Generalized Minimum Cost Flow and Arbitrage in Bitcoin Debit and Custodian Networks

Jose I. Orlicki**

RSK Labs, CABA C1414AXD, Argentina
jorlicki@rsk.co

Abstract. This project was focused on designing a tool for optimizing complex transactions in a traditional bank accounts, Bitcoin wallet accounts and Bitcoin exchanges. The challenges include transference fees eroding the amounts transferred, Bitcoin volatility and arbitrage. The tool should support hundreds of accounts with each account identified by a bank or exchange, and a currency. The basic user-case, we call it Debit Scenario, appears when an account consumes it capital and has very little funding (sink or demand account), then the user set a minimum funding for the account, and we need to balance all the accounts of the network, using many currency circulations though many paths to satisfy the specified funding limits. Another user case, we called it Custodian Scenario, is when certain accounts (i.e. supply or non-custodians) has upper limit on the amount of capital so we need to distribute the excess value in other accounts (i.e. demand or custodians) of the network. The problem was modelled as a multi-commodity min-cost max-flow problem with specific constraints including arbitrage information, and solved with linear programming. Simple and complex numerical scenarios are presented too.

Keywords: financial networks, optimization, Bitcoin, minimum cost circulation

1 Introduction

The Bitcoin and cryptocurrencies market is not very mature, lack of liquidity and price stability appears on a daily or weekly basis. Also, the regulatory landscape is very unstable with country and regulations changing very often. So, when funding bank accounts related to Bitcoin wallet operation you need to traverse a complex network of traditional banking, wire transfers and startup exchanges that offer a diversity of fees and delays on the transactions.

** This research was conducted as part of my Quant Trainee program at Xapo Holdings Ltd., during August-December 2017, and was presented as Capstone Project for the Master in Science of Financial Engineering at Stevens Institue of Technology. The project was directed by Professor Dr. Khaldoun Khashanah and I want to thank specially Wenceslao Casares and Federico Murrone from Xapo for the continous support of this project and discussions on many of the details of the problem statement.

A tool for generating execution plans for transferring funds across the financial network satisfying certain limits can be modelled as an Optimization Problem and can be very useful for the daily or weekly operation of individuals and companies providing custodian and other financial services. Traditional consumer banks do not offer high speed intraday transactions with other banks so we are only concerned with generating a daily or weekly execution plan. Although we are optimizing cost, some exploratory work on optimizing time has also been done, but is not detailed in this report. Also arbitrage opportunities must be taken into consideration to alleviate cost and make some profit.

In the rest of the report we show the literature review in Section 2. We show a mathematical model on Section 3, numerical results in Section 4 and including concluding remarks in Section 6.

2 Literature Review

The min-cost circulation problems have been investigated in the early day of maximum flow algorithms (Even et al., 1976; Truemper, 1977; Christofides et al., 1979). Namely, the multi-commodity integer flow problem was shown to be NP-complete even if the number of commodities is two, for both directed and undirected graphs (Even et al., 1976). Also, Truemper (1977) presents the relations between max flow positive gains problem with min cost in simple graphs. On Christofides et al. (1979) the authors presents how the problems of space, time and interest arbitrage can be solved using graph-theoretic algorithms. In our case, we will be using linear programming to solve a special case (one commodity per node) of the multi-commodity min cost circulation with arbitrage and fees (positive and negative gains) problem.

We analyzed the literature regarding foreign exchange arbitrage in complex networks and the circulation problem for the general multi-commodity and non-conservative gain-loss variations. Traditional triangular arbitrage assumes there is clique within a single exchange, any account with currency X can communicate with other accounts with currency Y by trading instantly. In our case, we are considering both triangular and market arbitrage when the different banks or exchanges quote different rates.

Lecture notes from advanced courses provide insight on current formulations of general circulation problems (Karger, 2012; Bansal, 2012; Williamson, 2012; Kingsford, Kingsford). Thesis from Wayne (2002) and a survey from Shigeno (2004) provides a complete detailed explanation of algorithmic for this kind of problems.

