

Due March 6, 2:45pm

Instructions: Same instructions as usual. Write your name, username, TA's name, and discussion section time on the first page of your homework. List your study partners for this homework.

As always, when asked for an algorithm, please provide: (a) pseudocode, (b) a brief explanation of the main idea/intuition, (c) a proof of correctness, (d) the asymptotic running time of the algorithm, using $O(\cdot)$ notation, (e) justification for your running time.

1. (100 pts.) Currency exchange

Shortest-path algorithms have an important application in the currency market. Suppose we have n currencies: e.g., dollars, Euros, etc. For any pair i, j of currencies i, j , there is an exchange rate $r_{i,j}$: you can buy $r_{i,j}$ units of currency j at the price of one unit of currency i .

Normally, these exchange rates satisfy the condition that you can't make money out of thin air by converting among the currencies. Mathematically, we normally have $r_{i,j} \times r_{j,i} < 1$: i.e., if you start with one unit of currency i , change it into currency j , and then change it back, you are left with less than one unit of currency i . Similarly, normally changing money from currency i to currency j to currency k back to currency i leaves you with less than you started with, and so on.

- (a) Assuming that the normal condition listed above holds, give an efficient algorithm for the following problem: given the exchange rates $r_{i,j}$ and two currencies s, t , find the cheapest sequence of conversions for converting currency s to currency t .

Hint: You may want to build a weighted graph. You can make the edge weights real numbers, if you want. It's possible to solve this problem in $O(n^3)$ time.

- (b) Occasionally, the normal condition listed above is violated, and it is possible to find a sequence of currency conversions that actually leaves you with more money than you started with. In other words, occasionally it is possible to find currencies i_1, \dots, i_k such that $r_{i_1, i_2} \times r_{i_2, i_3} \times \dots \times r_{i_{k-1}, i_k} \times r_{i_k, i_1} > 1$. This abnormal situation lets you make money for free, if you can discover this abnormal condition and exploit it by issuing the sequence of currency conversions you discovered. This is known as currency *arbitrage*. The catch is that these anomalous conditions normally last only for a very brief period of time, so you have to discover them quickly and execute your arbitrage before the anomaly disappears. And of course everyone else is trying to find this condition, too, so you have to find it before they do.

Show that the problem of detecting whether such an anomaly exists can be recast as the problem of detecting whether the weighted graph you introduced in part (a) has a negative cycle (i.e., a cycle whose length is negative). Based on this, describe an efficient algorithm to detect whether such an anomaly exists.

P.S. Yes, this really happens, and some investors really do make money this way! But I don't recommend trying this at home. In the mid-90's, one hedge fund hired a bunch of really smart people to design financial algorithms and began to engage in arbitrage on a massive scale, using borrowed money to increase their leverage. For several years they had incredible returns (over 40% per year). But then in 1998, they lost several billion dollars after a huge arbitrage deal on the international bond market went awry. The hedge fund had to be bailed out by the Fed and major banks, because of concerns that its losses were so huge that they could lead to a collapse in financial markets around the globe. Oops.