Analysis of Algorithms	**Problem Set 4**
CS 6820 Fall 2014	**Due 5pm, Wednesday, November 12, 2014**

This problem set has 5 problems with parts of varying difficulty. For full credit for a grade of A, solve four of the five problems. A full solution for each problem includes proving that your answer is correct. If your group cannot solve a problem, but can do some parts, or have partial results, write down how far you got, and why are you stuck.

Students may work on homework in groups of up to 2-3 people. Each group may turn in a single solution set that applies to all members of the group. However, students are asked to understand each of their group's solutions well enough to give an impromptu whiteboard presentation of the solution. You may use any fact we proved in class without proving the proof or reference, and may read the relevant chapters of the Kleinberg-Tardos or Kozen books, provided you state them clearly. However, you may **not use other published papers, or the Web to find your answer**.

Solutions can be submitted on CMS in pdf format (only). Please type your solution or write extremely neatly to make it easy to read. If your solution is complex, say more than about half a page, please include a 3-line summary to help us understand the argument.

Please ask any clarifying questions using Piazza, where we will post all answers.

(1) Consider the following simple model of gambling in the presence of bad odds. At the beginning, your net profit is 0. You play for a sequence of n rounds, and in each round, your net profit increases by 1 with probability 1/3, and decreases by 1 with probability 2/3.

Show that the expected number of steps in which your net profit is positive can be upper-bounded by an absolute constant c independent of the value of n.

(2) In the vertex cover problem, you are given an undirected graph $G = (V, E)$, possibly with weights $w_v \geq 0$ for each node $v \in V$. A *vertex cover* is a set $A \subset V$ of vertices such that A contains at least one end of every edge (A covers every edge). The vertex cover problem requires you to find a vertex cover A in a graph G with minimum total weight, that is, $\sum_{v \in A} w_v$ (or, in the unweighted case, minimum cardinality $|A|$). In this problem, we will consider the following simple randomized vertex cover algorithm.

Start with $S = \emptyset$.
While S is not a vertex cover,
    Select an edge e not covered by S.
    Select one end of e at random (both ends with equal probability).
    Add selected node to S.
Endwhile

We are interested in the expected cost of a vertex cover selected by this algorithm. We say that an algorithm is a c-approximation algorithm if the cost of the solution is at most c times the minimum possible cost.

(a.) Is this algorithm a c-approximation algorithm for the minimum weight vertex cover problem for some constant c? Prove your answer.

(b.) Is this algorithm a c-approximation algorithm for the minimum cardinality vertex cover problem for some constant c? Prove your answer.

Hint: For an edge, let p_e denote the probability that edge e is selected as an uncovered edge by this algorithm. Can you express the expected value of the solution in terms of these probabilities? To bound the value of an optimal solution in terms of the probabilities p_e, try to bound the sum of the probabilities for the edges adjacent to a given vertex v, that is, $\sum_{e \text{ adjacent to } v} p_e$.

(3) Assume you have n balls and n bins, and each ball is placed in a bin chosen independently at random (with each bin equally likely). Throughout this problem, use the approximation $(1 - 1/n)^n \approx 1/e$ whenever it is useful (where e is Euler's constant).

(a.) Prove that the expected number of empty bins approaches n/e for large n. Hint: remember that expectation is linear.

(b.) Assume that you have n jobs and n machines, and each job is run on a machine chosen independently at random (with each machine equally likely). Assume that if a machine is selected by more than one job, it will do the first job, and reject the rest. What is the expected number of rejected jobs?

(c.) Now assume in the above job-machine example that each machine will instead do the first two jobs, and reject the rest if more than two jobs are assigned to it. What is the expected number of rejected jobs now?

(4) Consider the following variant of the minimum cost perfect matching problem. Given a bipartite graph with houses and buyers, assume that each edge $e = (h, i)$ connecting a house h to a buyer i is annotated with a value $\text{val}(h, i)$ of the house h for buyer i. Finding a matching of maximum total value is said to maximize social welfare. The linear program we considered on the midterm,

$$\begin{aligned} x_e &\geq 0 \text{ for all } e \in E \\ \sum_{e \text{ adjacent to } v} x_e &\leq 1 \text{ for all } v \in V, \end{aligned}$$

$$\max \sum_e c_e x_e,$$

can be viewed as the relaxation of this matching problem. A solution to this program in integers with c_e defined to be $\text{val}(h, i)$ for each edge $e = (h, i)$, is a maximum value matching. In this problem, we consider the dual linear program that you worked out on the midterm. Dual variables correspond to nodes. Let p_h be the dual variable corresponding to the house h in a linear programming dual solution.

(a) A way to think of the market prices for houses is a set of prices p_h for each house, and a matching (not necessarily perfect) of the bipartite graph of houses and buyers such that

- each person i is matched to his/her favorite house: if he/she is matched to a house h, $\text{val}(h, i) \geq p_h$, and for any other houses k, $\text{val}(k, i) - p_k \leq \text{val}(h, i) - p_h$;
- if a person i is not matched to any house, he/she isn't interested in buying houses at their list price, i.e. $val(h, i) \leq p_h$ for all h.
- The sellers do OK as well: houses not assigned to any buyer have price $p_h = 0$, and all other houses have $p_h \geq 0$.

Show that under these conditions, the matching selected is of maximum total value, and even a maximum value fractional solution of the above linear program, and the prices of the houses are part of an optimal dual solution. What is the remainder of the corresponding dual solution?

(b) We have seen a repeated shortest paths based algorithm that finds minimum cost perfect matching. Show how to use the algorithm with repeated shortest path algorithm to find a maximum value matching.

(c) Show that this matching is also the maximum value solution to this linear program. Show that an optimal dual solution gives prices satisfying the conditions in (a). Show how to find the prices.

(5) It turns out there is a very fast randomized algorithm to solve linear programs when the number of variables, n, is a small constant. Suppose the linear program is:

$$\begin{aligned} \max cx \\ \text{subject to } 0 \leq x \leq 1 \\ \text{and } Ax \leq b \end{aligned}$$

The algorithm permutes the m constraints in $Ax \leq b$ into a random order, and it iteratively solves the linear program defined by the first i of these constraints, for $i = 1, \ldots, m$. Suppose $x^{(i)}$ is the optimum solution of the linear program subject to only the first i constraints. When the new constraint $a_i x \leq b_i$ is introduced, the algorithm does one of following two things:

- If $a_i x^{(i-1)} \leq b_i$, then $x^{(i-1)}$ is still optimal; set $x^{(i)} = x^{(i-1)}$.
- If $a_i x^{(i-1)} > b_i$, then the new optimum, $x^{(i)}$, must satisfy the linear equation $a_i x^{(i)} = b_i$. Use this equation to eliminate one of the n variables, expressing it as a linear function of the remaining $n-1$ variables. Substitute this expression to rewrite the linear program as a linear program in $n-1$ variables; recursively solve this LP to find the optimum point, $x^{(i)}$.

Prove that the algorithm's expected running time is bounded above by $O(2^{O(n \log n)} m)$. Thus, for any constant number of variables n, there is a randomized linear-time algorithm for linear programming.