Currency Arbitrage Detector

Seher Khan (seher.khan@khi.iba.edu.pk)

Abstract:

Due to inefficiencies in the pricing of currencies, it is possible to earn profits at low risk by identifying opportunities and engaging in fast trading. However, such opportunities are transient and difficult to identify. This project provides an implementation of a standard way of identifying such an opportunity using the Bellman Ford algorithm in Java.

It is possible to represent each currency as a node and buying and selling exchange rates as directed edges. Solving the problem then involves finding a path that maximizes the product of edge weights. Since it is simpler to find the shortest path in a graph, the problem was converted to such by applying a negative log transformation on the data. Given a source currency, the Bellman Ford algorithm was applied to find the negative weight cycle (if one existed) in the resulting graph that would allow currency arbitrage. Finally, breadth first search was used to identify currencies that were reachable from this cycle.

To ensure this program was useful in real time, a web scraper on Python (using the package Selenium) was also developed to extract exchange rates of 79 currencies (6241 entries) from the website, http://www.forex.pk/foreign-exchange-rate.htm. In the GUI, the user was given the choice of selecting what currencies to include in their graph and of eliminating edges.

Data Structure

1) Adjacency Matrix: Integer[][]

	AUD	CAD	EUR	.	.	.	ZMK
AUD	null	-log(e_{ac})	-log(e_{ae})	.	.	.	-log(e_{az})
CAD	-log(e_{ca})	null	-log(e_{ce})	.	.	.	-log(e_{cz})
EUR	-log(e_{ea})	-log(e_{ec})	Null	.	.	.	-log(e_{ce})
.	.	.	.	.			.
.	.	.	.		.		.
.	.	.	.			.	.
ZMK	-log(e_{za})	-log(e_{ez})	-log(e_{ze})	.	.	.	null

2) Vector of Currency Codes: String[]

"AUD"	"CAD"	"EUR"	.	.	.	"ZMK"

3) Vector of Currency Names: String[]

"Australian Dollar (AUD)"	"Canadian Dollars (CAD)"	"Euro (EUR)"	.	.	.	"Zambia Kwacha (ZMK)"

4) Vector of Distances: double[]

AUD	CAD	EUR	.	.	.	ZMK
d_a	d_c	d_e	.	.	.	d_z

5) Vector of Previous vertices: int[]

AUD	CAD	EUR	.	.	.	ZMK
$Prev_a$	$Prev_c$	$Prev_e$	.	.	.	$Prev_z$

6) Reachable from Set A: LinkedList<Integer>

LinkedList containing the index numbers of currencies that are reachable from Set A.
This is populated by a BFS function called by the detect NWC function

7) Negative Weight Cycle: LinkedList<Integer>

LinkedList containing the index numbers of currencies that form the negative weight cycle if one exists.
This is populated by a function called by the detect NWC function which involves traversing "previous" vertices

Choice of Data Structure: Adjacency List vs Adjacency Matrix

Since the graph is likely to be dense and number of vertices fixed (since a currency exchange outlet will deal in a fixed number of currencies), the space and time complexity of both adjacency matrix and adjacency list were similar. However, adjacency matrix provided an additional benefit of direct access. As a result, an adjacency matrix was employed to represent the currency exchange graph.

Big-O Analysis:

Space Complexity for the Graph was $O(|V|^2)$ where V represents the number of vertices.

Function	Complexity
Create graph from data file	**Time:** $O(|V|^2)$ *- here V is always 79, since reading from csv file and every line is read, irrespective of being stored or not*
Remove edge	**Time:** O(1)
Has edge – tells you if an edge exists	**Time:** O(1)
Detect NWC (Bellman Ford) – tells you if a negative weight cycle exists	**Time:** $O(|V|^3)$ **Space:** O(V) - *Need space for Set A (vertices that are relaxed in last cycle. Maximum is V, however, for large, dense graphs, this number is likely to be low)*
Find Negative Weight Cycle – identifies the negative weight cycle	**Time:** O(V)
Infinite Arbitrage Possible (through BFS) – is the destination vertex reachable from the negative weight cycle?	**Time:** $O(V^2)$

Example (Dry Run)

Data: **Nov 4, 2017 (file: Nov042017_forex_data.csv)**
Select AUD, CAD, EUR, PKR, AED, GBP

AUD: **null** 0.023678127354577146 0.41688001092346283 -4.390317172991616 -1.0332200699010574 0.5354599047243516
CAD: -0.023716526617316065 **null** 0.39319074723277614 -4.414019365710055 -1.0569218616139007 0.5118261240995742
EUR: -0.4168665308326069 -0.3931225849097779 **null** -4.807160374173071 -1.450066964016957 0.11867092971767863
PKR: 4.390058806371146 4.414549826379441 4.803621124711929 **null** 3.3581378922017087 4.919880930827792
AED: 1.03310548646599 1.0569906139772902 1.449873342762762 -3.3571026765508405 **null** 1.5687759307152107
GBP: -0.5354987239708748 -0.5118051438794908 -0.11867152971749854 -4.925844722020161 -1.5687409101019962 **null**