On the theoretical side, recent advancements have shown that general arbitrage problems are NP-complete (Palasek, 2014), and that ensembles of arbitrage-free networks can be a useful tool for financial models (Cai and Deng, 2016).

Our research and implementation follow the recent efforts to tackle specific circulation models adapted to foreign exchange arbitrage (Jones, 2001) and interest rate arbitrage (Cantú and Possani, 2012). These endeavors are basically a continuation of previous research, as mentioned earlier, by Christofides et al. (1979).

Very recently, Olver and Végh (2016) has shown a strongly polynomial algorithm for the generalized flow maximization problem.

3 Model

3.1 Basic Model

The problem we are solving is a generalized version of the max-flow circulation problem that includes cost and many commodities. Logically, we want to minimize cost, then is a min-cost max-flow integer circulation problem.

We are considering an integer flow, because we know that this case can be solved efficiently (Wayne, 2002). We can model the problem as an integer flow because we can use small fractions of a cent as minimum transactional atom. For example, in the Bitcoin Financial Network the minimum atom is the *satoshi* unit, with $100,000,000$ *satoshi* = 1 *bitcoin*.

In our case we are solving a multi-commodity problem that is intractable in the general case, but we are following a central reference (Wayne, 2002) that solved the problem for a single commodity. The contribution of this paper is how we transformed a specific multi-commodity problem into a single commodity problem, including the possibility of arbitrage.

> In the decision version of the problem, producing an integer flow satisfying all demands is NP-complete (Even et al., 1976).

Wayne (2002) models the non-conservative nature of the financial network is modelled using a gain factor. Gain can be positive for arbitrage and negative for fees charged by service providers.

> In traditional networks, there is an implicit assumption that flow is conserved on every arc. Many practical applications violate this conservation assumption. Freight may be damaged or spoil in transit; fluid may leak or evaporate. In generalized networks, each arc has a positive multiplier associated with it, representing the fraction of flow that remains when it is sent along that arc. The generalized maximum flow problem is identical to the traditional maximum flow problem, except that it can also model networks which "leak" flow.

Following Wayne (2002) directly we will model the circulation problem with link gain with a multiplier in each link of the graph. In the original model there are three link functions: capacity, cost, and gain. In our case the we only have two functions: capacity and gain/loss. We incorporated the fee percentages into the multiplicative gain function and the flat fees are incorporated into the linear equations as a scalar offset.

In our model we only have one type of commodity, i.e. currency, per node. If we have a two-currency account in one exchange we consider this case to be two different account nodes in our model. So, trading a currency for another is also another link. This is sound because to exchange between different currencies also pays fees. We also avoid a more general multi-commodity problem that is more difficult to solve. So, each account has its own conversion to a base currency, USD in our case, so we reduce the problem to a single-commodity problem.

Finally, the arbitrage situation is handled by introducing in the gain function the ratio r_B/r_A of the two exchange rates for an account link connecting account A with account B. In the bibliography, a similar approach for account rates and link arbitrage is called *reduced prices* (Karger, 2012), but is modelled targeting supply chain, not currency circulation. So, we can say the contribution of this research is converting from multi-commodity to single commodity and considering arbitrage and fees for the specific case of one commodity per node.

The model is presented in the next section.

Definition 1. *The input to the generalized minimum cost circulation problem is a generalized network $G = (V, E, c, g)$, where (V, E) is a directed graph with node set V and arc set E, $c : E \to \mathbb{R}_{\geq 0}$ is a* capacity function*, $g : E \to \mathbb{R}$ is a* gain *function. For notational convenience we assume that there are no parallel arcs so that each arc can be uniquely specified by its endpoints. We let $n = |V|$ and $m = |E|$. A* generalized circulation *or flow $f : E \to \mathbb{R}_{\geq 0}$ is a nonnegative function that satisfies the flow conservation constraints:*

$$\forall v \in V : \sum_{w \in V:(v,w)\in E} f(v,w) = \sum_{w \in V:(w,v)\in E} g(w,v) f(w,v)$$

All incoming flow to a node is flow is going out from the neighboring now and multiplied by the gain function.

