267397

BSc Computer Science

Functioning Cryptocurrency Arbitrage Bot

Dr Ben Evans

Statement of Originality

This report is submitted as part requirement for the degree of BSc Computer Science at the University of Sussex. It is the product of my own labour. The report may be freely copied and distributed provided the source is acknowledged. I hereby give permission for a copy of this report to be loaned out to students in future years.

Table of Contents

Chapter 1: Abstract

Building on the inefficiencies observed in the cryptocurrency market compared to traditional financial systems and contextualized by real-world cases such as the Kimchi Premium, this project aims to address these discrepancies through automation. The goal is to design, develop, and implement a functional cryptocurrency arbitrage bot that can identify and exploit price differences between two major exchanges—Binance and OKX. This system utilizes both WebSocket and REST APIs to retrieve real-time data, Analyse cross-exchange price variations, and detect potential arbitrage windows. By integrating live data feeds, accurate symbol mapping, fee considerations, and optimized transfer network selection, the bot can identify viable arbitrage opportunities with a realistic assessment of profitability. The architecture supports modular expansion and maintains up-to-date symbol and fee data to ensure sustained accuracy. Ultimately, this project demonstrates that automated arbitrage can serve as a practical approach to profit generation and potentially contribute to the price efficiency of cryptocurrency markets.

Chapter 2: Introduction

Over recent years, the popularity of cryptocurrencies has significantly increased, attracting attention from both retail and institutional investors. This growing interest is reflected in the cryptocurrency market's substantial rise in total market value, surpassing $2 trillion by late 2024 (Chowdhury, 2024). To put this into perspective, the traditional commodity market for silver recorded a market value of approximately $1.4 trillion during the same period (Silver Institute, 2024). This substantial growth underscores the expanding influence of digital currencies in the global financial landscape. The surge activity is attributed to several factors that differentiate cryptocurrencies from traditional financial instruments.

2.1 Overview of Cryptocurrencies Market

Firstly, cryptocurrencies are built upon blockchain technology, which uniquely combines transparency and user privacy. Blockchain's decentralized structure means that all transactions can be publicly verified while still allowing users a measure of anonymity (Nakamoto, 2008). This significantly contrasts with traditional financial systems, where banks, governments, and regulatory bodies play central roles in validating and overseeing transactions. Blockchain provides users with direct, sovereign control over their assets, eliminating dependency on centralized entities such as banks or governmental authorities (Catalini & Gans, 2016).

Secondly, cryptocurrencies offer extraordinary global accessibility. Investors worldwide can participate in crypto markets regardless of geographic location or traditional banking availability. All that's needed is a stable internet connection and a digital wallet. This capability has democratized access to financial markets, particularly benefiting regions that lack adequate conventional financial infrastructure.

Lastly, unlike traditional markets bound by specific trading hours and holiday closures, cryptocurrency markets operate continuously, 24 hours a day, seven days a week. This constant availability allows traders flexibility and responsiveness, although it also contributes to higher volatility and dynamic market conditions (Gandal & Halaburda, 2016).

Yet, despite these clear advantages, the cryptocurrency market remains notably inefficient compared to traditional financial markets. Each exchange exhibits varying degrees of trading volume, liquidity, and fees. Such fragmentation creates frequent and noticeable

price differences for identical cryptocurrencies across platforms (Makarov & Schoar, 2020). For example, Bitcoin could consistently be priced higher on OKX Exchange than on Binance Exchange due to disparities in local demand, transaction fees, or liquidity constraints. These persistent price differences not only reflect underlying market inefficiencies but also present potential opportunities for systematic exploitation.

2.2 Arbitrage Definitions

Arbitrage involves buying and selling the same asset simultaneously across different exchanges to profit from price discrepancies. Essentially, arbitrage exploits temporary differences in asset prices to generate profit without exposure to price volatility (Makarov & Schoar, 2020). In traditional financial markets, arbitrage opportunities rarely last long, given their highly efficient trading systems, quick dissemination of information, and closely integrated market structures. Cryptocurrency markets, however, frequently offer lasting arbitrage opportunities because of their decentralized nature, diverse regulatory environments, and fragmented marketplace (Xiong & Luo, 2024).

The presence of arbitrage in cryptocurrency markets highlights ongoing inefficiencies and indicates that these markets are still developing compared to traditional financial systems. Importantly, arbitrage activities help improve market efficiency by reducing price disparities, balancing market prices, and enhancing liquidity across platforms (Xiong & Luo, 2024).