Unselect some edges according to the graph below

Updated Graph:
AUD: **null** 0.023678127354577146 **null null null null**
CAD: **null null null null** -1.0569218616139007 **null**
EUR: **null** -0.3931225849097779 **null null null null**
PKR: 4.390058806371146 4.414549826379441 **null null** 3.3581378922017087 **null**
AED: 1.03310548646599 **null** 1.449873342762762 **null null null**
GBP: **null** -0.5118051438794908 **null null null null**

D: Infinity Infinity Infinity 0.0 Infinity Infinity
PREV: -1 -1 -1 -1 -1 -1

1st iteration of Bellman Ford
Looking at edge: (AUD, CAD)
Looking at edge: (CAD, AED)
Looking at edge: (EUR, CAD)
Looking at edge: (PKR, AUD)
Relaxed edge: (PKR, AUD)
D: 4.390058806371146 Infinity Infinity 0.0 Infinity Infinity
PREV: 3 -1 -1 -1 -1 -1
Looking at edge: (PKR, CAD)
Relaxed edge: (PKR, CAD)
D: 4.390058806371146 4.414549826379441 Infinity 0.0 Infinity Infinity
PREV: 3 3 -1 -1 -1 -1
Looking at edge: (PKR, AED)
Relaxed edge: (PKR, AED)
D: 4.390058806371146 4.414549826379441 Infinity 0.0 3.3581378922017087 Infinity
PREV: 3 3 -1 -1 3 -1
Looking at edge: (AED, AUD)
Looking at edge: (AED, EUR)
Relaxed edge: (AED, EUR)
D: 4.390058806371146 4.414549826379441 4.808011234964471 0.0 3.3581378922017087 Infinity
PREV: 3 3 4 -1 3 -1
Looking at edge: (GBP, CAD)

2nd iteration of Bellman Ford
Looking at edge: (AUD, CAD)
Relaxed edge: (AUD, CAD)
D: 4.390058806371146 4.413736933725723 4.808011234964471 0.0 3.3581378922017087 Infinity
PREV: 3 0 4 -1 3 -1
Looking at edge: (CAD, AED)
Relaxed edge: (CAD, AED)
D: 4.390058806371146 4.413736933725723 4.808011234964471 0.0 3.356815072111822 Infinity
PREV: 3 0 4 -1 1 -1
Looking at edge: (EUR, CAD)
Looking at edge: (PKR, AUD)
Looking at edge: (PKR, CAD)
Looking at edge: (PKR, AED)
Looking at edge: (AED, AUD)
Relaxed edge: (AED, AUD)
D: 4.389920558577812 4.413736933725723 4.808011234964471 0.0 3.356815072111822 Infinity
PREV: 4 0 4 -1 1 -1
Looking at edge: (AED, EUR)
Relaxed edge: (AED, EUR)
D: 4.389920558577812 4.413736933725723 4.806688414874584 0.0 3.356815072111822 Infinity
PREV: 4 0 4 -1 1 -1
Looking at edge: (GBP, CAD)

3rd iteration of Bellman Ford
Looking at edge: (AUD, CAD)
Relaxed edge: (AUD, CAD)
D: 4.389920558577812 4.413598685932389 4.806688414874584 0.0 3.356815072111822 Infinity
PREV: 4 0 4 -1 1 -1

Looking at edge: (CAD, AED)
Relaxed edge: (CAD, AED)
D: 4.389920558577812 4.413598685932389 4.806688414874584 0.0 3.356676824318488 Infinity
PREV: 4 0 4 -1 1 -1
Looking at edge: (EUR, CAD)
Relaxed edge: (EUR, CAD)
D: 4.389920558577812 4.413565829964806 4.806688414874584 0.0 3.356676824318488 Infinity
PREV: 4 2 4 -1 1 -1
Looking at edge: (PKR, AUD)
Looking at edge: (PKR, CAD)
Looking at edge: (PKR, AED)
Looking at edge: (AED, AUD)
Relaxed edge: (AED, AUD)
D: 4.389782310784478 4.413565829964806 4.806688414874584 0.0 3.356676824318488 Infinity
PREV: 4 2 4 -1 1 -1
Looking at edge: (AED, EUR)
Relaxed edge: (AED, EUR)
D: 4.389782310784478 4.413565829964806 4.80655016708125 0.0 3.356676824318488 Infinity
PREV: 4 2 4 -1 1 -1
Looking at edge: (GBP, CAD)