It is feasible *if, in addition, it satisfies the* capacity constraints*:*

$$\forall (v,w) \in E : f(v,w) \leq c(v,w)$$

We can define the cost of a circulation f *as*

$$cost(f) = \sum_{(v,w)\in E} f(v,w)[1 - g(w,v)]$$

Notice that $1 - g(v,w)$ is the simple financial return of the that traversed edge. For example, a -0.01 (1%) percent fee for a trade or transference, or a positive 0.02 (2%) percent gain if there is a big arbitrage scenario between two accounts with the same currency. The generalized minimum cost circulation problem is to find a feasible generalized circulation of minimum cost.

Although we defined an interesting problem, to make it more realistic, we need to add supply and demand nodes, nodes generating excess commodity or demanding an amount of commodity. In our model, we used balance upper and lower constraints, but this can be translated to as excess supply on top of upper maximum bound and excess demand below lower minimum desired balance.

3.2 Complete Model

Because our final model has more constraints than the basic model, such including a balance for each node and upper and lower constraints on the model, we decided to directly implement the model using Linear Programming (Even et al., 1976) and the Simplex Algorithmic, that has many robust implementations. The other alternative was to extend the very efficient polynomial but specialized algorithms. These are polynomial in time complexity, described by Wayne (2002) and Karger (2012), which solve the problem by eliminating negative feedback loops from the graph circulation. Eliminating flow-absorbing circuit algorithms are very fast but the theory we assumed was beyond the scope of this final project that wanted to target an industry problem with a robust solution. Simplex solutions are provable slower but using native C/C++ implementations we can scale to hundreds of nodes as desired.

So, we present the detailed minimization problem we solved using the Simplex Algorithm.

Considering the problem with n = number of node accounts, m = number of bank links, also noted arcs or edges:

Definition 2. *We assume that the exchanges rates to base currency, USD, are fixed constants for this problem:*

- *CCY) $ccy(v_1)$,..,$ccy(v_n)$: currency name (foreign exchange or crypto) for each node*
- *RATES) $r(v_1)$,..,$r(v_n)$: currency rates to base currency for each node, for example BTC/USD, how many American dollars we have for each bitcoin.*
- *BAL) $b_0(v_1)$,..,$b_0(v_n)$: initial balance for each node;*
- *PFEE) $fee(e_1)$,..,$fee(e_m)$, $fee(e_j) \in \mathbb{Q}_{[0,1]}$: fractional fee per each link;*
- *FLAT) $flat(e_1)$,..,$flat(e_m)$: flat fee per each link, discounted after fee percentage is discounted.*
- *GAIN) $g(e_j) = g(v,w) = \frac{r(w)}{r(v)}(1 - fee(e_i))$, : gain by traversing the link, considering fractional fees and price arbitrage.*

Definition 3. *We introduce the variables of the linear model:*

- *VAR0) $bc(v_1)$,..,$bc(v_n)$, : current balances for each node in their original currency such as Bitcoin. We are not using theses variables in the model, we just use the following base currency variables;*
- *VAR1) $b(v_1)$,..,$b(v_n)$, : current balances for each node in base currency, $b(v_i) = bc(v_i) * r(v_i)$;*
- *VAR2) $t(v_1)$,..,$t(v_n)$, : unknown target balances after unknown circulation;*
- *VAR3) $f(v,w)$ for each $(v,w) \in E$, : unknown transferred cash-flow circulation across each bank link, non-negative.*

So, we have, considering VAR1 to VAR4, a total of $3 * n + m$ linear variables. We have $X = (x_1, \ldots, x_{3*n+m})$.

Definition 4. *The linear equality constraints ($A_{eq}X^T = b_{eq}$) are:*

– *EQ1)* $\forall i \in 1 \dots n, j \in 1 \dots n : A_{i,i} = 1,\ A_{i,j} = 0,\ i \neq j\ : b(v_i) = b_0(v_i)$ *current balance is constant.*
– *EQ2)*

$$\forall v \in V : \sum_{u \in V:(u,v) \in E} [g(u,v)f(u,v) - flat(u,v)] \; - \sum_{w \in V:(v,w) \in E} f(v,w) = t(v) - b(v)$$

: conservation of flow modulo gain (fees and arbitrage), in-going minus out-going flow equals difference in balance.

