Fall 12-14-2018

Intra-exchange Cryptocurrency Arbitrage Bot

Eric Han
San Jose State University

Intra-exchange Cryptocurrency Arbitrage Bot

A Project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Eric Han

December 2018

The Designated Project Committee Approves the Project Titled

Intra-exchange Cryptocurrency Arbitrage Bot

by

Eric Han

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSÉ STATE UNIVERSITY

December 2018

Dr. Thomas Austin	Department of Computer Science
Dr. Mark Stamp	Department of Computer Science
Dr. Jon Pearce	Department of Computer Science

ABSTRACT

Intra-exchange Cryptocurrency Arbitrage Bot

by Eric Han

Cryptocurrencies are defined as a digital currency in which encryption techniques are utilized to regulate generation of units of currency and verify the transfer of funds, independent of a central governing body such as a bank. Due to the large number of cryptocurrencies currently available, there inherently exists many price discrepancies due to market inefficiencies. Market inefficiencies occur when the price of assets do not reflect their true value. In fact, these types of pricing discrepancies exist in other financial markets, including fiat currency exchanges and stock exchanges. However, these discrepancies are more significant in the cryptocurrency domain due to the low levels of government regulation, higher amounts of speculation, and human behaviors driven by investors seeking profit. These types of pricing discrepancies can be eliminated to some extent by executing arbitrages, which are defined as a sequences of trades beginning and ending with the same asset which result in more of that asset at the end of the trading sequence. Through executing arbitrages, the market should become more efficient.

This project was an attempt to execute intra-exchange arbitrage on the well-known cryptocurrency exchange Binance and generate profit, and as a side effect make the cryptocurrency exchange market more fluid. Although the project did not record phenomenal profits, it did successfully generate several hundred dollars over the course of several months, independent of market fluctuations.

ACKNOWLEDGMENTS

I want to thank my family: KY Han, Sue-Jane Han, and Jason Han for their undying support of me through this period of my life, along with Professor Thomas Austin, Professor Mark Stamp, Professor John Pearce, and Professor Katerina Potika from San Jose State University Computer Science Faculty. Their outstanding pedagogy has inspired me as a student to try novel ideas, no matter how absurd they sound.

DEDICATION

I would like to dedicate this Project to all my my friends and family who are going through struggles and feel like there's no light at the end of the tunnel. Remember, a any dream you have written down with a date becomes a goal. A goal broken down into steps becomes a plan. A plan backed by actions makes your dreams come true.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

Introduction

1.1 Overview

Cryptocurrencies present many new opportunities that can be attributable to its utilization of blockchain technology. With technologies such as smart contracts, distributed ledger, and currency virtualization behind it, it is no wonder that there are over 4000 cryptocurrencies (or alt-coins) in existence today [1]. As a side effect of this explosion in number of cryptocurrencies, a similar rise in cryptocurrency exchanges has also occurred. Coinbase, Kraken, Bitfinex, Bitrex, and Poloniex are just a few of the many existing cryptocurrency exchanges[2]. These sites provide Application Programming Interfaces (APIs) to their users so they can trade programmatically. Due to the existence of these APIs, market inefficiencies present in cryptocurrency markets can be capitalized on through trades that execute on these inefficiencies.

In fact, this technique, known as arbitrage, has been performed by banks and financial institutions for many years. Unfortunately, access to a high availability API often requires significant financial investment and access is typically only given to banks and financial institutions such as investment houses[3]. The main research question this project seeks to answer is: *Can arbitrage techniques be applied to cryptocurrency markets successfully?*

1.2 Research Objective

The research objective is to develop an automated bot that can successfully find and execute arbitrage opportunities, and ultimately yield profits on the Binance exchange and make the market more efficient. I propose to apply this arbitrage technique to Binance, currently the largest cryptocurrency exchange in the world, because I believe that larger price discrepancies exist due to the high number of currencies available, at the time of this writing over 150, as well as the inherent volatility present in cryptocurrency values.

Building the foundation for a public-use library for an intra-exchange arbitrage bot will be be part of the research objective. Ideally, if one single application could make profits, then the objective can be extended to experiment running multiple instances of the same application to extend profits and further stabilization of the cryptocurrency-exchange toward market efficiency. If the arbitrage bot works on Binance, steps can be taken to abstract the code so it can be applied to other exchanges as well.

CHAPTER 2

Background and Related Work

2.1 History and Background

Currency arbitrage involves buying and selling currency pairs from different brokers to take advantage of their different spreads. The currency-spread is defined as the difference between “bid” and “ask” prices of an asset. “Bid” refers to what a buyer offers to pay for a specific asset, while “Ask” refers to the selling price of an asset offered by a seller. The size of the currency spread is often used to measure the liquidity and efficiency of the market. A higher currency-spread means the market is less efficient, while a lower currency-spread indicates a more efficient market.

In the fiat currency market, banks often use arbitrage to take advantage of pricing discrepancies between multiple currency pairs to make a profit. Bots that trade on conventional traditional markets such as Bloomberg and NASDAQ exist but are available exclusively to investment houses and brokers. These conventional market bots need access to exchange data from the market. This exchange data is typically not available to laymen [4].

The transparent nature of the blockchain gives cryptocurrency traders access to an exchange's order book and design trading bots that act on this data. The blockchain is essentially a distributed ledger with all transaction histories, maintained by users and miners of the network. As a result of this, there are already many bots that target cryptocurrency exchanges, such as “Crypto Trader”, “Haasbot”, “Zenbot”, “Gekko”, and “BTCRobot.” Many of these bots have a monthly subscription fee ranging from $60USD to $3,500USD. (At the time of this writing Bitcoin was over 8000USD) Most utilize an inter-exchange arbitrage strategy [5].

2.2 Types of Currency Arbitrage

As a result of the large number of cryptocurrency exchanges, two types of arbitrages can be performed to capitalize on market inefficiencies. The first is "inter-exchange" arbitrage, and the second is "intra-exchange" arbitrage. The former refers to executing arbitrage sequences within an exchange, while the second means executing arbitrages across different exchanges. Each has its own benefits and challenges, which will be discussed below.

- The first technique, inter-exchange arbitrage, was explored by Norry [5] in his article on Bitcoin trading bots. This type of arbitrage surveys cryptocurrency prices on many different exchanges, and finds pricing discrepancies between them. When a pricing discrepancy is found, a lower priced cryptocurrency will be bought on one exchange, then transferred and sold on the other. By doing these arbitrages across many different exchanges, the pricing of cryptocurrencies should stabilize more to a consistent market value, and the executor of the arbitrages can gain profits directly linked to the pricing differences found [4].
- A second type of arbitrage technique, which is similar to the first one, also takes advantage of pricing spreads between two exchanges. In the trading bot "Blackbird," when an spread is discovered that is large enough to cover fees of short-selling and buying on the two respective exchanges, a short-sell is performed on the exchange where the price is higher, and a long is executed on the exchange where the price is lower. When the two prices eventually converge, the positions are settled (sold for the long position, and bought in the short-sell position). The net profit is basically half the spread minus the trading fees, multiplied by the volume of cryptocurrency traded [6]. This technique claims to be market-neutral in that it doesn't expose the user to any risks associated with market fluctuations. It also eliminates the need to transfer assets between

different exchanges.

- A third technique was discovered while the exploring MIT homework set solutions by Demaine and Goldwasser [7]. This third technique, ‘‘intra-exchange’’ arbitrage, is basically making a sequence of trades starting and ending with a specific currency, but you end up with more of the starting currency. The Bellman-Ford algorithm [8] can be used to discover pricing discrepancies in an exchange, and these arbitrage opportunities can then be executed by a bot. This is the type of arbitrage explored in this paper. [7]

2.3 Current Cryptocurrency Arbitrage Implementations and Challenges

Osipovich and Jeong [9] discuss the case of Stefan Qin, a 21-year-old Australian based in California, who has built a business out of cryptocurrency arbitrage. In 2016, he founded Virgil Capital, a hedge fund specializing in cryptocurrency arbitrage. He put his studies at San Francisco's Minerva Schools on hold to run the fund, which returned over 400 % last year after fees and now manages $23.5 million. [9] Qin is doing inter-exchange arbitrage, or buying on one exchange, and selling on another. This technique presents several challenges. The most significant challenge is withdrawal times. Because when we buy a currency on an exchange, and want to withdraw it to a different exchange, it needs to be processed by the blockchain and mined. Unless the miner's fee is high, it is likely that the transaction will take a significant length of time, after which the arbitrage opportunity would disappear. The other challenge is that you often need significant amounts of capital to really make any significant impact on equalizing the rates on the two target exchanges.

In an article written by ‘‘scrawl’’(username), he says that intra-exchange arbitrage, presents its own set of challenges. First off, many exchanges have a rate-limit on their API, which limits the number of API requests you can make on it for a given time period. These rate-limits vary from exchange to exchange, but they tend to start from around 60 requests a minute, and may go up to 120 requests per minute [10]. A second challenge this technique presents is the very small window in which you have to execute these arbitrages. In the FOREX market, arbitrage opportunities exist for at most one second, by which time the opportunity has already been taken advantage of [3]. A third challenge is that in cryptocurrency exchanges, a flat commission fee is charged for each trade. These fees can differ from exchange to exchange, but tend to range from 0.05% to 0.5% per trade. These are all factors that need to be taken into consideration when building an arbitrage bot [5].

From Norry's article on cryptocurrency trading bots, many implementations of trading bots currently exist on the market. "Blackbird", "Haasbot", "Zenbot", "Gekko", "CryptoTrader", and "3Commas" are a several available cryptocurrency trading bots available on the market currently.

