Homework 5
Due: 3:30pm Tuesday May 3, 2016

1. In class, we have discussed the randomized min-cut algorithm using **contraction**. However, we didn't discuss how to implement the contraction algorthm. Assuming the multi-graph is given in an adjacency list format. Describe how to implement the contraction algorithm efficiently.

2. A gold processor has two sources of gold ore, source A and source B. In order to keep his plant running, at least three tons of ore must be processed each day. Ore from source A costs 20 USD per ton to process, and ore from source B costs 10 USD per ton to process. Costs must be kept to less than 80 USD per day. Moreover, Federal Regulations require that the amount of ore from source B cannot exceed twice the amount of ore from source A. If ore from source A yields 2 oz. of gold per ton, and ore from source B yields 3 oz. of gold per ton, how many tons of ore from both sources must be processed each day to maximize the amount of gold extracted subject to the above constraints? Show how this problem can be solveding using linear programming.

3. Assume you are given a list of real numbers. Your goal is to determine if there is a pair of numbers in this list whose product is exactly 1. What is the fastest deterministic algorithm you can devise to solve this problem? Assume that you can do exact multiplication and division in $O(1)$ time.

   Now suppose you have access to a hash table. Can you use it to get a faster algorithm? What properties would you like the hash function to have?

4. Assume that a set of n distinct numbers are inserted into a heap in random order. What is the expected number of times that the minimum value in the tree changes?

5. You walk into a room, and see a row of n cards. Each one has a number x_i written on it, where i ranges from 1 to n. However, initially all the cards are face down. Your goal is to find a local minimum: that is, a card i whose number is less than or equal to those of its neighbors, $x_{i-1} \geq x_i \leq x_{i+1}$.

   The first and last cards can also be local minima, and they only have one neighbor to compare to. Clearly there can be many local minima, but you are only responsible for finding one of them. Obviously you can solve this problem by turning over all n cards, and scanning through them. However, show that you can find such a minimum by turning over only $o(n)$ cards.

6. Show that how you can solve the shortest path problem using linear programming. Specifically, you may assume that you are given a weighted directed graph $G(V, E)$, and two vertices $s, t \in V$. The goal is to find the shortest $s - t$ path.

7. Simulate the execution of the augmenting path based maximum flow algorithm for the network in Figure 7.

8. Design an efficient algorithm to find a minimum-sized vertex cover for a given bipartite graph.

   Recall that a vertex cover of an undirected graph $G(V, E)$ is a subset of vertices C such that each edge in the graph is incident to at least one vertex from C.

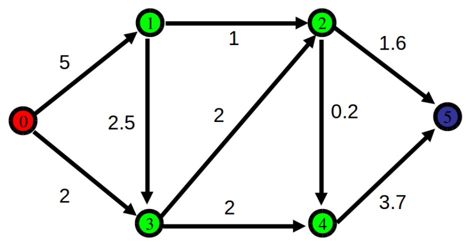

Figure 7: A flow network. The source vertex is 0 and the sink vertex is 5.

9. Suppose you have an LP algorithm that can only handle LPs with non-negative variables. Show to how to reduce the general LP to this one.