Notice that we used the gain constants $g(v,w)$ from Definition 2 to include fees and arbitrage. Also in the following inequalities we do not include negative values because we are not modeling negative liabilities.

Definition 5. *The inequality constraints* ($A_{ub}X \leq b_{ub}$) *are :*

– *INEQ1)* $i \in 1, \dots, n,\ \|t(v_i) - b(v_i)\| \leq maxDelta_i$ *: $2n$ equations;*
– *INEQ2)* $i \in 1, \dots, n,\ 0 \leq t(v_i) \leq maxBalance_i$ *: $2n$ equations;*
– *INEQ3)* $i \in 1, \dots, n,\ t(v_i) \geq minBalance_i \geq 0$ *: n equations;*
– *INEQ4)* $j \in 1, \dots, m,\ 0 \leq f(e_j) \leq maxCapacity_j$, *maximum capacity or liquidity of the link, $2m$ equations.*

Notice that there are no requirements for initial balances, can even be negative, because we assume the problem is that these initial balances have a problem of excess or lack of capital, so we need to find the flow to reach adequate target balances.

Definition 6. *Target function to minimize is the summation of the following functions:*

– *MIN1) maximum target balances:* $\sum_{v \in V} -t(v_i)$
– *MIN2) minimum flow:* $\sum_{(v,w) \in E} f(v,w)$

Notice that, there are no negative flows in the directed graph, although we can support negative balances.

4 Numerical Experiments

We present some simple and complex scenarios to show the model is solved using the linear programming implementation. We include a simple cost optimization scenario, a simple arbitrage scenario and a complex scenario with dozens of circulations.

When fractional or flat fees are not mentioned the default value is zero. If upper limit on flow in a link is not mentioned is assumed to be boundless or very large. If currency rates are not mentioned is assumed that they are the same for all nodes with the same currency.

4.1 Optimizing Cost

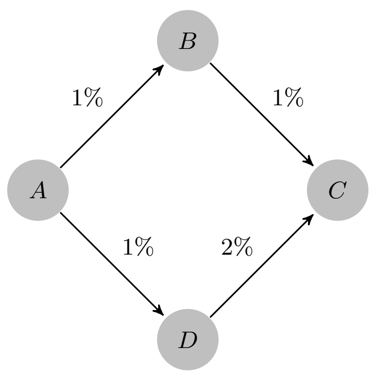
In this case, there is a demand of currency in node C and there is an excess supply in A that can be used to satisfy the demand if we can traverse nodes B or D. Depending on the cost and limits we might want to send the currency across B, D or both.

Table 1. A simple minimum cost example with 4 nodes and 2 paths.

	ccy	b	$minBalance$	maxBalance
A	USD	120	50	100
B	USD	80	50	100
C	USD	30	50	100
D	USD	80	50	100

	fee %
$A \to B$	1
$A \to D$	1
$B \to C$	1
$D \to C$	2

Table 2. Simple cost scenario, diagram, and solution



	$f(v,w)$	$f(v,w)[1-g(v,w)]$
$A \to B$	20.00	19.80
$A \to D$	0.00	0.00
$B \to C$	20.20	20.00
$D \to C$	0.00	0.00

	$t(v)-b(v)$	$t(v)$
A	−20.00	100.00
B	-0.40	79.60
C	20.00	50.00
D	0.00	0.00

As we can see the intermediate node B was chosen because that path has a combined fractional fee that is smaller. Notice that the flow with minimum cost also uses cash from node B if we send all the currency we need from A we are charged two hop fees for the complete amount, but in this case for a little fraction we can only pay one fee, only on the second hop.

