can be twice the cost of a TSP Path. Also show that TSP tour is always at most twice the cost of a TSP path.
  - Obtain a simple 2-approximation for the TSP-Path problem via the MST heuristic.
  - **Hard:** Obtain a 5/3-approximation for the TSP-Path problem by modifying the Christofides heuristic appropriately.
- **Hard:** Consider the load balancing problem we discussed in lecture. One can obtain a $(1+\epsilon)$-approximation in polynomial time for any fixed $\epsilon > 0$. The goal of this problem is to give you an outline of this algorithm. Suppose we knew the optimum load is α^*. Partition the jobs into "large jobs" L which consists of all jobs which are bigger than $\epsilon\alpha^*$ and "small jobs" S which consists of all jobs which are smaller than $\epsilon\alpha^*$.
  - Suppose we have scheduled the big jobs L first and obtained a schedule with makespan at most $(1 + \epsilon)\alpha^*$. Describe an adaptation of the greedy list scheduling we discussed in

class to schedule the small jobs on top of the schedule for big jobs, and show that the resulting makespan is at most $(1 + 2\epsilon)\alpha^*$.

- Consider the big jobs L. Round up each job's size to next highest power of $(1+\epsilon)$. That is, if a job's size is between $(1 + \epsilon)^i$ and $(1 + \epsilon)^{i+1}$ we treat it as a job of size $(1 + \epsilon)^{i+1}$.
  - Show that the number of *distinct* job sizes that remain after the rounding is $O(1/\epsilon^2)$.
  - Describe a dynamic programming based algorithm to find an optimum schedule for the rounded up jobs — recall that we saw a special case of this for 3 job sizes in a previous home work. What is the running time of your algorithm?
  - Prove that if there is a schedule of makespan α^* for the original big jobs then there is a schedule of makespan at most $(1 + \epsilon)\alpha^*$ for the rounded up big jobs.
- Can you put the ingredients together to obtain a $(1+\epsilon)$-approximation in $n^{poly(1/\epsilon)}$ time? In particular, you also need to show how to guess α^* via binary search. This last step may be a bit hard.

- See Jeff's homework 11 from Spring 2016. `https://courses.engr.illinois.edu/cs473/sp2016/hw/hw11.pdf`