2.3 Motivation

One fascinating example illustrating the unique characteristics of cryptocurrency markets is the "Kimchi Premium." This term describes a phenomenon where cryptocurrencies, notably Bitcoin, trade at considerably higher prices on South Korean exchanges compared to international platforms. Such premiums typically arise due to intense local demand, strict capital controls, and regulatory restrictions within South Korea (Park et al., 2019). The Kimchi Premium notably peaked during major cryptocurrency rallies, such as those in late 2017 and peaked at 20.8% on May 19, 2021, highlighting persistent arbitrage opportunities and underlying inefficiencies in crypto trading (Petukhina et al., 2021).

Observing intriguing phenomena like the Kimchi Premium significantly enhanced my interest in cryptocurrency arbitrage and directly aligned with my academic pursuits in computer science. It presented a unique opportunity to apply theoretical concepts learned throughout my studies practically and innovatively. Specifically, the idea of systematically detecting and analysing market inefficiencies through custom-developed logic, algorithms, and independently sourced real-time data appealed strongly to my academic curiosity and personal goals.

2.4 Aim and Objectives

Therefore, the Aim of this project is to design, develop, and evaluate a functional cryptocurrency arbitrage bot capable of autonomously identifying and exploiting real-time price discrepancies across multiple cryptocurrency exchanges. The bot should operate continuously, adapt to market volatility, and provide clear reporting and alerts to support decision-making and transparency.

To achieve this aim, the project will focus on the following key objectives:

1. **Real-Time Price Monitoring**
   Develop a system that continuously retrieves and monitors live cryptocurrency price data across multiple exchanges using both REST and WebSocket APIs. The bot must compare overlapping trading pairs and identify potential arbitrage opportunities based on current market data.

2. **Arbitrage Algorithm**
   Implement a custom-built algorithm that will calculate price discrepancies between exchanges while considering trading fees, withdrawal costs, and exchange-specific conditions.

3. **Simulated Trade Execution**
   Although the initial objective was to automate live trade execution, this functionality was limited to simulated trades due to time and technical constraints. Transactions are logged and evaluated in a simulation environment to verify the logic and profitability of arbitrage opportunities without using real funds.

4. **Email Notification System**
   Integrate an automated email alert system after each cycle, the bot sends an email detailing the most profitable arbitrage opportunity detected, including the trading pair, exchange names, network used, price spread, and estimated net profit.

2.5 Professional Considerations

The development of a cryptocurrency arbitrage bot involves a range of professional responsibilities, especially given the financial nature of the technology and the reliance on public APIs and real-time market data. This section outlines how the project complies with professional standards, including data handling, legal awareness, system integrity, and adherence to the British Computer Society (BCS) Code of Conduct.

1. Public Interest and User Safety

The arbitrage bot was designed and tested in a simulation-only environment, with no real trades executed. This approach ensures that no user funds are at risk and that all operations remain ethically neutral. By simulating transactions and logging potential opportunities, the bot minimizes exposure to financial harm or irresponsible trading behaviour.

2. Privacy and Data Handling

The system does not collect, store, or transmit any personal user data. All market information is retrieved via public APIs and stored in-memory, ensuring that the project maintains user anonymity and complies with data privacy best practices. No authentication keys or personal credentials are embedded within the system's codebase.

3. Legal and Regulatory Compliance

Given the regulatory constraints surrounding cryptocurrency in the UK, this project restricts itself to simulation-based logic and avoids real-time asset transfer or financial execution. The exchanges selected—Binance and OKX—were chosen in part due to their API accessibility within UK compliance boundaries. Future iterations involving live trading would require deeper legal analysis to ensure full regulatory compliance.

4. Professional Integrity and Accuracy

The bot prioritizes accuracy and transparency by simulating full trade cycles, including fees, slippage, and network selection. All arbitrage signals are logged in structured formats and sent to users through secure, real-time email notifications. This ensures a clear audit trail and promotes responsible use of the tool.

5. Inclusivity and Accessibility

While the system was designed for technical users, care was taken to ensure that logging, result interpretation, and system configuration were readable and modifiable. With modular architecture and human-readable logs, the tool encourages accessibility for academic researchers, developers, and hobbyists alike.

6. Alignment with BCS Code of Conduct

This project upholds the core values of the BCS Code of Conduct, including:

- Acting in the public interest by avoiding misuse of financial technology.
- Operating within professional competence through simulation-only testing.
- Ensuring transparency, fairness, and respect in design decisions and documentation.
- Promoting best practices in secure data handling and risk avoidance.