4.2 Arbitrage

Now we solve an arbitrage scenario including two currencies. In this case there is a demand of currency (USD) in node F and there is an excess supply of Bitcoin (BTC) in A that can be used to satisfy the demand if we can traverse nodes

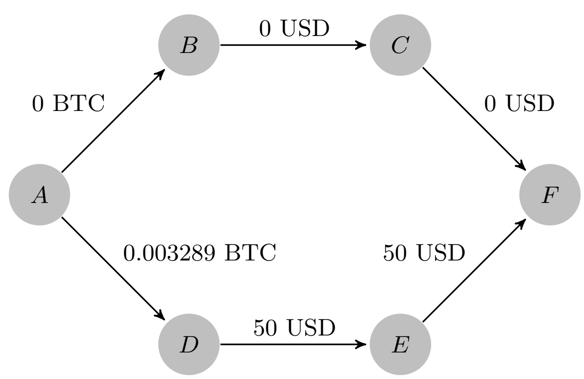
$\{B, C\}$ or $\{D, E\}$. Depending on the cost, limits, and rates we might want to send the currency across nodes $\{B, C\}$, $\{D, E\}$ or both paths. For simplicity, we assume zero fees on all links.

This scenario is designed to mimic the situation where we are moving BTC from an exchange with lower rates (A) to an exchange with higher rates (B or D), then we sell the BTC to get USD (nodes C or E) and finally we transfer the USD dollar to a third bank.

Table 3. A simple arbitrage scenario, including 6 nodes and 2 paths. Remember we use base currency USD for balances $b(v)$ but we show BTC nodes in BTC for illustration purposes.

	ccy	b	b_{BTC}	$minBalance$	$maxBalance$	rate (USD)
A	BTC	15,150	1.01	1	2	15,000
B	BTC	15,100	1	1	2	15,100
C	USD	50	–	50	100	1.00
D	BTC	15,200	1	1	2	*15,200*
E	USD	50	–	50	10	1.00
F	USD	0	–	50	100	1.00

Table 4. A simple arbitrage scenario: solution diagram and optimal solution



	$f(v, w)$	$f(v, w)[1 - g(v, w)]$	$f_{BTC}(v, w)$
$A \to D$	49.34	50.00	0.003289
$D \to E$	50.00	50.00	-
$E \to F$	50.00	50.00	-

	$t(v) - b(v)$	$t(v)$
A	−49.34	15100.66
B	0	15100
C	0	5
D	0	15200
E	0	50
F	+50	50

Notice that although the demand was 50 USD, the method managed to find a solution using an amount of BTC that was worth less than 50 USD in the original exchange A. This is called spatial arbitrage (Christofides et al., 1979).

4.3 Complex scenario with an exponential number of positive flows

Finally, more numerical results but including a class of random graphs. As the most simple class of non-trivial random directed graphs, we will generate optimal circulation for directed graph where each account node has two incoming edges coming from random neighbors. The cost of this kind of scenario will be analyzed theoretically and verified empirically assuming a fixed fractional fee per edge (1%), a fixed excess supply per account node (1 USD), and a single demanding node with an established demand.

Each account node has two incoming directed edges, so in the worst-case scenario, we have a directed tree flowing from the leaves to a single root node. We will only have one sink node v demanding supply, the root of the tree. For the tree, we have $2^h - 2$ nodes at distance h or smaller able to supply funds to the root node.

If balance is bigger than the minimum then demand for this account node is zero, and is negative if current balance is smaller than the minimum:

$$demand(v) \coloneqq 0 \ or \ minBalance(v) - b(v) \quad if \ minBalance(v) \geq b(v)$$

If demand has the following form:

$$demand(v) = 2^k - 2$$

we know that we need at least nodes with distance k or smaller from the root to satisfy the demand. For small fees, for example 0.01 (1%) or smaller, were the fees as return is close the log-returns (??, log), we will observe that total cost is at least a minimum of:

$$2^1 * 1 * fee + 2^2 * 2 * fee + \ldots + 2^{k-1} * (k-1) * fee$$

because 2 accounts need to hop 1 link to reach the demanding root node, at least 4 nodes need to traverse 2 links to the reach the demanding node, etc. Can be larger than this because as the fees leak some of the currency, extra nodes will be needed to satisfy the demanding root account.