- "Blackbird" - Blackbird is a supposedly an inter-exchange, market neutral arbitrage bot written in C++ because it actually doesn't sell, but short-sells a currency on one exchange when a significant price difference is observed. When the prices eventually equalize, it settles the position and reaps in the profit. In addition, The buy/sell and sell/buy trading activities are done in parallel on two different exchanges, independently. Advantage: no need to deal with transfer latency issues [6].
- "Hassbot" - Haasbot offers 3 different pricing plans- one for beginners at a price of 0.073BTC, one for advanced traders at 0.208BTC, and one for growing investors priced at 0.127BTC. These prices are for one year. Haasbot is written in C# and offers a many customizations for bot trading strategies. Haasbot can use technical indicators like RSI, MACD, and Fibonacci. There are proprietary safeties and insurances to keep your investments safe. There is also an auto-tune

feature you can use to optimize your trading strategy. Historical/Real time testing, advanced notifications and reporting are also available, and the platform is developer friendly: meaning you can write your own code to develop and our customize your own bots. Most importantly, the platform offers a plethora of technical indicators you can use to your advantage when developing a bot trading strategy [11].

- "Zenbot" - Zenbot is a command-line inter-exchange cryptocurrency trading bot using Node.js and MongoDB. Features it includes are: Fully-automated technical-analysis-based trading approach, full support for GDAX, Poloniex, Kraken, Bittrex, Quadriga, Gemini, Bitfinex, CEX.IO and Bitstamp; plugin architecture for implementing exchange support, or writing new strategies; simulator for backtesting strategies against historical data; "Paper" trading mode which operates on a simulated balance while watching the live market; configurable sell stops, buy stops, and (trailing) profit stops; and flexible sampling period and trade frequency - averages 1-2 trades/day with 1h period, 15-50/day with 5m period [12].
- "Gekko" - Gekko is a free and open source inter-exchange Bitcoin TA trading and backtesting platform that connects to popular Bitcoin exchanges. It is written in JavaScript and runs on Node.js. It offers a "Paper" Trader and "Tradebot" that performs actual trades. Gekko offers plug-in support and the ability to develop your own trading strategies. Gekko supports 3 different exchanges (including Bitfinex, Bitstamp and Poloniex) [13].
- "CryptoTrader" - CrytpoTrader is an cloud-based inter-exchange arbitrage bot that requires no software installation. All major crypto-currency exchanges are supported for both backtesting and live trading. It also offers a strategies marketplace where strategies can be bought and sold. It offers backtesting

trading strategies to see how a chosen strategy would work in different market conditions [14].

- “3Commas” - 3Commas is a trading platform that offers several interesting features, including concurrent stop-loss and take-profit trades. It also includes a feature called Trailing features which allows you to rake in profits at a higher threshold than you initially set. It also has long (longer time frame) and short (shorter time frame, typically day trading) bots available. It also offers a QFL bot that performs well in stable markets. The QFL trading strategy is another version of trading based on "price support" and the focus on finding "dead cat bounce" trading opportunities. It also offers a composite bot option which lets you have a list of coins you want the bot to trade and it will then manage your balance automatically. This option allows for optimal balance usage, and is much easier to use then multiple usual bots. [15].

It is not known whether or not these trading bots can discover or execute on arbitrage opportunities. Most bots come with the option to design your own trading strategies and even provide test environments to test your trading bot strategy out. However, it was noted by Norry that some of the trading bots with better reviews would require up to 0.32BTC (2048 USD at the time of this writing) per month licensing fee [5, 10].

2.4 Bellman-Ford and Asset Arbitrage

In this section, we will go over the mathematical foundations behind calculating arbitrage opportunities, which specially utilize the negative-cycle detection property of the Bellman-Ford algorithm. The Bellman-Ford algorithm is a shortest-paths algorithm similar to Dijkstra’s. It utilizes relaxation to find the shortest distance from any one node to another node in a connected graph. Note that the graph we are dealing with in this thesis is not fully-connected, e.g. not every node is connected directly with another. The nodes are typically connected through 4 main nodes, ‘‘BTC’’(Bitcoin), ‘‘ETH’’(Ethereum), ‘‘BNB’’(Binance Coin), and ‘‘USDT’’(Tether: Cryptocurrency backed 1:1 by the US Dollar- Not Verified!)

In the Homework 7 solutions by ‘‘Demaine’’ and ‘‘Goldwasser’’ [7], they presume a situation where there is a suitable weighted, directed graph G = *(V,E)*, which we form as follows. G is composed of n vertices which comprise V, this is the number of cryptocurrencies available to trade. An edge $e_{i,j}$ from v_i to v_j and an edge $e_{j,i}$ from v_j to v_i exist if there is a trading pair between two vertices, and these edges comprise E. The full set of edges, E and vertices, V, comprise our graph, G. An arbitrage opportunity is defined as a case where:

$$R[i_1, i_2] \times R[i_2, i_3] \times R[i_3, i_4] \times \cdots \times R[i_{k-1}, i_k] \times R[i_k, i_1] > 1$$

where $R[i_i, i_j]$ is the exchange rate going from a currency i to a different currency j They further note that this scenario is only true when:

$$\frac{1}{R[i_1, i_2]} \times \frac{1}{R[i_2, i_3]} \times \frac{1}{R[i_3, i_4]} \times \cdots \times \frac{1}{R[i_{k-1}, i_k]} \times \frac{1}{R[i_k, i_1]} < 1$$

Taking an advantage of the multiplicative to addition property of logarithms, they note that if you take the logarithm of both sides you can instead represent this condition as a sum:

$$\ln \frac{1}{R[i_1, i_2]} + \ln \frac{1}{R[i_2, i_3]} + \ln \frac{1}{R[i_3, i_4]} + \cdots + \ln \frac{1}{R[i_{k-1}, i_k]} + \ln \frac{1}{R[i_k, i_1]} < 0$$

Using this intuition, we can then represent our an edge weight from v_i to v_j simply as:

$$e_{i,j} = \ln \frac{1}{R[i, j]} = -\ln R[i, j]$$

[7]

Thus, this representation allows us to calculate, depending on current market exchange rates polled, all the negative cycles present and also the most profitable one. The arbitrage sequence is determined by tracing the predecessor when a negative cycle is found, and a set can be handily used to determine what the exact sequence is. Although the run-time of Bellman-Ford is $f(n) = n^3$, our n is small enough that the cubic property of the run-time does not really impact program performance; rate-limiting would by an exchange's API would be a bigger concern than the run-time of the Bellman-Ford algorithm in our program. From my experience the Binance REST API is not rate-limiting. I have polled Binance's REST API every 0.0001ms and did not get any errors.

CHAPTER 3

System Architecture

3.1 Technical Approach/Methodology

An object-oriented approach to this problem was taken. Currencies were represented as vertices and exchange rate pairs as edges. The Bellman-Ford algorithm, a single-source shortest-path algorithm that can detect negative cycles, was utilized to discover whether or not arbitrage opportunities exist on the Binance exchange, and when they did show up, an execution on the most profitable one was made. Steps to implement this are detailed below:

1. Poll the REST API Binance to obtain current active trading pairs and their respective current exchange rates.

2. Dynamically generate the vertices and edges from data obtained in step 1.

3. A graph is constructed from the edges and vertices from step 2. Bellman-Ford will be executed on these edges and vertices.

4. Store the highest profitable trade sequence generated from Bellman-Ford as our "best arbitrage sequence".

5. Execute trades on the "best arbitrage sequence" stored in previous step with the Binance API.

6. Write following values to `.csv` file immediately after trade, which includes:

    - Time-stamp
    - Arbitrage sequence
    - Portfolio BTC balance
    - Snapshot value (at current exchange rates- exchangeRatesMid)
    - Actual portfolio value (calculated with exchangeRatesMid)

- ‘‘amountBuffer’’ value
- ‘‘sigDigBuffer’’ value
- Exchange (Binance in this project)

7. Repeat previous steps indefinitely.

The steps outlined above should steady stream of profits while making the Binance exchange more liquid and efficient.

3.2 Implementation

This project was implemented in Java(1.8+).

3.2.1 Program Setup (List of classes and other files used)

- Main - Used to execute interaction with Binance's REST API. Has static HashMaps of exchange rates, code that interacts and extracts data from the API for construction of our graph for use by Bellman-Ford. All dialogs for user interaction are also in the Main class.
- Trader - Used for calculating account balances (without arbitrage and holding), doing actual currency conversion from either BTC/ETH to all others, or the other way around, determining amount and pricing for each trade in a arbitrage sequence, and executing trades. Also used for taking account snapshots.
- ShouldTrade - This is a class used by the Trader class to filter out trades that had an amount requiring a precision of significant digits < 2. I tried higher values than 2 but no trades were executed. This makes sense because most pairs had a requirement of 2 or less for amount precision.
- Trade - Used to return a LimitOrder object provided by XChange's library. This LimitOrder object has a price that is determined by the best sell price if we are "buying", and the best "bid" price if we are selling. The amount is determined through a dialog when the program first runs, but I typically

used 0.0025BTC $\approx$ \$16USD at the time of this writing. 1BTC $\approx$ \$6400USD in November, 2018. I chose this value of Bitcoin to use because it is slightly over the minimum order value, which is approximately \$10 USD. We want to use the least amount of BTC available, so that all our orders will fill immediately and not be left as an open order to be filled later. As the total value of a trade goes up, the likelihood that it will not be filled immediately goes up. I noticed good results with about \$15 USD. It is interesting to note, however, that all my open trades did fill. Some took a few months, but they all did complete.