2.6 Project Outline

The following chapters of this report present a structured progression of the project. Chapter 3 introduces the technical foundations of cryptocurrency trading and arbitrage, providing the necessary background on exchanges, APIs, and arbitrage strategies. Chapter 4 reviews existing literature and compares prior arbitrage bot implementations to this project. Chapter 5 outlines the system architecture, design logic, and strategy planning, followed by Chapter 6, which details the actual implementation of the bot. The results of the bot's performance are analysed in Chapter 7, leading into Chapter 8, which discusses the limitations and challenges encountered. The final chapter, Chapter 9, concludes the report with key findings and suggestions for future work.

Chapter 3: Technical Background

This chapter provides an overview of the technical concepts, tools, and systems relevant to cryptocurrency arbitrage and automated trading environments. It is intended to help readers understand the technical landscape in which the project is situated. Rather than focusing on the specific design or implementation of this project, the chapter explains foundational topics such as exchanges, APIs, networks, trading pairs, and types of arbitrage strategies, providing the necessary context for understanding the discussions in later chapters.

3.1 Cryptocurrency Market in Practice

While foundational concepts such as blockchain and decentralization have been introduced earlier, this section focuses on the technical aspects of cryptocurrency markets relevant to this project.

3.1.1 Exchange

A cryptocurrency exchange is a digital platform that facilitates the trading of digital assets, allowing users to buy, sell, or convert cryptocurrencies. These exchanges serve as crucial infrastructure within the broader crypto market, enabling price discovery, liquidity, and access to a wide range of crypto assets. Functionally, they operate similarly to traditional financial exchanges, but often with fewer regulatory requirements and greater accessibility.

Cryptocurrency exchanges can be broadly categorized into two types: centralized exchanges (CEXs) and decentralized exchanges (DEXs). Centralized exchanges are managed by a third-party organization that provides custodial services, order-matching systems, and user interfaces. Examples of well-established CEXs include Binance, OKX, and Coinbase. These platforms typically offer high liquidity, fast trade execution, and user-friendly features, making them the preferred choice for most retail and institutional traders. However, CEXs require users to deposit their funds into exchange-managed wallets, which introduces a level of custodial risk.

On the other hand, decentralized exchanges function without a central authority and allow users to trade directly from their personal wallets through smart contracts. DEXs such as Uniswap and PancakeSwap emphasize transparency, user sovereignty, and censorship resistance. Although they eliminate the need for trust in a centralized intermediary, DEXs

often suffer from lower liquidity, limited trading pairs, and slower execution times, particularly during periods of network congestion.

In the context of arbitrage, both CEXs and DEXs can present opportunities due to pricing inefficiencies across platforms. However, centralized exchanges are generally more suitable for high-frequency and latency-sensitive strategies due to their mature API infrastructure and order book systems, which are further examined in later sections of this chapter.

3.1.2 Trading Volume and Market Capitalization

Liquidity and trading volume are critical factors that influence the success and reliability of arbitrage strategies. Trading volume refers to the total value of assets exchanged within a given timeframe, typically measured per trading pair or per exchange. Liquidity, on the other hand, describes how easily assets can be bought or sold without significantly affecting their price. High liquidity generally corresponds with deeper order books and tighter bid-ask spreads, resulting in more stable and efficient trading environments.

For arbitrageurs, these factors are not just technical preferences, they are operational necessities. Executing arbitrage trades requires entering and exiting positions quickly and with minimal slippage. On exchanges or pairs with low liquidity, even modest trade sizes can shift prices unfavorably, reducing or completely eliminating potential profits.

3.1.3 Maker-Taker Fee

Most centralized cryptocurrency exchanges operate on a maker-taker fee model, which differentiates transaction fees based on how a trade interacts with the order book. In this model, a maker is a trader who places a limit order that adds liquidity to the market by not immediately matching with an existing order. A taker, by contrast, is a trader whose order executes instantly against an existing order, thereby removing liquidity.

Fee structures typically reward makers with lower fees or even rebates, while takers are charged higher fees for the immediacy of their execution. These fee distinctions are crucial in the context of arbitrage, where net profit margins can be extremely tight. Even small fee differences significantly affect the final profitability of an arbitrage trade. If both entry and exit trades are executed as taker orders, the cumulative fees can quickly erode any gains from price discrepancies.

3.1.4 Order Book Mechanics and Depth

An order book is a real-time ledger maintained by exchanges that lists all outstanding buy and sell orders for a trading pair. It is composed of bid orders (buy offers) and ask orders, sorted by price. The top of the book, the highest bid and the lowest ask, represents the best available prices for immediate execution. These prices, however, are often associated with limited volume, especially on less liquid pairs.