4th iteration of Bellman Ford
Looking at edge: (AUD, CAD)
Relaxed edge: (AUD, CAD)
D: 4.389782310784478 4.4134604381390545 4.80655016708125 0.0 3.356676824318488 Infinity
PREV: 4 0 4 -1 1 -1
Looking at edge: (CAD, AED)
Relaxed edge: (CAD, AED)
D: 4.389782310784478 4.4134604381390545 4.80655016708125 0.0 3.356538576525154 Infinity
PREV: 4 0 4 -1 1 -1
Looking at edge: (EUR, CAD)
Relaxed edge: (EUR, CAD)
D: 4.389782310784478 4.413427582171472 4.80655016708125 0.0 3.356538576525154 Infinity
PREV: 4 2 4 -1 1 -1
Looking at edge: (PKR, AUD)
Looking at edge: (PKR, CAD)
Looking at edge: (PKR, AED)
Looking at edge: (AED, AUD)
Relaxed edge: (AED, AUD)
D: 4.3896440629911435 4.413427582171472 4.80655016708125 0.0 3.356538576525154 Infinity
PREV: 4 2 4 -1 1 -1
Looking at edge: (AED, EUR)
Relaxed edge: (AED, EUR)
D: 4.3896440629911435 4.413427582171472 4.806411919287916 0.0 3.356538576525154 Infinity
PREV: 4 2 4 -1 1 -1
Looking at edge: (GBP, CAD)

5th iteration of Bellman Ford
Looking at edge: (AUD, CAD)
Relaxed edge: (AUD, CAD)
D: 4.3896440629911435 4.4133221903457205 4.806411919287916 0.0 3.356538576525154 Infinity
PREV: 4 0 4 -1 1 -1
Looking at edge: (CAD, AED)

Relaxed edge: (CAD, AED)
D: 4.3896440629911435 4.4133221903457205 4.806411919287916 0.0 3.35640032873182 Infinity
PREV: 4 0 4 -1 1 -1
Looking at edge: (EUR, CAD)
Relaxed edge: (EUR, CAD)
D: 4.3896440629911435 4.413289334378138 4.806411919287916 0.0 3.35640032873182 Infinity
PREV: 4 2 4 -1 1 -1
Looking at edge: (PKR, AUD)
Looking at edge: (PKR, CAD)
Looking at edge: (PKR, AED)
Looking at edge: (AED, AUD)
Relaxed edge: (AED, AUD)
D: 4.3895058151978095 4.413289334378138 4.806411919287916 0.0 3.35640032873182 Infinity
PREV: 4 2 4 -1 1 -1
Looking at edge: (AED, EUR)
Relaxed edge: (AED, EUR)
D: 4.3895058151978095 4.413289334378138 4.806273671494582 0.0 3.35640032873182 Infinity
PREV: 4 2 4 -1 1 -1
Looking at edge: (GBP, CAD)

6th iteration of Bellman Ford (this extra iteration is done to detect if a negative weight cycle exisits)
Looking at edge: (AUD, CAD)
Relaxed edge: (AUD, CAD) // CAD to be added to A
D: 4.3895058151978095 4.413183942552386 4.806273671494582 0.0 3.35640032873182 Infinity
PREV: 4 0 4 -1 1 -1
Looking at edge: (CAD, AED)
Relaxed edge: (CAD, AED) // AED to be added to A
D: 4.3895058151978095 4.413183942552386 4.806273671494582 0.0 3.3562620809384858 Infinity
PREV: 4 0 4 -1 1 -1
Looking at edge: (EUR, CAD)
Relaxed edge: (EUR, CAD) // CAD to be added to A, CAD is already in A so won't be added again
D: 4.3895058151978095 4.413151086584804 4.806273671494582 0.0 3.3562620809384858 Infinity
PREV: 4 2 4 -1 1 -1
Looking at edge: (PKR, AUD)
Looking at edge: (PKR, CAD)
Looking at edge: (PKR, AED)
Looking at edge: (AED, AUD)
Relaxed edge: (AED, AUD) // AUD to be added to A
D: 4.389367567404475 4.413151086584804 4.806273671494582 0.0 3.3562620809384858 Infinity
PREV: 4 2 4 -1 1 -1
Looking at edge: (AED, EUR)
Relaxed edge: (AED, EUR) // EUR to be added to A
D: 4.389367567404475 4.413151086584804 4.806135423701248 0.0 3.3562620809384858 Infinity
PREV: 4 2 4 -1 1 -1
Looking at edge: (GBP, CAD)

A: [1, 4, 0, 2]
Q: [1, 0, 4, 2]

Doing BFS (to find what nodes are reachable from Set A, and hence the negative weight cycle, these will be the possible destination currencies):
dequed vertex is CAD
AUD is added to reachableFromA

AED is added to reachableFromA
dequed vertex is AUD
CAD is added to reachableFromA and the Q.
dequed vertex is AED
EUR is added to reachableFromA
dequed vertex is EUR
dequed vertex is CAD

reachableFromA: [0, 4, 1, 2]

Locate negative weight cycle
D: 4.389367567404475 4.413151086584804 4.806135423701248 0.0 3.3562620809384858 Infinity
PREV: 4 2 4 -1 1 -1
Begin from prev of CAD: EUR
Follow prev v times:
prev: AED; prev: CAD; prev: EUR; prev: AED; prev: CAD; prev: EUR;
Now certainly in nwc
Add EUR to nwc
And follow prev until EUR repeats.
prev: AED; prev: CAD; prev: EUR; NWC: [2, 1, 4, 2]