Lexx Pino & Rohan Reddy
Math 260

November 24, 2020

Project Discussion

Description

Our project idea was to test the Bellman Ford algorithm as an application finding currency arbitrage opportunities given data sets in the form of currency exchange rates. The files data1.csv through data5.csv contain lists of exchange rates for various currencies, each containing different pathways to currency arbitrage. The files are read in through a csv reader in the Arbitrage class that converts the csv into a pair of lists, which becomes inputs for the Bellman Ford algorithm below it.

In the bellman.py file is the code for the arbitrage class along with the algorithm itself. The algorithm reads in a list of currencies along with pairs of exchange rates (all possible currency pairing given the list) which comes from the csv file. Then it initializes a graph with edges between each possible pair of vertices. An important step here is taking the negative log of conversion rates which becomes the edges of the graph, since this is how the algorithm detects negative weight cycles. Given a starting vertex, it calculates the shortest distance to each of the other vertices. It returns all the negative weight cycles, which are the actual pathways for exploiting arbitrage opportunities in the forex market. These negative cycles are then displayed.

A computational challenge faced is finding the existence of all negative-weight cycles and representing distinct cycles, as opposed to many different paths that contain the same negative cycle but enter and exit the cycle at different nodes. We could not find a workable solution to address this issue, so our code as it is now outputs all negative weight paths, which can result printing the same, or very similar, currency trading paths. This is also the reason we did not get to attempt transaction fee handling. While volume-based transaction fees would be difficult to incorporate into the structure of the graph, a flat-fee could be processed by counting the number of edges in the arbitrate opportunity-cycle and computing how much total cost in fees the trader is incurring.

For a bit of mathematical context, the Bellman Ford algorithm is capable of detecting negative-weight cycles in a directed graph. We can represent arbitrage opportunities as negative-weight cycles. For example, if we start with S units of currency, and we make multiple exchanges, thereby multiplying S by a series of exchange rates, we desire to end with more than S units of currency:
$S(r_1)(r_2)(r_3)... > S$

$r_1 r_2 r_3 ... > 1$

$log(r_1 r_2 r_3 ...) > 0$

$log(r_1) + log(r_2) + log(r_3)... > 0$

$-log(r_1) + (-log(r_2)) + (-log(r_3))... < 0$

Thus, if each edge weight is $-log(r_i)$, the existence of a negative cycle represents an arbitrage opportunity.

Discussion

What intrigued us the most during this project was understanding why arbitrage is not generally accessible to retail investors like ourselves. The reality behind currency arbitrage is that the opportunities are exploited nearly instantaneously by trading algorithms and market corrections, or they are just not profitable because of transaction fees outweighing the profit margins. In general, the data sets we wrote up provide numerous paths to arbitrage but all have nearly negligible profit margins.