The concept of market depth refers to how much volume is available at each price level. A "deep" order book contains a wide range of bids and asks with significant volume, which helps stabilize price movements during large trades. A shallow book, by contrast, may show attractive top-of-book prices but offer minimal volume behind them. In such cases, executing a trade larger than the available liquidity at the best price causes the trade to cascade through the book—this is known as slippage.

3.1.5 Trading Pairs

A trading pair represents a market where one cryptocurrency is exchanged for another, typically in the format *Base Currency / Quote Currency*. For example, in BTC/USDT, BTC is the asset being traded, and USDT is used to measure its value. This structure is consistent across most exchanges and is essential for identifying arbitrage opportunities.

Stablecoins like USDT, USDC, and BUSD are commonly used as quote currencies due to their stable value, making them ideal for cross-exchange comparisons. In some cases, exchanges may display trading pairs using different naming conventions (e.g., BTC-USDT or BTC_USDT), which must be standardized in the bot to ensure accurate pair matching. Correct alignment of trading pairs is crucial to avoid misidentifying opportunities and ensuring accurate calculations.

3.1.6 Network and Transaction Fee

Every cryptocurrency asset is tied to a specific blockchain network that governs how it is transferred. Each network, such as Ethereum, Tron, or Binance Smart Chain, has its own

fee structure, confirmation time, and congestion level, all of which affect transaction cost and speed.

Many assets, like USDT or USDC, exist across multiple networks (e.g., ERC-20, TRC-20, BEP-20), but these versions are not interchangeable. This means both the asset and its network must be supported by the exchanges involved in a trade.

In arbitrage strategies, especially spatial arbitrage where assets are transferred between platforms, the choice of network can significantly impact profitability. Higher fees or slower confirmation times can erode gains, while selecting a faster, cheaper network improves execution efficiency. For this reason, network type and fee estimates are considered when evaluating and simulating arbitrage opportunities.

3.2 API Formats

Application Programming Interfaces (APIs) are essential tools in modern cryptocurrency trading. They enable software programs, such as trading bots, to interact with exchange platforms by retrieving market data, placing orders, and monitoring account balances in real time. APIs serve as the communication layer between an arbitrage bot and various cryptocurrency exchanges, allowing the bot to function autonomously and efficiently across multiple platforms.

3.2.1 REST

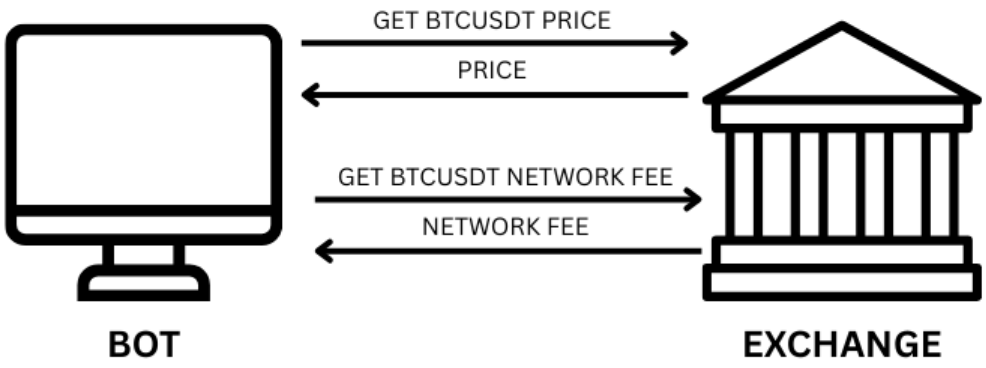


Figure 3.1 REST API Communication Flow Between Trading Bot and Exchange

Representational State Transfer (REST) APIs are one of the most common and widely adopted formats used in cryptocurrency exchange integrations. A REST API operates over standard HTTP protocols, using requests such as GET, POST, and DELETE to communicate between a client (e.g., a trading bot) and a server (e.g., a cryptocurrency exchange).

REST APIs are stateless, meaning each request from the client to the server is processed independently, without storing any context from previous interactions. While this makes REST APIs reliable and easy to implement, it can also introduce latency when compared to real-time data streams. In a case where an arbitrage bot polling for updated prices via REST might miss short-lived opportunities due to delays between requests.

