Algorithms
Computer Science 140 & Mathematics 168
Spring 2012
Homework 8
Due Thursday, March 8

On this assignment, you may appeal to any results that we've proved in class without reproving them. If you wish to use any such result, just state the result that you're using.

1. **[30 Points] Arbitrage: An Algorithmic Get-Rich-Quick Scheme!** You've been hired by the Wall Street firm of Weil, Proffet, and Howe at a whopping salary.

   Weil, Proffet, and Howe has just entered the **arbitrage** business. Arbitrage is a money-making scheme involving anomalies in international currency exchange rates. For example, imagine that 1 U.S. dollar buys 0.8 Zambian kwachas, 1 Zambian kwacha buys 10 Mongolian tughriks, and 1 Mongolian tughrik buys 0.15 U.S. dollars. Then, by converting currencies, a trader can start with 1 U.S. dollar and buy $0.8 \times 10 \times 0.15 = 1.2$ U.S. dollars. By capitalizing on such anomalies quickly (before they're detected and corrected by the markets), huge amounts of money can be made. This, of course, requires efficient algorithms![1]

   We'll assume that there are n currencies and we're given the exchange rates in the following adjacency list format: We have an array of n entries, one for each currency. Element i in the array is a linked list of all the currencies that can be exchanged directly from currency i. For each currency j in that linked list, there is a positive value $R[i, j]$ indicating that one unit of currency i buys $R[i, j]$ units of currency c_j. Note that the number of currencies in the list for currency i may be as large as $n - 1$ (there are $n - 1$ other currencies) but it may be substantially smaller since there may not be direct exchange rates established between some pairs of currencies. Moreover, note that due to exchange anomalies, it is not necessarily the case that $R[i, j] = 1/R[j, i]$. For example, if 1 Zambian kwacha buys 10 Mongolian tughriks, it may not necessarily be the case that 1 Mongolian tughrik buys 1/10 of a Zambian kwacha.

   (a) For this first part of the problem, assume that there exist no cycles that allow you to get arbitrarily rich via arbitrage. Here's your objective: Given the adjacency list of currencies and exchange rates and a particular "starting" currency c_i, determine the maximum amount of each of the other currencies that you can obtain, beginning with 1 unit of currency c_i. Describe an algorithm for solving this problem *by slightly modifying some existing algorithm*. Very briefly explain how the proof of correctness would work (you don't have to write out the complete proof - one or two sentences about how the proof would be structured will suffice) and give its running time. For full credit, make sure that your algorithm is fast. In particular, if each currency can only be directly exchanged for a constant number of other currencies, your algorithm's running time should be $O(n^2)$.

[1] It's interesting to note that there are many firms on Wall Street who have in-house algorithms experts!

(b) While modifying an existing algorithm is perfectly reasonable, another approach is to use an existing algorithm and to modify the *data* instead of the *algorithm!* The advantage of this approach is that you can use an existing algorithm off-the-shelf without monkeying with its guts (ugh, bad metaphor!). In this part of the problem, describe how you could modify the data and apply some existing algorithm to solve the problem in part (a). *Hint:* Apply some simple function to the edge weights, solve a standard shortest path problem with these new edge weights, and finally apply some other function to get back the solution that we're looking for (the maximum profit attainable for each currency starting with our start currency). What is the running time of your algorithm using this approach? Note that applying the functions to change the edge weights takes time and must be accounted for in your run time.

(c) Now assume that the exchange rates are such that it may be possible to get rich via arbitrage. That is, there may exist a cycle of currency exchanges that allows you to make more of your starting currency than you had initially (which may or may not involve your starting currency). Describe how you could modify one of your two approaches above to determine if it's possible to profit from arbitrage with the given currencies and exchange rates. (For this part of the problem, the algorithm need only return a boolean indicating whether or not it is possible to make profit using arbitrage.) Briefly explain why your algorithm is correct.

(d) Finally, your boss at Weil, Proffet, and Howe would like to find the set of currencies in an actual profitable cycle if one exists. Your boss is not at all concerned about the running time, as long as it is polynomial time. For example, using some polynomial time algorithm numerous times is just fine. (Weil, Proffet, and Howe has a very fast supercomputer - so as long as your algorithm runs in polynomial time, they'll be able to use it.) However, your boss *does* want a proof that your algorithm finds a profitable cycle if one exists. Your job will be much easier if you don't modify any algorithm further, since doing so will require substantial effort in proving its correctness. However, if you construct your algorithm from one or more existing algorithms (algorithms from class and/or your algorithms from the previous part of this problem), you can use the fact that they have already been established to be correct! Describe the algorithm for finding a positive profit cycle and give a proof of its correctness. The proof should end up being quite short.