- Vertex(v_i)- Used to model each cryptocurrency as a node in a graph
- Edge($e_{i,j}$) - Used to model exchange rates between these cryptocurrency nodes. *Note that when creating the Edge class, we use the best buy price for going from one currency to another (e.g. BTCUSDT symbol, going from BTC to USD), since that would be the price we could immediately sell BTC for USDT at. Conversely, we would use the best sell price when buying a currency with another, since that would be the cheapest we could immediately buy BTC with USDT in this example at.(e.g. BTCUSDT symbol, going from USDT to BTC)
- Graph(G) - Used to model the collection of Vertices and Edges, and execute the Bellman-Ford algorithm. Also used to find the negative cycle with the largest weight (highest profitability), and find the path associated with that cycle. When tracing the route with predecessor, remember to reverse the route since we are tracing backwards.
- CurrencyConverter - Used to convert currencies between each other at current exchange rates, and ensure correct precision (number of significant digits) in price and quantity fields when creating limit orders. I read the number of significant digits from a .csv file but I know there is a REST API endpoint provided by Binance that provides this data.

- Utilities - Used to log data in results `.csv` file.
- BinanceTradingRule-Master.csv - used to store the precision of quantity
- BinanceTradingRule-MinPrice.csv - used to store the precision for pricing
- binanceConfig.properties - This file stores your apiKey and apiSecret, make sure not to push it to Github! (i.e. store it outside your project folder)

3.2.2 Libraries Used

- XChange by Knowm - This was a very handy library that provided easy to use classes and builder methods for interaction with Binance's API. They have implemented their library for more than 60 exchanges, and their code can be viewed on Github at XChange Github Repository.
- GSON - a JSON interaction library by Google. This library made it easy to parse and interact with JSON data returned by Binance's REST API.
- Unirest - a simple Java library for making HTTP verb requests, such as GET and POST in this project.
- XChange-stream - a Java libary allowing for interaction with websockets. Allows you to subscribe to updates in different websocket channels, such as ticker, trades, and orderbook, and also unsubscribe from a channel.

3.2.3 Miscellaneous Data Structures Used

- ArrayList<Vertex> vertices - Used to store all the cryptocurrencies available on the Binance exchange.
- ArrayList<Edge> edges - Used to store all the edges and their weights calculated for use in Bellman-Ford.
- HashMap<String, Double> exchangePrices - This would store the prices for an instant sell (highest buy), and instant buy (lowest sell).
- HashMap<String, Double> exchangeRates - This was used by the Currency-

Converter class to make conversion rates between currencies easier and more intuitive.

- HashMap<String, Double> exchangeRatesMid - This was used to get an accurate portolio value. I notices that I couldn't use exchangeRates because it would often underestimate my account value. When I averaged the lowest "ask" and highest "sell", and stored it in exchangeRatesMid, I obtained a much more accurate picture of my portfolio value. This was verified by checking with my portfolio balance on Binance.
- LinkedHashMap<String,Integer> sigDigs - This was used to store the precision required for the quantity when creating LimitOrder objects to execute.
- LinkedHashMap<String,Integer> sigDigsForPricing - This was used to store the precision required for the price when creating LimitOrder objects to execute.
- HashMap<String, Edge> edgeMap - This was used in construction of edges
- HashMap<String, Vertex> vertexMap - This was used in construction of vertices. HashSet<Vertex> setOfVertices - This was used in construction of vertices. HashSet<Edge> setOfEdges - This was used in construction of edges
- Properties prop - used to store apiKey and apiSecret for authentication to Binance REST API.

CHAPTER 4

Experiments and Results

In the table below, it shows the ending amount of bitcoins after each arbitrage sequence is executed. I chose to execute trades starting and ending with bitcoin because I felt this strategy would be a good way to verify whether the arbitrage was working or not. As you can see from the table, the average amount of bitcoin increased a little less than 0.1% after each arbitrage sequence. This makes sense, given that the trading commission Binance takes is approximately 0.05% for each trade made. This commission was calculated before depending on the length of the trade sequence. For example, for 4 trades, approximately 0.2% (4 * 0.05) would be initially added as commission to the potential arbitrage; we will call this “potential arbitrage”. After adding the commission to the arbitrage ratio opportunity, an additional parameter “amountBuffer” was then added on top of this “potential arbitrage”, to compensate for the limitations on precision for the price and amount fields imposed by the Binance REST API. Different values were tested for “amountBuffer” and the results will be shown below. “AmountBuffer” is an integer value used in the Trader class, though decimal values would work too; it adds a buffer on top of the commission for a wider arbitrage profit “space”. Through my experimentation, I discovered that Binance is a highly liquid cryptocurrency exchange. However, arbitrage opportunities still do exist and can be capitalized on.

4.1 Bitcoin Only Experiment

After verifying that the arbitrage sequence is correct in execution, I removed the “BTC” only filter so that I would be able to execute on significantly more opportunities.

I collected data using different values for “amountBuffer” and “sigDigBuffer”. “amountBuffer” refers to the buffer on top of the estimation of the trading fees Binance

BItcoin With Arbitrage	Additional Bitcoins	Percent Increase	Increase Ratio from Beginning
0.03130317	0	0	1
0.03134911	0.00004594	0.147	1.001467583
0.0313673	0.00001819	0.058	1.002048674
0.03140468	0.00003738	0.119	1.003242803
0.03141825	0.00001357	0.0432	1.003676305
0.03142991	0.00001166	0.0371	1.004048791
0.03144331	0.0000134	0.0426	1.004476863
0.03145775	0.00001444	0.0459	1.004938158
0.03146342	0.00000567	0.018	1.00511929
0.03147615	0.00001273	0.0405	1.005525958
0.03149407	0.00001792	0.0569	1.006098424
0.03156073	0.00006666	0.212	1.008227921
0.03160409	0.00004336	0.137	1.009613084
0.0316491	0.00004501	0.142	1.011050957
0.03168059	0.00003149	0.0995	1.012056926
Average Percent Increase Per Trade:		0.08562142857	

Figure 1: Table showing Additional Bitcoins, Percent Increase, and Percent Increase from Beginning vs Arbitrage Executions.

takes on each arbitrage sequence, which is dependent on the arbitrage sequence length. Different values for “amountBuffer” were used, ranging from 1 to 6. “sigDigBuffer” refers to precision we would like to have on our trades. For ‘sigDigBuffer”, a value of 2 would mean that any symbol with a precision less than 2, ie. 1 and 0, would be filtered out. Of course, we would want to increase the precision of our arbitrage execution, but by increasing the “sigDigBuffer”, but the number of arbitrage opportunities found is decreased. By decreasing “sigDigBuffer” to 1 or 0, the number of arbitrages executed increase. The precision of the arbitrage is directly proportional to “sigDigBuffer”. Increasing “sigDigBuffer” past of a value of 2 results in nearly no trades being executed. Hence, for Binance, a “sigDigBiffer” of 2 is necessary to execute intra-exchagne arbitrage. The tables below show the results for runs with different values for “amountBuffer” and “sigDigBuffer”. Note that these results are specific to Binance and would likely be different for a different exchange.

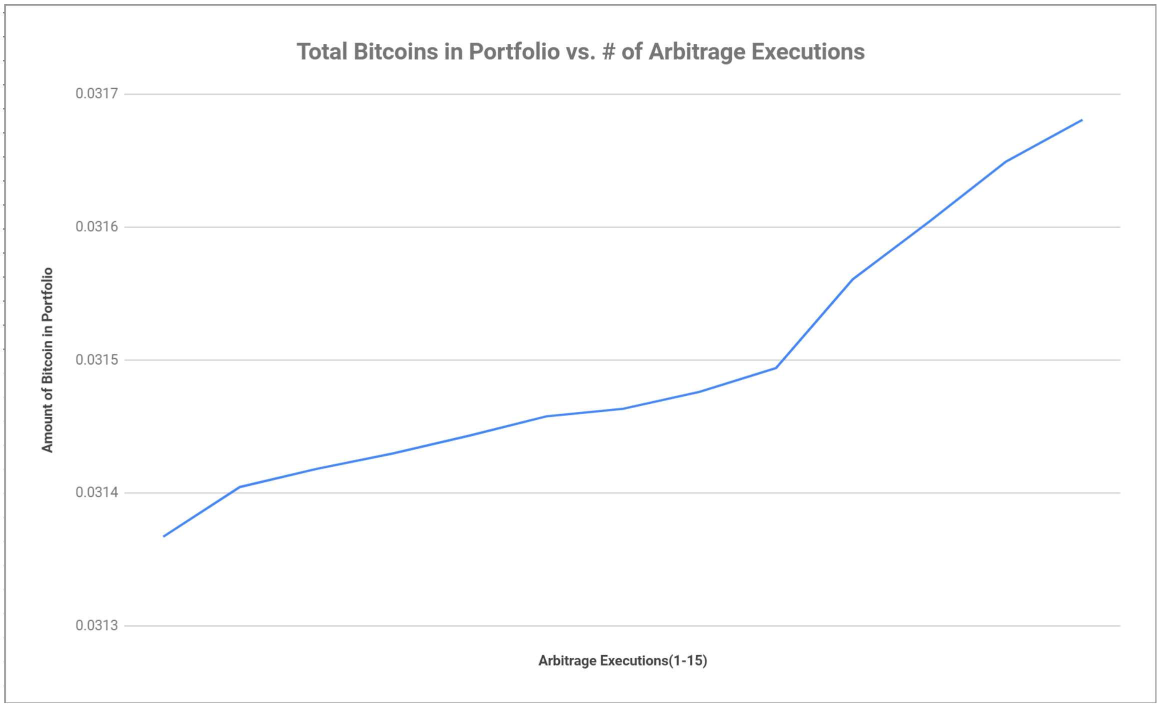

Figure 2: Total Account Bitcoin Value vs. Arbitrage Sequences Executed.