3.2.2 WebSocket

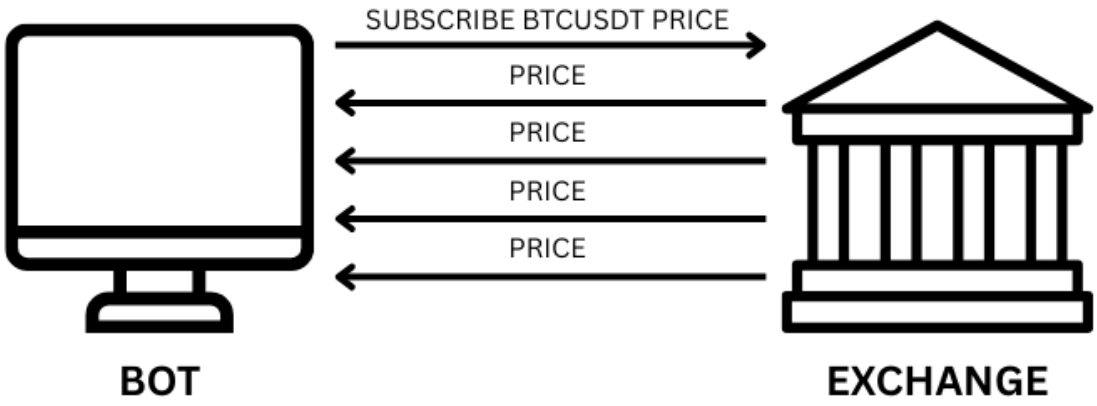


Figure 3.2 WebSocket API Communication Flow Between Trading Bot and Exchange

WebSocket is a communication protocol that provides full-duplex, real-time data transmission between a client and a server over a single, persistent connection. Unlike REST APIs, which rely on a request-response model, WebSocket enables the server to push data to the client immediately when new information is available. This characteristic makes WebSocket particularly well-suited for applications requiring low-latency and continuous data updates, such as real-time trading and arbitrage.

In the context of cryptocurrency exchanges, WebSocket APIs are commonly used to stream live market data, including price updates, order book changes, and trade executions. Once the connection is established, the client subscribes to specific data channels, such as ticker information for a trading pair, and the server sends updates as they occur. This eliminates the need for repeated polling, reducing both latency and bandwidth usage.

3.3 Arbitrage Strategies

Arbitrage can be approached through various strategies, each designed to exploit different kinds of market inefficiencies. From inter-exchange arbitrage to statistical and decentralized strategies, the field continues to evolve with market complexity and technological advancement. However, this report will focus only on the types of arbitrages most relevant to the project scope.

3.3.1 Spatial Arbitrage

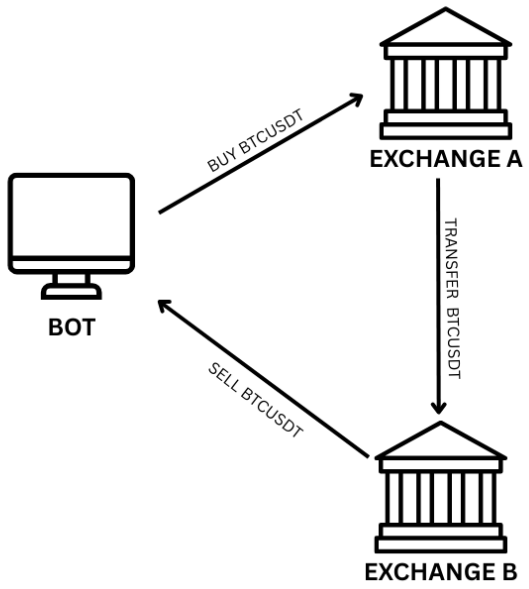


Figure 3.3 Workflow of Spatial Arbitrage Between Two Exchanges

Spatial arbitrage, commonly known as inter-exchange arbitrage, involves buying an asset at a lower price from one exchange and immediately selling it at a higher price on another exchange. The primary objective is to profit from the existing price discrepancy. This strategy typically requires transferring the asset between exchanges to execute the sale, thus involving some level of transfer time.

When implementing traditional arbitrage, several key considerations must be accounted for. These include transaction fees such as trading fees, withdrawal fees, and network fees, all of which can significantly impact profitability. Additionally, adequate liquidity on both exchanges is essential to avoid price slippage and ensure successful trade execution.

3.3.2 Market-Neutral Arbitrage

Market-neutral arbitrage, often referred to as long/short cross-exchange arbitrage, is a strategy designed to exploit price discrepancies between two exchanges without transferring the underlying asset. Unlike spatial arbitrage, which involves physically moving assets between platforms and incurring network fees and delays, market-neutral arbitrage relies on executing simultaneous offsetting positions across exchanges. This method aims to profit from the price spread while minimizing exposure to the broader market's direction.

In a typical setup, the arbitrageur buys an asset (long position) on the exchange where it is undervalued and simultaneously sells the same asset (short position) on another exchange where it is overvalued. The strategy assumes that the spread between the two prices will converge over time. When the gap narrows, both positions are closed, securing a profit from the differential. This structure ensures that the trader is not directly affected by whether the market as a whole goes up or down, hence the term market-neutral.

