CS 787: Advanced Algorithms — Homework 1

Out: 09/27/16 **Due: 10/18/16**

Guidelines

- This homework consists of 7 problems. You are required to solve and turn in all of the problems.
- Some of the problems are difficult, so please get started early. Late submissions do not get any credit.
- Please typeset your solutions.
- Homework may be done in pairs. Please write your names clearly on your homework.

Problems

1. For $i \in \{1, \cdots, n\}$, let X_i denote a random variable whose value is drawn independently and uniformly at random from the interval $[0, 1]$. Let $X = \min_{i\in[n]} X_i$. Compute the expectation of X.

2. Consider adapting the randomized min-cut algorithm from class to finding an s-t min-cut in an unweighted graph. At every stage, we have an s-supernode and a t-supernode. Of all the edges that do not go between the s-supernode and the t-supernode, we pick a random edge and contract it. We continue until we are left with only the two supernodes.

   (a) Show that there are graphs for which this algorithm produces an s-t min-cut with exponentially low probability. (A specific example family of graphs, parameterized by size, would suffice.)

   (b) What is the largest number of distinct s-t min-cuts that a graph can have? Give a graph that achieves this number.

3. Recall that a vertex cover of a graph $G = (V, E)$ is a set of vertices $S \subset V$ such that each edge has at least one endpoint in S. We analyzed the following algorithm in class and showed that it obtains a 2-approximation for the Minimum Vertex Cover problem.

   (i) Start with $S \leftarrow \emptyset$.

   (ii) Pick an edge (u, v) such that $\{u, v\} \cap S = \emptyset$. Add both u and v to S.

   (iii) If S is a vertex cover, halt, else go to Step (ii).

   (a) Consider changing step (ii) of the algorithm to the following: "Pick an edge (u, v) such that $\{u, v\}\cap S = \emptyset$. Add an arbitrary endpoint of the edge to S."

   For any given integer $n > 0$, construct an instance of size n on which this algorithm may return a set $\Omega(n)$ times larger than the smallest vertex cover.

   (b) Next consider changing step (ii) to the following: "Pick an edge (u, v) such that $\{u, v\} \cap S = \emptyset$. Flip a coin and with probability $1/2$ add u to S, else add v to S."

   Show that this variant achieves a 2-approximation. *(Hint: Consider any vertex v in the optimal vertex cover, and let $N(v)$ be the set of the neighbors of v as well as v itself. What is $E[|S \cap N(v)|]$?)*

   (c) Suppose each vertex v has a weight $w(v)$, and the objective is to pick a set of smallest weight. Give an example to show that the above algorithms (the original one and the two variants) do not work for this problem.

(d) Finally consider changing step (ii) so that upon picking edge (u, v), we add u to S with probability $\frac{w(v)}{w(u)+w(v)}$ and add v to S with probability $\frac{w(u)}{w(u)+w(v)}$. If W is the weight of the least-weight vertex cover, show that this variation obtains an expected weight $\mathrm{E}[\sum_{v \in S} w(v)] \leq 2W$.

4. Recall that in the *Coupon Collector* problem, at every step you receive a random coupon out of a set of n different coupons; The goal is to determine how many coupons you need to collect in expectation before you have seen at least one of each kind. Consider the variant where at every step the probability of receiving the ith coupon is p_i. Note that $\sum_{i=1}^{n} p_i = 1$. Prove that the expected number of steps in the process is at least P and at most $O(P \log n)$, where $P = \max_{i \in [n]}(1/p_i)$.

5. In class we developed DP-based exact as well as approximate algorithms for the Knapsack problem. In this problem you will develop an LP-based approximation.

   (a) Write down an LP relaxation for the Knapsack problem.

   (b) Describe what the optimal solution to the LP looks like.

   (c) Develop a rounding that achieves a 2-approximation. *(Hint: Use the fact that the LP's value is an upper bound on the optimal solution; Construct a solution out of the items that the LP picks up fractionally.)*

6. In the s-t shortest paths problem, you are given a directed graph with non-negative lengths ℓ_e on edges, and special nodes s and t. Your goal is to find the length of the shortest path from s to t. Write down an LP to solve this problem exactly. Prove that the LP solves the problem exactly. The size of the LP should be polynomial in the size of the graph.

7. In the *Max-Cut* problem, we are given an unweighted graph G with vertex set V and edge set F; Our goal is to partition V into V_1 and V_2, so that the total number of edges "cut" by the partition, $|F \cap (V_1 \times V_2)|$, is maximized. Consider the following randomized algorithm: for every vertex $v \in V$, independently flip an unbiased coin; Let V_1 be the set of all vertices for which the coin came up heads, and let V_2 be the remaining vertices.

   (a) What is the expected number of edges cut?

   (b) Determine an upper bound on the probability that fewer than $|F|/4$ edges were cut. Try to obtain as small a bound as you can.