After verifying that my arbitrage logic was correct, I decided to remove my BTC only filter and to execute on all arbitrage sequences. An interesting question came into mind: *How would I capture whether or not I was making profits through arbitrage if I was executing on all coins?* To address this issue, I decided to take a snapshot of my balances at a certain point in time, consisting of a HashMap of coins and their respective balances. I would then convert all those coins into Bitcoin at the current market rate on the exchange, using a data structure called exchangeRatesMid (This is the average of the best sell and best buy price). This calculation gave me a reasonable estimate of my account value if I held onto my coins and had **not** executed arbitrage. I could then compare my actual Binance account value, using the same rates stored in exchangeRatesMid, to this snapshot value to gauge whether or not I was gaining or losing value through executing arbitrages. In using this methodology, I could also negate market movements of the highly volatile cryptocurrency market and obtain a neutral picture of the success of my program.

The following section shows 6 tables and graphs demonstrating the impact of "amountBuffer" on the profitability of my arbitrage program. The line graphs shown simply shows whether or not my portfolio value increases or decreases depending on the value of "amountBuffer". The following table is for "amountBuffer" = 1.

4.2 "amountBuffer"= 1

Date and Time	Arbitrage Sequence	Portfolio Value (Arbitrage)	Portfolio Value (Held)	Difference	Difference Adjusted to 0
2018/09/06 12:26:04	BTC ZEN ETH STRAT BTC	2549.570718	2519.160255	30.41046326	0
2018/09/06 12:28:41	BTC QTUM ETH BTC	2544.537583	2513.466547	31.07103677	0.6605735043
2018/09/06 12:28:53	BTC HC ETH NANO BTC	2547.842267	2517.044043	30.79822396	0.3877606979
2018/09/06 12:32:01	BTC ZEN ETH KMD BTC	2549.510824	2517.286848	32.22397567	1.813512404
2018/09/06 12:33:15	BTC MTL ETH USDT BTC	2549.485938	2517.404268	32.08166949	1.671206231
2018/09/06 12:36:24	BTC ZEN ETH MTL BTC	2549.504876	2518.010168	31.49470845	1.084245184
2018/09/06 12:36:45	BTC MTL ETH PIVX BTC	2557.934204	2527.417775	30.51642954	0.1059662801
2018/09/06 12:39:10	BTC ONT ETH ZEN BTC	2559.481107	2528.560265	30.92084208	0.5103788214
2018/09/06 12:43:38	BTC WAN ETH REP BTC	2557.891455	2526.190459	31.70099519	1.290531925
2018/09/06 12:44:54	BTC WAN ETH USDT BTC	2555.748352	2525.664834	30.08351815	-0.3269451107
2018/09/06 12:45:22	BTC AE ETH USDT QTUM BTC	2556.922434	2527.031238	29.89119584	-0.519267427
2018/09/06 12:45:24	BTC NEBL ETH USDT BTC	2558.922126	2529.82369	29.09843639	-1.312026874
2018/09/06 12:45:30	BTC NEO ETH SALT BTC	2561.922984	2533.259376	28.66360706	-1.746856202
2018/09/06 12:46:03	BTC KMD ETH USDT BTC	2563.359048	2535.351148	28.00789916	-2.402564104
2018/09/06 12:46:04	BTC EDO ETH USDT BTC	2566.969498	2538.927275	28.04222299	-2.368240276
2018/09/06 12:46:24	BTC VIA ETH BTC	2567.210485	2539.142322	28.06816329	-2.342299975
2018/09/06 12:56:13	BTC MCO ETH TRIG BTC	2567.146222	2536.020108	31.12611478	0.7156515206
2018/09/06 12:56:44	BTC ZEN BNB LTC USDT ETH REP BTC	2566.350301	2536.52123	29.82907035	-0.5813929085
2018/09/06 12:58:21	BTC ICX USDT BCC BTC	2568.426229	2537.428829	30.99740021	0.5869369439
2018/09/06 13:03:03	BTC ZEN BNB BTC	2568.862192	2537.786377	31.07581448	0.6653512182
2018/09/06 13:05:19	BTC ZEN ETH BTC	2573.518887	2542.734366	30.78452138	0.3740581153
2018/09/06 13:05:29	BTC ZEN ETH SKY BTC	2569.872481	2539.690779	30.18170216	-0.2287611005
2018/09/06 13:06:04	BTC ZEN ETH USDT BTC	2570.829893	2540.453969	30.37592392	-0.03453934756
2018/09/06 13:06:24	BTC ZEN ETH BTC	2570.624428	2540.354812	30.26961644	-0.1408468212
2018/09/06 13:07:04	BTC TRIG ETH BTC	2572.209243	2542.429356	29.77988686	-0.6305764011
2018/09/06 13:07:11	BTC TRIG ETH CLOAK BTC	2571.719366	2541.631005	30.0883605	-0.3221027604
2018/09/06 13:10:27	BTC RLC ETH MTL BTC	2571.748937	2541.287186	30.46175067	0.05128741175
2018/09/06 13:11:08	BTC RLC ETH KMD BTC	2571.934511	2541.507325	30.42718617	0.01672290808
2018/09/06 13:14:57	BTC ZEN ETH BTC	2572.564017	2541.587028	30.97698983	0.5665265677
2018/09/06 13:31:19	BTC KMD ETH LUN BTC	2570.611486	2539.898843	30.71264373	0.3021804698
2018/09/06 13:35:35	BTC ZEN BNB ETH WAN BTC	2567.783591	2537.571554	30.21203638	-0.1984268843
2018/09/06 13:37:28	BTC ARK ETH GVT BTC	2561.365385	2532.047661	29.31772418	-1.092739084
2018/09/06 13:42:55	BTC TRIG BNB ETH USDT BTC	2559.658748	2530.248274	29.41047398	-0.9999892859
2018/09/06 13:43:00	BTC ZEC ETH LSK BTC	2556.674952	2527.97399	28.70096133	-1.709501929
2018/09/06 13:44:10	BTC BTG ETH BTC	2557.222613	2530.772655	26.44995826	-3.960505004
2018/09/06 13:46:09	BTC PIVX ETH USDT NEO BTC	2559.152843	2533.3147	25.83814307	-4.572320191
2018/09/06 13:47:17	BTC EDO ETH BTC	2561.713264	2536.255961	25.45730246	-4.953160806
2018/09/06 13:48:59	BTC STRAT ETH XMR BTC	2561.60788	2536.506415	25.10146579	-5.308997474
2018/09/06 13:49:02	BTC STRAT ETH XMR BTC	2562.288347	2536.908844	25.37950281	-5.030960454
2018/09/06 13:50:45	BTC ONT ETH ZEN BTC	2563.279697	2538.262159	25.01753816	-5.392925101
2018/09/06 13:51:08	BTC LUN ETH USDT BNB NANO BTC	2563.265471	2538.277795	24.98767559	-5.422787669

Figure 3: Table showing Profitability with amountBuffer = 1.

As you can see from the table and graph below, an actual loss is shown with "amountBuffer" = 1. A higher frequency of arbitrages is noticed, but the profitability is most likely due to loss in precision due to the limitations imposed by Binance's REST

API on both the price and amount. After executing nearly 50 arbitrage sequences with an average amount of 0.002 BTC, I noticed a loss of approximately 6USD equivalent, so I decided to move on to higher values of "amountBuffer". The initial difference in portfolio value is due to previous experiments with arbitrage at higher values that had resulted in profit. I went back to an "amountBuffer" of 1 to log data in order to verify that it is indeed a losing proposition.

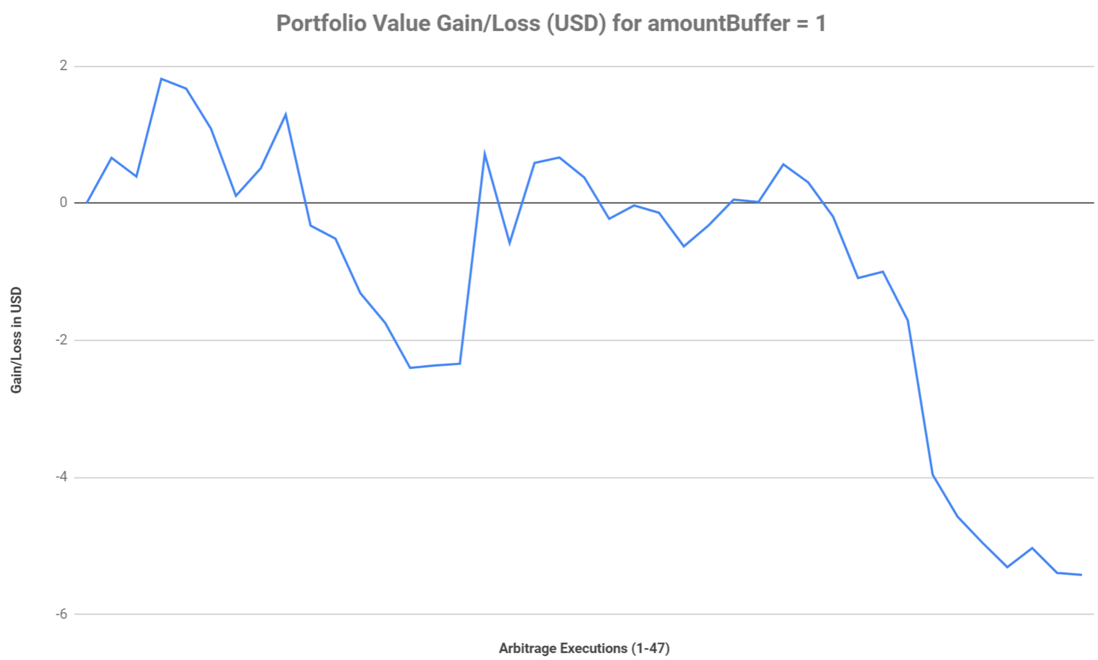

Figure 4: Graph showing arbitrage results with amountBuffer = 1.