One of the key advantages of this approach is that it eliminates the need for blockchain transfers, avoiding associated costs and delays. However, it comes with its own requirements, such as the ability to short-sell assets, which is typically only available on certain margin-enabled exchanges. It also assumes sufficient liquidity and leverage access on both platforms to support simultaneous execution.

Chapter 4: Literature Review and Related Works

This chapter explores the existing body of literature and practical tools relevant to cryptocurrency arbitrage. It begins by examining how arbitrage functions in the context of traditional and digital financial markets, with particular focus on the inefficiencies unique to the cryptocurrency ecosystem. The discussion includes foundational research on market inefficiency, price dispersion across exchanges, and the role of regulatory fragmentation in enabling arbitrage opportunities. In addition to academic perspectives, the chapter also reviews notable existing arbitrage bots, such as the Blackbird system, highlighting their strategies, limitations, and how they compare to the approach implemented in this project. By understanding both theoretical foundations and practical precedents, this review establishes the relevance and innovation of the bot developed herein.

4.1 Arbitrage in Cryptocurrency Markets

Arbitrage has been a core idea in finance for many years. It is often seen as a way for markets to correct themselves, helping prices stay accurate. However, Lynn A. Stout (2002), in her paper "The Mechanisms of Market Inefficiency," argued that real-world markets are not perfect. She pointed out that investor behaviour, transaction costs, and uneven access to information make it difficult for arbitrage to always work effectively. Her work challenged the traditional belief that markets are always efficient, and that arbitrage will quickly remove any price differences.

More recent studies have shown that it has become even harder to take advantage of arbitrage opportunities in traditional financial markets. As trading technology has improved and global markets have become more connected, any price gaps are often closed almost instantly. Hassani, Huang, and Silva (2019), in a paper published in the journal *Economies*, note that while arbitrage opportunities still exist, they are much harder to profit from due to increased competition and tighter market conditions.Therefore, many traders and researchers have started looking at cryptocurrency markets instead, where inefficiencies are still common.

These studies show that while arbitrage is not a new idea, its usefulness in traditional markets has become limited. This has increased interest in cryptocurrencies, where the lack of regulation, different trading rules, and many separate exchanges often lead to price gaps that can be exploited.

In the context of cryptocurrency, however, the picture is different. Makarov and Schoar (2020) provided one of the earliest in-depth analyses of arbitrage in cryptocurrency markets. They showed that, unlike traditional markets, crypto exchanges are fragmented and poorly integrated. As a result, significant price differences for the same asset often persist across platforms. These inefficiencies are especially noticeable across regions, where capital controls and regulatory differences create barriers to arbitrage.

Building on this, Foley, Karlsen, and Putniņš (2022) explored how exchange characteristics such as fees, verification processes, and regional demand contribute to persistent price differences across crypto markets. Their findings reinforced Makarov and Schoar's conclusions, showing that despite the growing maturity of digital asset markets, structural and regulatory fragmentation continues to present challenges to price convergence.

More recently, Jalan and Matkovskyy (2023) examined how the cryptocurrency market has changed since major events such as the FTX collapse. Their research found that the crypto ecosystem is becoming more regulated and more connected. Because of this, prices across different exchanges are starting to match more closely. This makes it harder for traders to find price gaps that they can use for arbitrage. They also noted that bigger financial institutions and better trading systems are making the market more efficient overall.

These changes suggest that it is getting much more difficult to find easy or obvious arbitrage opportunities in today's crypto market. Unlike in earlier years, traders now need faster systems and smarter strategies to find any small remaining chances for profit. This growing difficulty is exactly what motivates this project: to explore whether profitable arbitrage opportunities still exist in the current, more efficient crypto trading environment.

4.2 Review of Existing Arbitrage Bots

While academic studies provide valuable insight into the mechanics and challenges of arbitrage in cryptocurrency markets, practical implementations offer a clearer picture of how such strategies are executed in real-world conditions. Among the various open-source tools available, one of the most referenced is the Blackbird Arbitrage Bot, which has served as both a learning resource and a functional example of cross-exchange arbitrage automation.

Blackbird Arbitrage Bot, developed by Eric Teti in 2017, was designed to exploit price differences between exchanges without transferring assets across networks. Instead, it follows a market-neutral arbitrage strategy by holding balances on two exchanges and executing simultaneous buy and sell orders when a profitable spread is detected. This method avoids blockchain transfer fees and delays, relying entirely on dual-exchange positioning.

