CS 170 Algorithms
Spring 2015 Christos Papadimitriou and Prasad Raghavendra HW 5

Due Feb. 27, 5:00pm

Instructions. This homework is due Friday, February 27, at 5:00pm electronically. Same rules as for prior homeworks.

1. (20 pts.) Negative edges

(a) Consider a directed graph with no negative cycles in which the only negative edges are those that leave some node s; all other edges are positive. Will Dijkstra's algorithm, started at s, work correctly on such a graph? Give a proof if your answer is yes, otherwise give a counter-example.

(b) Now consider the case (a directed graph with no negative cycles) in which the only negative edges are those that leave v, a vertex different from the start node s. Given two vertices s,t, give an efficient algorithm to compute the shortest path from s to t (should have the same asymptotic running time as Dijkstra's algorithm).

2. (25 pts.) Roads and Planes

In a country with n cities, we have two methods of travel: roads and planes. (more formally, our graph is $G = (V, R \cup P)$, where V is our set of cities, R is the set of roads, and P are the set of planes). A road i is specified as an **undirected** edge (u_i, v_i, C_i) that connects the two cities u_i, v_i with a **non-negative** cost C_i. A plane j is specified as a **directed** edge (u_j, v_j, C_j) which connects city u_j to v_j (one way) with cost C_j. Due to some weird airline award program, the cost of a plane edge **can possibly be negative!**. There is also an additional constraint that, if there is a plane edge from u_j to v_j, then there is no sequence of roads and planes that we can follow that takes us back from v_j to u_j.

Assume we have the the adjacency list representation of roads and planes (two separate data structures), and we can freely swap between the two methods of travel (i.e. we can take an arbitrary amount of roads, then planes, then roads, and so on...).

Given two cites s,t, give an efficient algorithm to compute the shortest path from s to t (should have same running time as Dijkstra).

Hint: Notice that the directed (plane) edges are between different connected components formed by the undirected edges (roads).

3. (20 pts.) Currency exchange

Shortest-path algorithms can be applied to currency trading. Suppose we have n currencies $c_1, c_2, \ldots, c_n$: e.g., dollars, Euros, bitcoins, or whatever. For any pair i, j of currencies, there is an exchange rate $r_{i,j}$: you can buy $r_{i,j}$ units of currency c_j at the price of one unit of currency c_i. Assume that $r_{i,i} = 1$ and $r_{i,j} \geq 0$ for all i, j. These exchange rates satisfy the condition that $r_{i,j} \times r_{j,i} < 1$ for all i, j: if you start with one unit of currency c_i, change it into currency c_j, and then change it back, you are left with less than one unit of currency c_i.

(a) Give an efficient algorithm for the following problem: given a set of exchange rates $r_{i,j}$ and two specific currencies s, t, find the most advantageous sequence of currency exchanges for converting currency s into currency t. We recommend that you represent the currencies and rates by a graph whose edge lengths are real numbers.

(b) Occasionally, it is possible to find currencies $c_{i_1}, \ldots, c_{i_k}$ such that $r_{i_1,i_2} \times r_{i_2,i_3} \times \cdots \times r_{i_{k-1},i_k} \times r_{i_k,i_1} > 1$. This means that by starting with one unit of currency c_{i_1} and then successively converting it to currencies $c_{i_2}, c_{i_3}, \ldots, c_{i_k}$ and finally back to c_{i_1}, you would end up with more than one unit of currency c_{i_1}. Such anomalies last only a fraction of a minute on the currency exchange, but they provide an opportunity for profit.

Given an efficient algorithm for detecting the presence of such an anomaly. Use the same graph representation as for part (a).

4. (10 pts.) Maximum Spanning Tree

Show how to find the maximum spanning tree of a graph, i.e. the spanning tree of largest total weight.

5. (10 pts.) Spanning Trees and Shortest paths

Suppose you are given a weighted graph $G = (V, E)$ where all edge weights are positive and distinct.

1. Prove that G has a unique minimum spanning tree.
2. Suppose you are also given a distinguished vertex s. Is it possible for a tree of shortest paths from s and a minimum spanning tree in G to not share any edges? If so, give an example. If not, give a reason.

6. (15 pts.) Another MST Algorithm

In this problem, we will develop a new algorithm for finding minimum spanning trees. It is based upon the following property:

Pick any cycle in the graph, and let e be the heaviest edge in that cycle. Then there is a minimum spanning tree that does not contain e.

(a) Prove this property carefully.

(b) Here is the new MST algorithm. The input is some undirected graph $G = (V, E)$ (in adjacency list format) with edge weights $\{w_e\}$.

```
Sort the edges according to their weights;
for edge $e \in E$, in decreasing order of $w_e$ do
    if $e$ is part of a cycle of $G$ then
        $G = G - e$ (that is, remove $e$ from $G$);
return $G$
```

Prove that this algorithm is correct.

(c) What is the overall time taken by this algorithm, in terms of $|E|$?