4.3 "amountBuffer" = 2

Date and Time	Arbitrage Sequence	Portfolio Value (Arbitrage)	Portfolio Value (Held)	Difference	Difference Adjusted to 0
2018/09/06 14:34:57	BTC PPT ETH KMD BTC	2553.646688	2530.252937	23.39375153	0
2018/09/06 14:42:32	BTC DASH ETH REP BTC	2549.41217	2524.977961	24.43420849	1.040456956
2018/09/06 15:00:18	BTC NAV ETH XMR BTC	2551.922806	2526.491793	25.43101268	2.037261144
2018/09/06 15:09:24	BTC PPT ETH BTC	2558.567485	2534.087802	24.4796835	1.085931969
2018/09/06 15:18:04	BTC DASH ETH ZEN BTC	2556.590846	2533.492606	23.09824076	-0.2955107714
2018/09/06 15:26:31	BTC NAV ETH DGD BTC	2556.604474	2532.22934	24.37513395	0.9813824144
2018/09/06 15:27:33	BTC SALT ETH USDT NEO BTC	2556.777087	2532.456394	24.32069298	0.9269414509
2018/09/06 15:32:50	BTC EDO ETH NEO BTC	2557.418173	2533.353082	24.06509066	0.6713391225
2018/09/06 15:33:14	BTC MCO ETH ETC BTC	2557.430404	2533.566379	23.86402445	0.4702729152

Figure 5: Table showing Profitability with amountBuffer = 2.

The table above shows experiments with an "amountBuffer" = 2. Results were similar to "amountBuffer" = 1, but with an actual small gain of approximately 0.5USD after just 10 arbitrage executions. The reason for so little data on "amountBuffer" = 2 is because I found the program was much more profitable at higher values of "amountBuffer". The graph below shows the table in a line bar format.

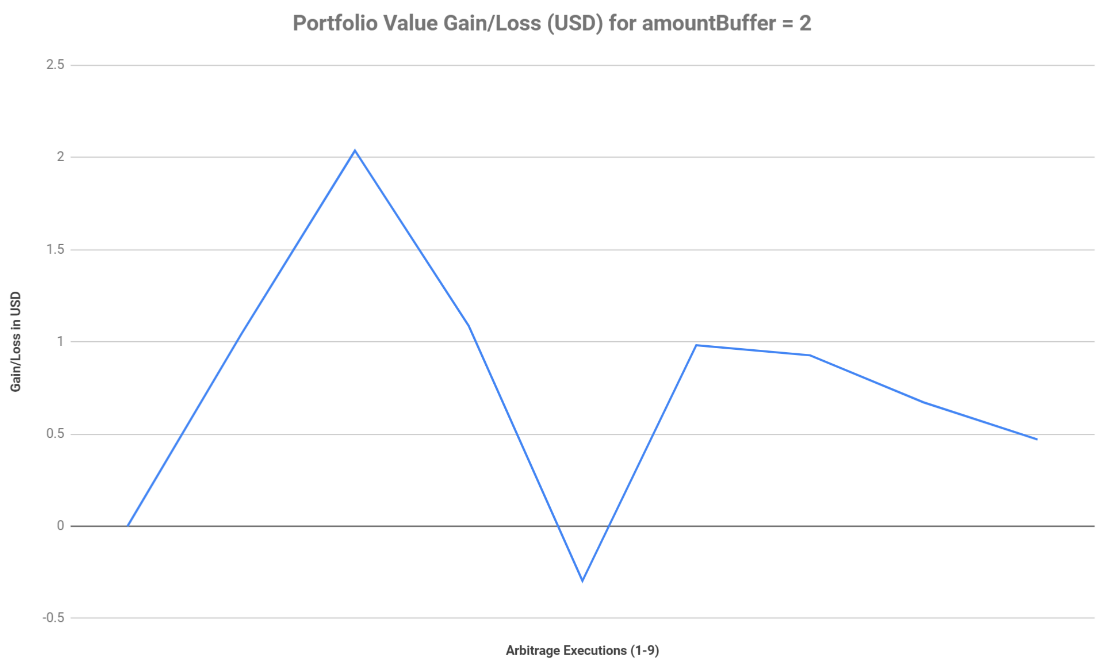

Figure 6: Graph showing arbitrage results with amountBuffer = 2.

4.4 "amountBuffer" = 3

The table above shows experimental results with an "amountBuffer" value of 3, I started noticing some actual gains with respect to arbitrage execution. After 20 arbitrage executions, a net gain of approximately 7.5USD was noticed. This was exciting and I decided to bump up the value of "amountBuffer" even higher. The graph shown below shows the results from using an "amountBuffer" value of 3.

Date and Time	Arbitrage Sequence	Portfolio Value (Arbitrage)	Portfolio Value (Held)	Difference	Difference Adjusted to 0
2018/09/06 15:40:40	BTC GVT ETH USDT BTC	2557.337769	2533.86077	23.47699906	0
2018/09/06 15:55:10	STRAT ETH ONT BTC STRAT	2566.890903	2543.406618	23.48428424	0.00728517909
2018/09/06 16:02:38	XZC ETH CLOAK BTC XZC	2579.353635	2555.548115	23.80552038	0.3285213195
2018/09/06 16:05:35	BTC MTL ETH PPT BTC	2586.524508	2562.359072	24.16543582	0.6884367674
2018/09/06 16:06:05	BTC SKY BNB USDT QTUM BTC	2587.139312	2563.010245	24.12906686	0.6520678002
2018/09/06 16:07:23	MCO BTC BNB MCO	2591.926596	2567.56516	24.36143608	0.8844370219
2018/09/06 16:11:01	ETH CLOAK BTC ARK ETH	2586.785859	2562.217668	24.56819106	1.091191999
2018/09/06 16:15:24	ETH DGD BTC INS ETH	2589.330843	2565.589769	23.74107423	0.2640751769
2018/09/06 16:21:36	NEBL BTC PPT ETH NEBL	2586.129511	2562.916275	23.21323626	-0.2637627947
2018/09/06 16:26:04	ETH ICX BTC HC ETH	2591.750205	2568.118682	23.63152262	0.1545235666
2018/09/06 16:28:29	GVT BTC WAVES ETH GVT	2594.243736	2569.48922	24.75451589	1.277516834
2018/09/06 17:04:08	ETH WTC BTC GVT ETH	2605.422392	2574.553385	30.86900734	7.392008282
2018/09/06 17:07:43	NEO ETH RLC BTC NEO	2599.118789	2568.742848	30.37594071	6.898941655
2018/09/06 17:08:43	USDT NEO BNB USDT	2592.88995	2560.203053	32.68689662	9.209897564
2018/09/06 17:15:16	BTC ONT ETH MTL BTC	2595.061354	2563.286756	31.77459792	8.297598865
2018/09/06 17:24:06	BTC STEEM ETH BCD BTC	2605.347886	2574.59171	30.7561755	7.279176446
2018/09/06 17:27:40	ZEC ETH QTUM BTC ZEC	2600.718504	2569.344523	31.37398085	7.896981792
2018/09/06 17:30:09	NEBL BTC BNB NEBL	2605.561493	2574.151567	31.40992561	7.932926549
2018/09/06 18:00:11	BNB EOS BTC WAVES BNB	2611.09653	2580.088571	31.00795918	7.530960121

Figure 7: Table showing Profitability with amountBuffer = 3.

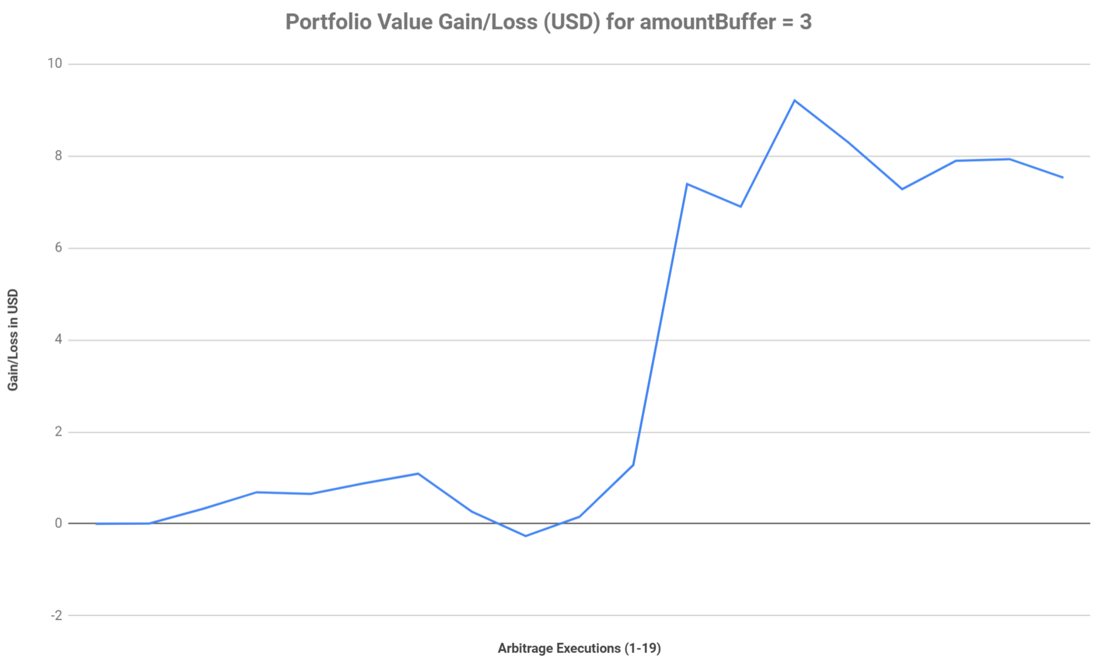

Figure 8: Graph showing arbitrage results with amountBuffer = 3.