Technically, Blackbird was implemented in C++ and focused solely on the BTC/USD trading pair. It retrieves data through REST API polling, which—while simple and effective at the time—introduces latency that may limit its effectiveness in today's faster-moving markets. The bot also lacks support for WebSocket-based real-time data, which has become essential for low-latency trading. Additionally, Blackbird has not seen active development in recent years, and its static design offers little flexibility in adapting to modern API standards or expanding to other trading pairs or strategies.

Despite these limitations, Blackbird laid important groundwork and remains a useful reference for understanding arbitrage mechanics. This project draws inspiration from Blackbird's foundational structure but takes a different approach. The goal is to create a more modular, adaptable bot using Python, with support for multiple trading pairs and a combination of REST and WebSocket APIs for improved responsiveness. Unlike Blackbird's purely market-neutral design, this project also explores spatial arbitrage strategies that involve transferring assets between exchanges.

While Blackbird introduced many of the key components of an arbitrage system, the evolving crypto landscape demands faster, more flexible tools. The full implementation, including system architecture, data flow, and trading logic, will be outlined in detail in the following chapters

Chapter 5: Approach, Design and Strategies

This chapter outlines the systematic approach taken to design and implement the cryptocurrency arbitrage bot. It begins by detailing the overall architecture of the system, which is structured around modular and asynchronous data pipelines for real-time performance. Following this, the chapter explains the rationale behind key design decisions, such as the use of in-memory data handling and concurrent task execution. Emphasis is placed on the data retrieval and processing workflows, including how trading pairs are selected, how arbitrage opportunities are simulated using real market data, and how profitability is evaluated under realistic constraints. The strategies adopted—particularly spatial arbitrage—are discussed in relation to both practical viability and system extensibility. Together, these components form the technical backbone of the bot and enable it to operate effectively in a live, data-intensive environment.

5.1 Bot Architecture

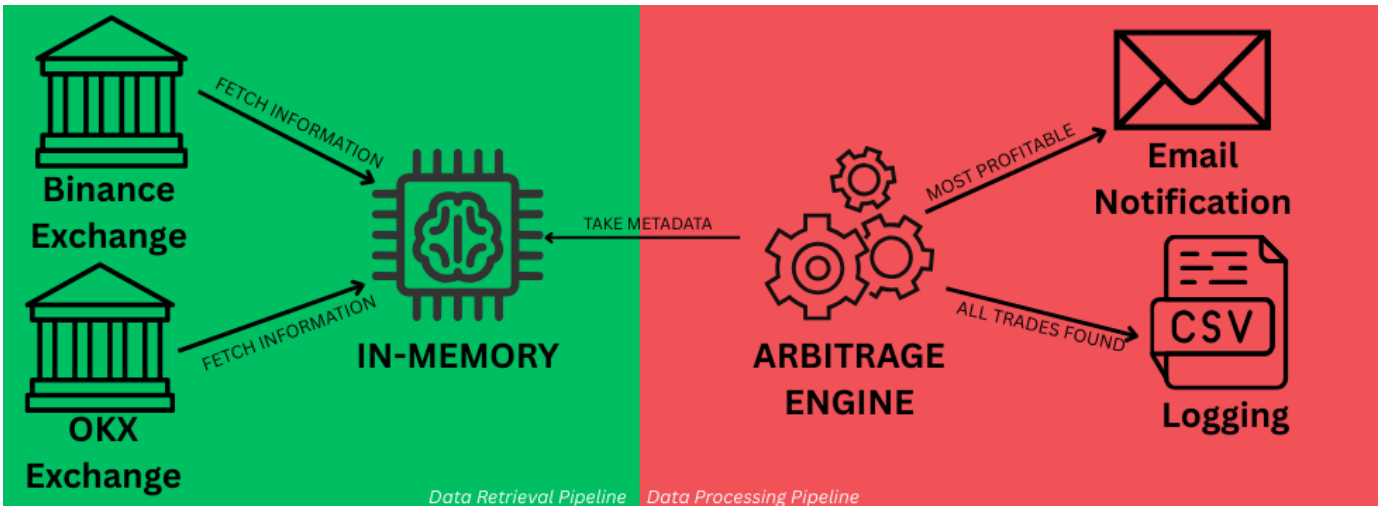


Figure 5.1 Architecture of the Cryptocurrency Arbitrage Bot

The architecture of the cryptocurrency arbitrage bot is structured around two primary functional pipelines: data retrieval and data processing. These pipelines operate concurrently and are interdependent, working together to maintain an up-to-date understanding of market conditions and to evaluate potential arbitrage opportunities in real-time.

The data retrieval pipeline is responsible for collecting live market data (including order book and ticker information) and network-related fee data from external exchanges. This information is streamed and updated asynchronously to ensure that the system operates with the latest available figures.