Arbitrage is not considered in this scenario, we have 4 different currencies but all account nodes with the same currency share the same exchange rate. All experiments in this subsection were run on a network of accounts with $n = 262$ and $m = 515$. The average running time for each single row was 1.934 seconds on a personal computer.

Fig. 1. Visualization of example complex random directed network.

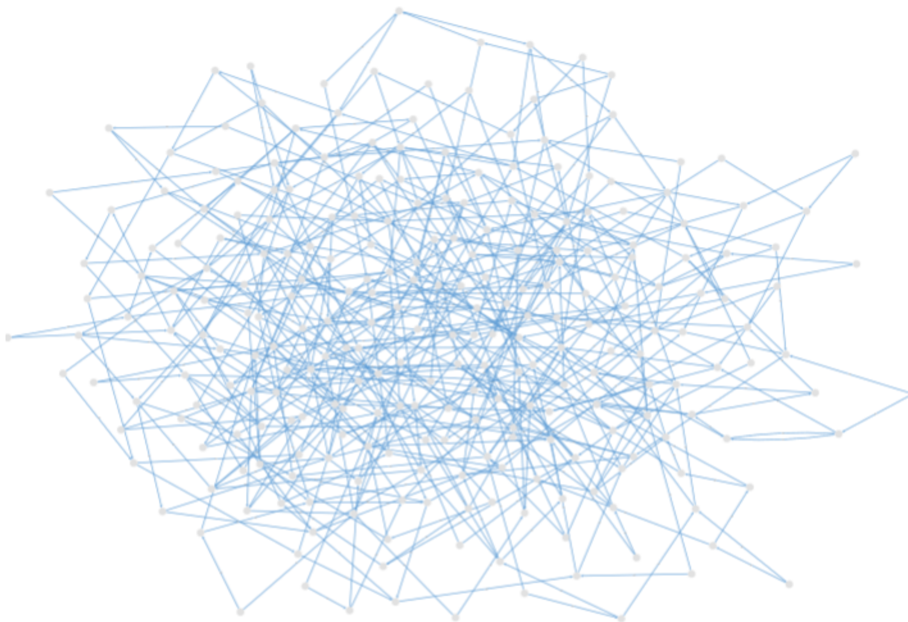


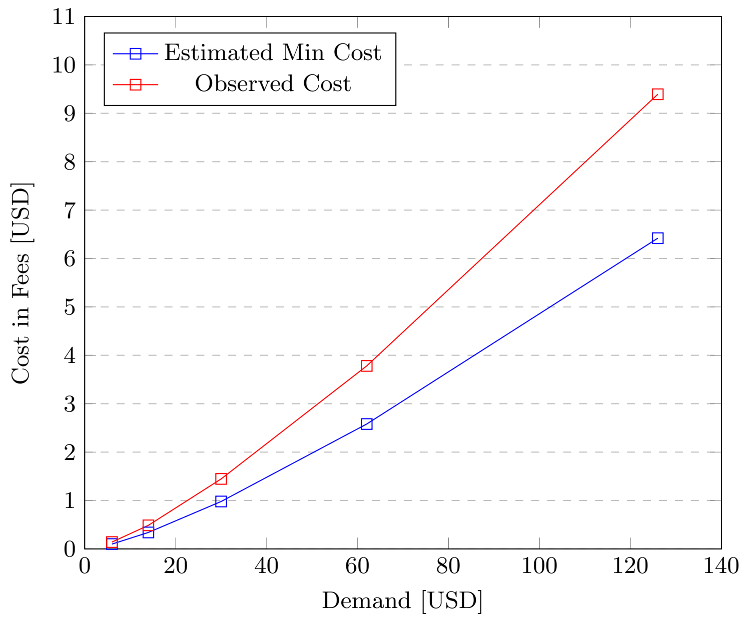
Table 5. Demand in USD versus Cost in USD for a 2-regular directed graph.

k	Demand	Analytical Cost	Observed Cost
3	6	0.1000	0.1418
4	14	0.3400	0.4870
5	30	0.9800	1.4461
6	62	2.5800	3.7804
7	126	6.4200	9.3931

5 Implementation

We chose Excel 2016 with Visual Basic for Applications automated features for the front-end of the application and Python 2.7 for the optimization back-end. The GLPK optimization engine was used. It can be invoked from Python and runs fast native C/C++ code. The alternative of using Python `scipy.optimize.linprog` was included but the interface is very similar and providing much lower speed.