4.5 "amountBuffer" = 4

The tables below show my experimental results of 3 runs with an "amountBuffer" value of 4. and the tables and graphs are listed in sequence. I noticed that at this value, the profitability of my program was at its highest. In Run 1, after nearly 70 executions, the program had made a net profit of approximately 20USD equivalent.

Date and Time	Arbitrage Sequence	Portfolio Value (Arbitrage)	Portfolio Value (Held)	Difference	Difference Adjusted to 0
2018/09/06 20:09:55	USDT BTC ICX USDT	2625.576318	2598.362656	27.21366176	0
2018/09/06 20:23:55	BTC DASH ETH NAS BTC	2620.846632	2592.56316	28.28347171	1.069809945
2018/09/06 20:27:09	ETH NANO BNB BTC WTC ETH	2617.38395	2587.869326	29.51462356	2.300961798
2018/09/06 20:32:45	BTC ICX ETH WAN BTC	2621.114406	2592.979056	28.13534923	0.921687463
2018/09/06 20:35:36	WTC ETH WAN BTC WTC	2622.602972	2595.009983	27.59298929	0.3793275301
2018/09/06 20:44:39	TRIG BTC GXS ETH TRIG	2626.223985	2595.964273	30.25971163	3.046049865
2018/09/06 21:07:26	NANO BNB AION ETH NEO BTC NANO	2607.60463	2576.344069	31.26056094	4.046899176
2018/09/06 21:48:19	BTC XZC BNB BTC	2608.633328	2576.844135	31.78919343	4.57553167
2018/09/06 21:56:29	BTC PPT ETH MCO BTC	2612.398381	2581.094291	31.30408947	4.090427705
2018/09/06 21:58:41	ETH VIA BTC ONT ETH	2610.577603	2578.265874	32.31172917	5.098067402
2018/09/06 22:58:18	LUN ETH GVT BTC LUN	2626.008443	2594.944085	31.06435769	3.850695924
2018/09/07 0:30:56	RLC BTC MCO ETH RLC	2588.651873	2551.733117	36.91875629	9.705094525
2018/09/07 0:31:00	RLC BTC ETH RLC	2589.376305	2552.429999	36.94630587	9.732644111
2018/09/07 0:37:05	RLC BTC NEO ETH RLC	2590.098213	2551.18694	38.91127385	11.69761209
2018/09/07 0:44:29	USDT BTC NEO USDT	2586.086373	2547.043752	39.04262094	11.82895917
2018/09/07 1:07:09	BTC KMD ETH VIA BTC	2578.467186	2539.878771	38.58841526	11.3747535
2018/09/07 1:08:31	BTC BNB USDT ETH VIA BTC	2576.814438	2537.766235	39.04820333	11.83454157
2018/09/07 1:13:39	BNB USDT QTUM BNB	2561.498262	2524.635022	36.86324059	9.649578828
2018/09/07 1:12:47	BTC USDT EOS BTC	2567.513112	2529.853909	37.65920324	10.44554147
2018/09/07 1:15:06	ETH NXS BTC USDT ONT ETH	2556.956426	2518.047442	38.90898426	11.6953225
2018/09/07 1:15:12	USDT ICX BTC QTUM USDT	2556.568943	2518.377181	38.19176191	10.97810014
2018/09/07 1:17:57	USDT QTUM ETH USDT	2554.199118	2519.353986	34.84513263	7.631470865
2018/09/07 1:19:08	ETH ZEC BTC BCD ETH	2549.905103	2513.749951	36.15515191	8.941490142
2018/09/07 10:57:28	AION ETH BTC AION	2528.386644	2483.246057	45.14058695	17.92692519
2018/09/07 12:37:11	ETH WTC BTC VIA ETH	2520.049909	2471.898718	48.15119084	20.93752908
2018/09/07 13:10:35	BNB BTC AE BNB	2520.225661	2473.815427	46.41023353	19.19657177
2018/09/07 13:17:57	AION BTC BTG ETH AION	2521.970121	2474.876843	47.09327744	19.87961568
2018/09/07 13:46:37	ZEN ETH BTC ZEN	2524.475724	2477.723135	46.75258836	19.53892659
2018/09/07 13:50:04	BTC CLOAK ETH BTC	2527.242725	2480.869614	46.37311115	19.15944938
2018/09/07 13:50:30	SALT BTC OMG ETH SALT	2531.178297	2484.42793	46.75036683	19.53670506
2018/09/07 14:11:17	AE ETH USDT ONT BTC AE	2529.208839	2485.965405	43.24343371	16.02977195
2018/09/07 14:17:09	BTC INS ETH STEEM BTC	2529.686148	2484.949642	44.73650649	17.52284473
2018/09/07 14:23:17	PIVX ETH BTC PIVX	2531.582116	2486.397328	45.18478757	17.97112581
2018/09/07 14:57:22	BTC MTL ETH XZC BTC	2524.668629	2479.883318	44.78531105	17.57164929
2018/09/07 15:09:27	BCD ETH ONT BTC BCD	2521.114028	2474.477666	46.63636207	19.42270031
2018/09/07 15:10:18	ETC BTC BCD ETH ETC	2522.100347	2475.182904	46.91744249	19.70378073
2018/09/07 15:11:15	ETH BTG BTC MTL ETH	2520.377598	2473.657082	46.72051631	19.50685454
2018/09/07 15:17:54	BTC ETH AION BTC	2514.751739	2468.21545	46.53628808	19.32262631
2018/09/07 15:18:01	ETH AION BTC QTUM ETH	2515.133807	2468.580357	46.55345054	19.33978878
2018/09/07 15:23:46	ETH MTL BTC QTUM USDT ETH	2515.366815	2468.779427	46.58738866	19.37372689
2018/09/07 16:38:57	HC BTC ARK ETH HC	2508.8857	2463.05502	45.83067966	18.6170179
2018/09/07 16:39:07	HC BTC NEO ETH HC	2508.248427	2462.495746	45.75268116	18.5390194
2018/09/07 16:56:22	DASH BTC XMR ETH DASH	2502.406353	2457.861605	44.54474888	17.33108712
2018/09/07 16:56:24	ONT USDT BTC XMR ETH ONT	2502.75744	2458.220756	44.53668365	17.32302188
2018/09/07 17:29:18	BTC PPT ETH ETC BTC	2524.048692	2479.505753	44.54293818	17.32927641
2018/09/07 17:39:51	ETH BTC HC ETH	2531.541164	2489.52272	42.01844443	14.80478266
2018/09/07 19:31:56	BTC LUN ETH USDT BTC	2529.349871	2487.118428	42.23144237	15.01778061
2018/09/07 19:54:41	BNB BTC NANO BNB	2530.383643	2488.555444	41.82819872	14.61453696
2018/09/07 20:04:33	BTC SKY ETH USDT NEO BTC	2526.660521	2485.762242	40.89827966	13.68461789
2018/09/07 20:05:09	USDT BTC LSK ETH USDT	2527.159397	2486.220063	40.93933387	13.72567211
2018/09/07 20:07:23	BTC ZEC ETH BTC	2527.624206	2486.889497	40.73470888	13.52104711
2018/09/07 20:14:45	REP BTC SKY ETH REP	2527.745906	2486.954692	40.79121429	13.57755252
2018/09/07 20:53:15	BTC NEBL ETH NEO BTC	2518.792214	2476.250522	42.54169253	15.32803076
2018/09/07 20:53:16	SKY ETH BTC SKY	2518.675399	2476.174975	42.50042352	15.28676176
2018/09/07 21:56:18	ETH BTC MTL ETH	2516.122418	2470.706333	45.41608448	18.20242272
2018/09/07 21:57:19	AION BTC LUN ETH AION	2514.69288	2470.160307	44.5325732	17.31891143
2018/09/07 22:27:26	ETH BTC TRIG ETH	2515.813884	2470.193377	45.62050717	18.40684541
2018/09/07 22:27:26	TRIG ETH BTC TRIG	2515.851583	2470.210458	45.64112503	18.42746327
2018/09/07 22:28:13	MTL ETH BTC MTL	2515.625284	2469.950156	45.67512762	18.46146586
2018/09/07 22:30:36	ZEC ETH NANO BTC ZEC	2517.376141	2471.577351	45.79878951	18.58512774
2018/09/07 22:37:30	ETH BNB BTC DGD ETH	2523.179947	2479.302584	43.87736272	16.66370096
2018/09/07 23:50:12	BTC SKY ETH NEO USDT BTC	2524.676806	2475.209091	49.46771496	22.25405319
2018/09/08 0:10:33	ETH NANO BTC LUN ETH	2528.431334	2481.163991	47.26734363	20.05368187
2018/09/08 0:19:08	USDT BTC PPT ETH USDT	2528.892812	2480.514758	48.37805408	21.16439231
2018/09/08 1:07:03	INS ETH VIA BTC INS	2536.4321	2488.962457	47.46964286	20.25598109
2018/09/08 1:13:53	ZEC ETH BTC ZEC	2537.700256	2490.379599	47.32065706	20.1069953
2018/09/08 1:21:55	ZEC BTC RLC ETH ZEC	2543.96167	2497.343259	46.61841147	19.4047497

Figure 9: Table showing Profitability with amountBuffer = 4 (Run 1).

This results of Run 1 are clearly shown in the graph below. Run 2's table and graphs are shown further below, with a net gain of almost 6USD. Run 3 is displayed lastly; over 26 arbitrage executions, a net portfolio value increase of approximately 6USD was recorded.

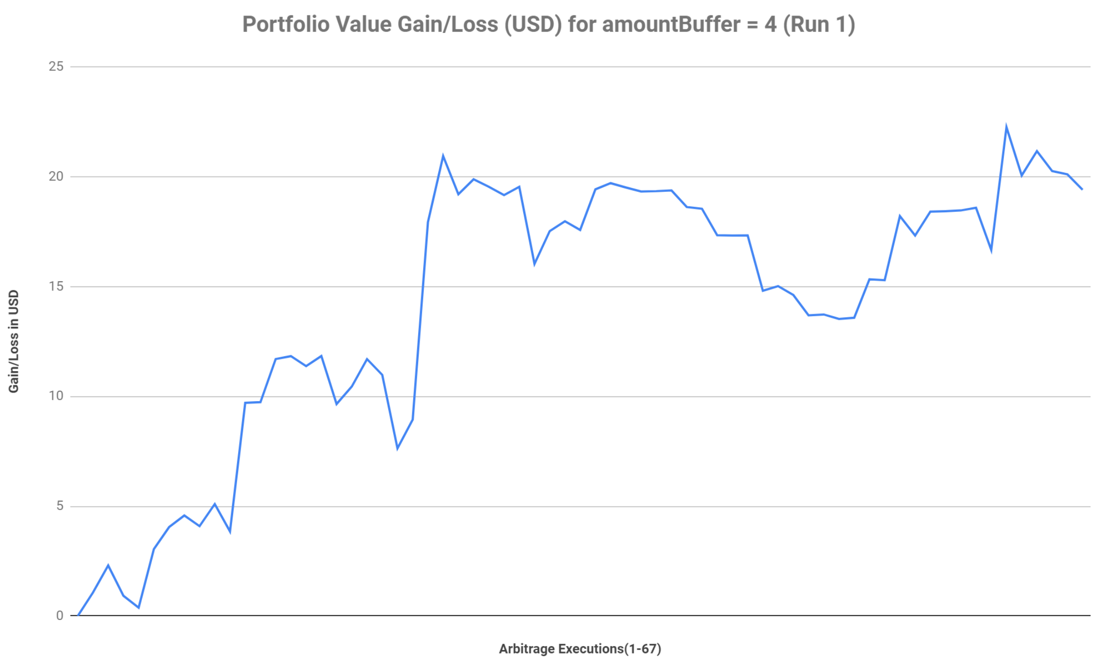

Figure 10: Graph showing arbitrage results with amountBuffer = 4 (Run 1).

Date and Time	Arbitrage Sequence	Portfolio Value (Arbitrage)	Portfolio Value (Held)	Difference	Difference Adjusted to 0
2018/09/10 14:59:52	EOS BTC ZEN ETH EOS	2383.417756	2300.403799	83.01395722	0
2018/09/10 15:44:48	ETH USDT ONT ETH	2374.910912	2292.453679	82.45723366	-0.5567235563
2018/09/10 15:45:36	BTC MTL ETH BCD BTC	2374.685034	2292.097969	82.587065	-0.4268922227
2018/09/10 15:46:51	BTC MTL ETH USDT BTC	2377.348023	2295.639939	81.70808433	-1.305872888
2018/09/10 15:49:51	BTC TRIG ETH LTC BTC	2378.306616	2296.25068	82.05593598	-0.9580212376
2018/09/10 15:50:36	TRIG ETH BCD BTC TRIG	2379.207408	2297.09895	82.10845884	-0.9054983844
2018/09/10 16:02:53	ONT BTC USDT ONT	2378.666992	2294.175789	84.49120323	1.47724601
2018/09/10 18:49:37	BNB WAVES ETH NAV BTC USDT BNB	2404.026442	2322.275622	81.7508198	-1.263137421
2018/09/10 18:59:35	BNB NANO BTC USDT BNB	2413.575337	2330.912059	82.66327803	-0.3506791927
2018/09/10 19:03:01	BTC WAVES ETH TRIG BTC	2416.07138	2331.262239	84.80914046	1.795183244
2018/09/10 19:10:35	BTC DASH ETH XMR BTC	2413.040129	2328.537036	84.50309288	1.489135655
2018/09/10 19:55:59	BTC ETH SKY BTC	2408.436498	2320.549278	87.88721937	4.873262149
2018/09/10 20:26:25	BCC ETH WAVES BNB BTC BCC	2409.195742	2320.332279	88.86346258	5.849505365
2018/09/10 21:03:50	BTC EOS BNB BTC	2404.382492	2316.551145	87.83134743	4.817390206
2018/09/10 21:14:49	BTC WAVES ETH BTC	2402.203444	2313.456619	88.74682438	5.732867161

Figure 11: Table showing Profitability with amountBuffer = 4 (Run 2).

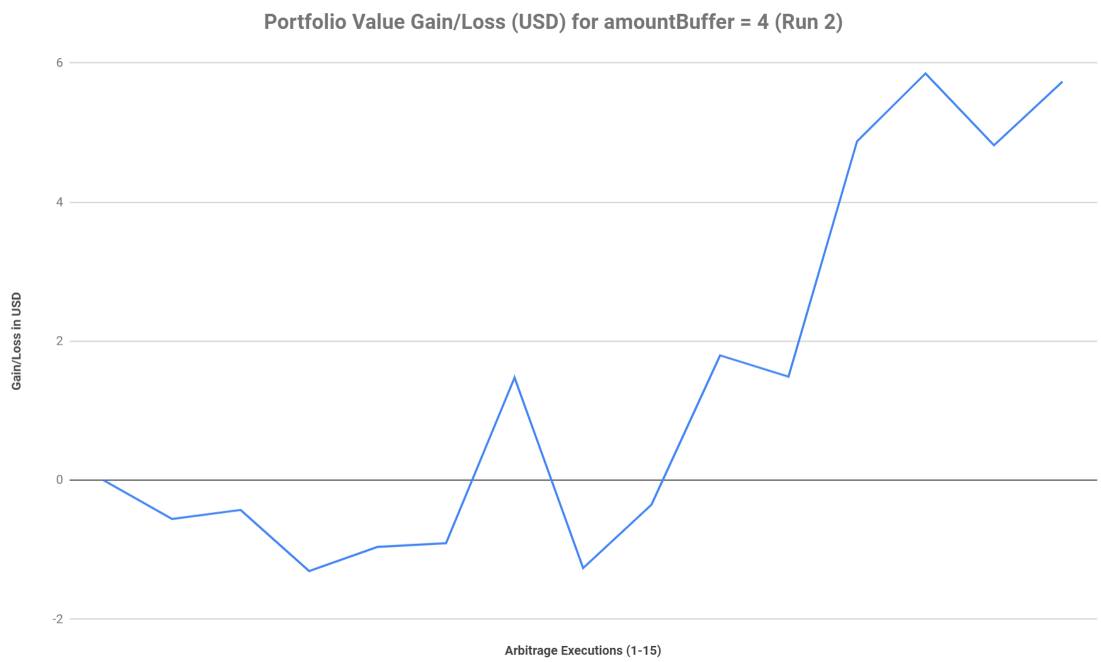

Figure 12: Graph showing arbitrage results with amountBuffer = 4 (Run 2).

Date and Time	Arbitrage Sequence	Portfolio Value (Arbitrage)	Portfolio Value (Held)	Difference	Difference Adjusted to 0
2018/09/11 15:11:03	BTC ICX USDT ETH WTC BTC	2288.641235	2197.041628	91.59960707	0
2018/09/11 15:20:41	BTC BCD ETH USDT QTUM BTC	2291.643678	2201.614678	90.02900036	-1.570606714
2018/09/11 15:23:19	BNB BTC XZC BNB	2296.354733	2207.734957	88.61977623	-2.979830838
2018/09/11 15:23:33	XZC BTC GVT ETH XZC	2296.242815	2207.372034	88.87078064	-2.728826429
2018/09/11 15:28:24	ETH WTC BTC NAS ETH	2298.176096	2204.24053	93.93556637	2.335959295
2018/09/11 15:31:19	SALT BTC ETH SALT	2300.883952	2205.694683	95.18926896	3.589661887
2018/09/11 15:32:07	GVT ETH BTC GVT	2301.547339	2206.933177	94.61416227	3.014555198
2018/09/11 15:33:04	ETH GVT BTC USDT ETH	2302.871706	2207.466253	95.40545226	3.805845185
2018/09/11 15:33:15	GVT BTC ETH GVT	2303.202278	2207.977676	95.22460153	3.624994464
2018/09/11 15:35:31	LTC BTC GVT ETH LTC	2302.507606	2207.32207	95.18553548	3.585928412
2018/09/11 15:35:41	ETH NANO BTC GVT ETH	2302.410621	2207.768001	94.64262005	3.043012982
2018/09/11 15:35:42	GVT ETH USDT BTC GVT	2302.25205	2207.698571	94.55347839	2.953871324
2018/09/11 15:36:13	ETH USDT BTC GVT ETH	2301.953274	2207.58096	94.37231356	2.772706486
2018/09/11 15:38:09	MCO ETH KMD BTC MCO	2301.964353	2207.753371	94.21098113	2.611374064
2018/09/11 15:42:06	NAV BTC ZEN ETH NAV	2305.078096	2212.026288	93.05180824	1.452201164
2018/09/11 15:44:18	BTC HC ETH SKY BTC	2304.525968	2210.988653	93.5373156	1.937708534
2018/09/11 15:45:40	GVT ETH BNB USDT BTC GVT	2305.628185	2211.224176	94.40400917	2.804402101
2018/09/11 15:46:09	BTC GVT ETH NEO BTC	2305.34421	2210.862404	94.48180564	2.882198565
2018/09/11 15:50:47	ETH ETC USDT BTC GVT ETH	2300.468728	2205.431562	95.03716613	3.43755906
2018/09/11 15:59:09	ARK BTC WTC ETH ARK	2297.781967	2200.990297	96.79167003	5.192062963
2018/09/11 16:11:01	ETH BCD BTC GVT ETH	2303.484912	2205.834107	97.65080529	6.051198217
2018/09/11 16:12:33	WTC ETH MTL BTC WTC	2303.511909	2206.279591	97.23231756	5.632710491
2018/09/11 16:13:58	GVT ETH SKY BTC GVT	2305.601714	2207.425142	98.17657163	6.576964561
2018/09/11 16:18:44	ETH HC BTC SKY ETH	2299.979011	2202.49151	97.48750114	5.887894073
2018/09/11 16:40:14	ETH KMD BTC NAS ETH	2292.96216	2198.402539	94.55962099	2.960013922
2018/09/11 16:51:00	ETH SKY BTC NEBL ETH	2295.377317	2197.619503	97.75781491	6.158207842

Figure 13: Table showing Profitability with amountBuffer = 4 (Run 3).