Fig. 2. Demand versus Cost for a 2-regular directed graph. Estimated minimum versus empirical observations.



Average Linear Programming running time for the 5 exponential scenarios of Section 4.3 was 2.01 seconds on a personal computer with a dual 2.30GHz, 2 Core(s), 4 logical processors machine.

There are two users of the front-end: a Treasury Manager that inputs changes in the constraints of the accounts and generates new execution plans using the tool; and a Treasury Operator that executes the individual transactions of the circulation plans and updates the balances. Remember that although many of the Bitcoin operations can be automated, traditional banking is still done on a manual basis, so the operator does most of the work manually. In the future, maybe more banks will provide API to automate their services (BBVA, 2017).

6 Conclusion

A detailed model for minimum cost circulation on bank or exchanges network transfer with multiple currencies was described and implemented using linear programming.

The numerical results suggest that the implementation has scalability on scenarios with an exponential growth in the number of transactions.

Future work can include time optimization. The current implementation includes an optional parameter for activating an experimental feature of modeling time as leaking fees. This is a promising solution but you need to translate back the time-as-money circulation to the final circulation with only fees and arbitrage.

Regarding liabilities the model supports negative balances but there is not ownership of the debt, so further models need to be devised for debt.

Bibliography

Why Log Returns (2011) https://quantivity.wordpress.com/2011/02/21/why-log-returns/.

Bansal, N. (2012). The circulation problem. *2WO08: Graphs and Algorithms (unrefereed)*.

BBVA (2017). BBVA API https://www.bbvaapimarket.com/.

Cai, M. and X. Deng (2016). Arbitrage in frictional foreign exchange market. In *Encyclopedia of Algorithms*, pp. 141–145.

Cantú, C. and E. Possani (2012). A circulation network model for the exchange rate arbitrage problem. *Journal of Business and Economics 4*(1), 30–61.

Christofides, N., R. D. Hewins, and G. R. Salkin (1979). Graph theoretic approaches to foreign exchange operations. *The Journal of Financial and Quantitative Analysis 14*(3), 481–500.

Even, S., A. Itai, and A. Shamir (1976). On the complexity of timetable and multicommodity flow problems. *SIAM Journal on Computing 5*(4), 691–703.

Jones, C. K. (2001). A network model for foreign exchange arbitrage, hedging and speculation. *International Journal of Theoretical and Applied Finance (IJTAF) 04*(06), 837–852.

Karger, D. (2012). 6.854 advanced algorithms. *Advanced Algorithms Lecture Notes (unrefereed)*.

Kingsford, C. The circulation problem. *CMSC 451: Max-Flow Extensions (unrefereed)*.

Olver, N. and L. A. Végh (2016). A simpler and faster strongly polynomial algorithm for generalized flow maximization. *CoRR abs/1611.01778*.

Palasek, S. (2014). Arbitrage-free exchange rate ensembles over a general trade network. Papers, arXiv.org.

Shigeno, M. (2004). A survey of combinatorial maximum flow algorithms on a network with gains. *Thesis Journal of the Japan Operations Research Society 47*(4), 244–264.

Truemper, K. (1977). On max flows with gains and pure min-cost flows. *SIAM Journal on Applied Mathematics 32*(2), 450–456.

Wayne, K. D. (2002). A polynomial combinatorial algorithm for generalized minimum cost flow. *Mathematics of Operations Research 27*(3), 445–459.

Williamson, D. P. (2012). Lecture 11. *ORIE 633 Network Flows Lecture Notes (unrefereed)*.