Figure 14: Graph showing arbitrage results with amountBuffer = 4 (Run 3).

With higher values of "amountBuffer", such as ones greater than 6, very few arbitrages are executed. This is due to the fact that Binance is the most highly traded exchange in the world and as such is highly fluid. As such, and due to extraneous circumstances, data for "amountBuffer" = 5 and 6 were unable to be collected and displayed. I did conclude that an "amountBuffer" value of 3 to 4 was ideal, however. With lower values of "amountBuffer", more but less profitable arbitrages are executed. With higher values of "amountBuffer", the profitability of each arbitrage executed increases, but an arbitrageur needs to take into account time. Values of 5 and 6 were not as profitable as 3 or 4, and anything higher than 6 would result in nearly nil arbitrage executions. From my experimental results, I concluded that "amountBuffer" is a critical and necessary parameter. Also, the total trade amount value you choose to execute with in the beginning is another factor that you need to consider. With higher trade values, you will likely end up with more open unfilled orders that will take time to complete. Thus, I suggest anyone trying to build an intra-exchange arbitrage bot to use as minimal a trade value as possible.

CHAPTER 5

Discussion and Observations

On an exchange like Binance, or any highly volatile market, one needs to take into consideration many factors. The first obviously is how you can measure the success of your program. I solved this issue by taking an account snapshot of a past account balance, and comparing it with the estimated portfolio value of my actual account balance after executing a series of arbitrages.

Another thing to take into consideration is that precision matters when it comes down to arbitrage. If we could have just 7-9 significant digits for both amount and pricing, I'm positive "amountBuffer" wouldn't be needed, and if it was needed, a value of 1 would work just fine.

5.1 Important Topics

- Don't use market orders. When you use a market order, it fills at the best possible price available. Because we are executing arbitrage based on best available exchange pricing, executing a market order leaves you susceptible to filling your market order by filling multiple orders in the book with higher prices in a buy-side order, and lower prices in a sell-side order, in effect nullifying the validity of the arbitrage.
- There is a specific precision (number of significant digits) required for the amounts on an order for each trading pair. Many symbols have a precision requirement of 0, which means that the amount must be an integer. These symbols were filtered out when finding arbitrage opportunities to execute because the integer precision would likely result in inaccurate executions of arbitrage.
- There is also a specific precision required for the price, but because we are executing limit orders, there is no filtering or buffering required for the price of an order.

- In this project's implementation, an initial investment of cryptocurrency equivalent to $3000 USD was made, with an additional injection of $2000 USD later on. This amount was divided by the number of coins on Binance, and a specific amount of each coin was bought. The reason for diversification is because we are executing limit orders, we want the arbitrage to go all the way through at the time of execution. If we were executing market orders, this would not be necessary. (But don't use market orders.)
- It is also interesting to note that all limit orders will usually fill at some point in the future. This was found to be the case in this project. All limit orders filled within a timespan of 2 months. Due to this inherent property of working with crypto-exchanges, or any stock market in particular, an unfilled limit order will have the side effect of giving you an inaccurate snapshot of your arbitrage. Ideally, you would want all the trades to complete after executing the order. Hence, that is why you saw spikes and jumps in my graphs. These are most likely due to past limit orders filling in the future and increasing my portfolio value through the delayed arbitrage profit.
- Due to inherent volatility of cryptocurrency market, you should take an account snapshot of your starting coin balances, and store it into a hashmap with each coin as the key and balance as the value. You can then use this hashmap to calculate the account value if you had not executed arbitrage. This same method was used to calculate present value of coin balances. Because Binance's API returns coins that are not listed on the exchange with the "getBalances" method provided by the wallet class in the XChange library, I only polled the server for the coin if it is contained in the list of vertices in my Main class.

CHAPTER 6

Conclusion and Future Work

In this project, an intra-exchange arbitrage bot for the Binance cryptocurrency exchange was created and successfully deployed. During the implementation of the project, two main challenges were encountered, one due to the precision offered by the REST API offered, and the other due to the volatility of the cryptocurrency market. To try to combat the limited precision imposed on amount and price when executing trades, a parameter called ‘‘amountBuffer’’ was introduced to give the arbitrage more breathing room. To combat market volatility, account snapshots were taken of previous account balances and used to compare to current portfolio value; both these values were calculated at the current market exchange rates during the time of execution. Overall, it can be said that it is possible to create an intra-exchange arbitrage bot for the cryptocurrency market, though I believe a significantly larger investment of capital would be required for a sit back and watch your cryptocurrency portfolio grow type of situation.

It is interesting to note that for future work, websockets provided by an exchange could be an area that could be explored further. My exploration of websocket usage use in this project was very limited.

Another exciting direction this project could go is undergoing modification for an exchange like Hitbtc, which actually pays out a commission of 0.01% when you place an order that adds liquidity to the market. A market maker is an entity, such as a person or a bot, that does not buy and sell at the best price offered. In other words, the entity will sell at a price higher than the current best market "ask" price, and buy at a price lower then the best "bid" price. Because we have this data when we poll any exchange's API, we can easily adjust our program to perhaps adjust our price to 99.5-99.9% or 100.1-100.5% of the best market price available, depending on if you were buying or selling, respectively, and get paid commission for doing so. From my experience with Binance, all my limit orders eventually filled within 1-2 months. Hitbtc is a very good choice for executing arbitrage because you then can directly execute the arbitrage and rake in arbitrage profit along with the trading commission received. However, due to the large number of coins on Hitbtc, about 450 at the time of this writing, the capital required to perform a project like this would require a significant amount of capital. I look forward to implementing this for the Hitbtc exchange in the near future.

LIST OF REFERENCES

[1] Wikipedia, ‘‘Cryptocurrency,’’ https://en.wikipedia.org/wiki/Cryptocurrency, 2018, (Accessed on 11/1/2018.

[2] Wikipedia, ‘‘Cryptocurrency exchange,’’ https://en.wikipedia.org/wiki/Cryptocurrency_exchange, 2018, (Accessed on 11/1/2018.

[3] J. Wei, ‘‘Conditions for no triangular arbitrage with transaction costs: A pedagogical note,’’ *Journal of Education for Business*, vol. 73(1), pp. 44--47, 1997.

[4] M. T. T. Ito, K. Yamada and H. Takayasu, ‘‘Free lunch! arbitrage opportunities in the foreign exchange markets.’’ *Cambridge: National Bureau of Economic Research, Inc*, 2012, (Accessed on 4/6/2018).

[5] A. Norry, ‘‘Beginner's guide to bitcoin trading bots,’’ https://blockonomi.com/bitcoin-trading-bots, 2018, (Accessed on 3/15/2018).

[6] butor(Github user), ‘‘Advantage of short-selling and minimum budget,’’ https://github.com/butor/blackbird/issues/100, 2017, (Accessed on 4/11/2018).

[7] E. Demiane and S. Goldwasser, ‘‘Problem 7 set solutions,’’ https://courses.csail.mit.edu/6.046/spring04/handouts/ps7sol.pdf, 2004, (Accessed on 2/30/2018).

[8] Wikipedia, ‘‘Bellman-ford algorithm,’’ https://en.wikipedia.org/wiki/Bellman%E2%80%93Ford_algorithm, 2018, (Accessed on 11/1/2018.

[9] A. Osipovich and E. Jeong, ‘‘Bitcoin's crashing? that won't stop arbitrage traders from raking in millions.’’ http://search.proquest.com.libaccess.sjlibrary.org/docview/1993889011?accountid=10361, 2018, (Accessed on 4/6/2018).

[10] scrawl(username), ‘‘A brief look at cryto arbitrage trading,’’ https://steemit.com/cryptocurrency/@scrawl/a-brief-look-at-crypto-arbitrage-trading, 2016, (Accessed on 2/15/2018).

[11] unknown, ‘‘Our automated crypto trading platform features,’’ https://www.haasonline.com/features/, 2018, (Accessed on 11/17/2018).

[12] unknown, ‘‘Zenbot,’’ https://github.com/DeviaVir/zenbot, 2018, (Accessed on 11/17/2018).

[13] unknown, ‘‘Zenbot,’’ https://gekko.wizb.it/, 2018, (Accessed on 11/17/2018).

[14] unknown, ‘‘Cryptotrader- build your own trading bot in minutes,’’ https://cryptotrader.org/, 2018, (Accessed on 11/18/2018).

[15] unknown, "3commas," https://3commas.io/#tab5/, 2018, (Accessed on 11/18/2018).