In parallel, the data processing pipeline consumes this retrieved data, simulates arbitrage trades across exchanges, and evaluates their profitability. When a viable opportunity is identified, it is logged for analysis and an immediate email alert is dispatched to the user.

At the core of the system is a shared, in-memory data structure that acts as a live repository for all incoming and processed data. By decoupling the retrieval and processing responsibilities—and allowing both to run asynchronously—the system is able to maintainresponsiveness and accuracy even under rapid market fluctuations. This modular design enhances both scalability and reliability, making it easier to extend and maintain over time.

5.2 Data Retrieval Pipeline

The data retrieval pipeline is responsible for fetching real-time market data and fee information from external exchanges. These components operate asynchronously to minimize latency and maximize responsiveness.

While this chapter introduces the architecture and modular roles of both pipelines, the detailed workflow of the data retrieval pipeline is intentionally deferred to Chapter 6. This separation is made to keep the design discussion focused and avoid overwhelming readers with low-level implementation details. The retrieval process, though essential, is largely technical and better explored in context with code-level explanations

5.2.1 Exchange selection

The selection of exchanges plays a critical role in the functionality and reliability of an arbitrage system. For this project, Binance and OKX were chosen as the primary trading platforms due to a combination of practical, technical, and regulatory considerations.

Firstly, both Binance and OKX are highly established and globally recognized exchanges. They consistently rank among the top platforms in terms of user base, trading volume, and market liquidity. High liquidity is especially important for arbitrage systems, as it reduces the risk of slippage and ensures that simulated trades reflect realistic market conditions. Their large volumes of popular trading pairs provide a stable environment for testing and evaluating cross-exchange pricing opportunities.

Secondly, from a technical standpoint, both exchanges offer comprehensive API access, including WebSocket support for real-time data and RESTful endpoints for accessing metadata, network fees, and account-level configurations. Their APIs are well-documented, reliable, and widely supported by developer tools, which significantly eases integration and ensures system robustness during continuous operation.

Lastly, the decision was also influenced by regulatory constraints in the United Kingdom, which limit the availability and features of certain exchanges to UK-based users. Binance and OKX are among the few major platforms that still provide unrestricted public API access while remaining accessible under current UK compliance standards. This made them both practical and compliant choices for the scope of this academic project.

5.2.2 Trading Pair Selection

The set of trading pairs used by the arbitrage bot is not hardcoded or limited to a fixed list of assets. Instead, it is dynamically determined based on the available listings from both Binance and OKX. This approach ensures that the bot is resilient to changes such as the listing or delisting of asset pairs on either exchange. By dynamically querying both platforms' public APIs, the bot identifies only those trading pairs that are present on both exchanges, allowing it to operate adaptively in a live market environment.

For each selected trading pair, the bot retrieves several critical attributes required for accurate simulation and analysis. These include:

- Real-time price ticker
- Order book depth
- Supported withdrawal networks
- Network-specific fees
- Network-specific transfer limits

These attributes are continuously updated to reflect the latest market conditions and are stored in memory to ensure low-latency access for subsequent processing stages.

By adopting a dynamic and data-driven approach to trading pair selection, the system remains highly flexible, automatically incorporating newly listed pairs and excluding unavailable ones without manual intervention. Moreover, the timely retrieval of market and network attributes provides a strong foundation for the next pipeline, data processing

which relies on this information to simulate trades, evaluate profitability, and trigger notifications.

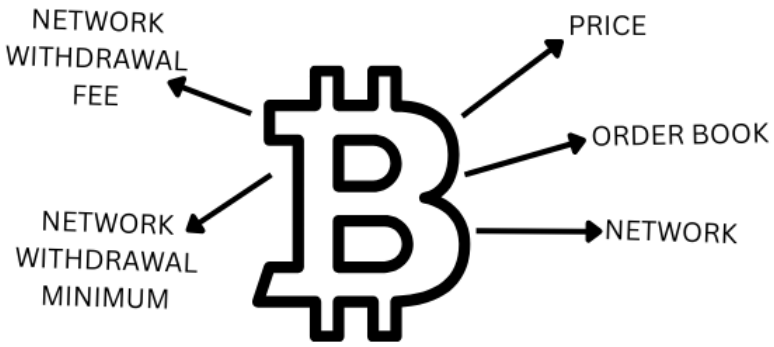


Figure 5.2 Asset Pair Attributes Illustration

5.3 Data Processing pipeline

The data processing pipeline transforms retrieved market data into actionable insights. It performs arbitrage simulation, logging, and user notification.

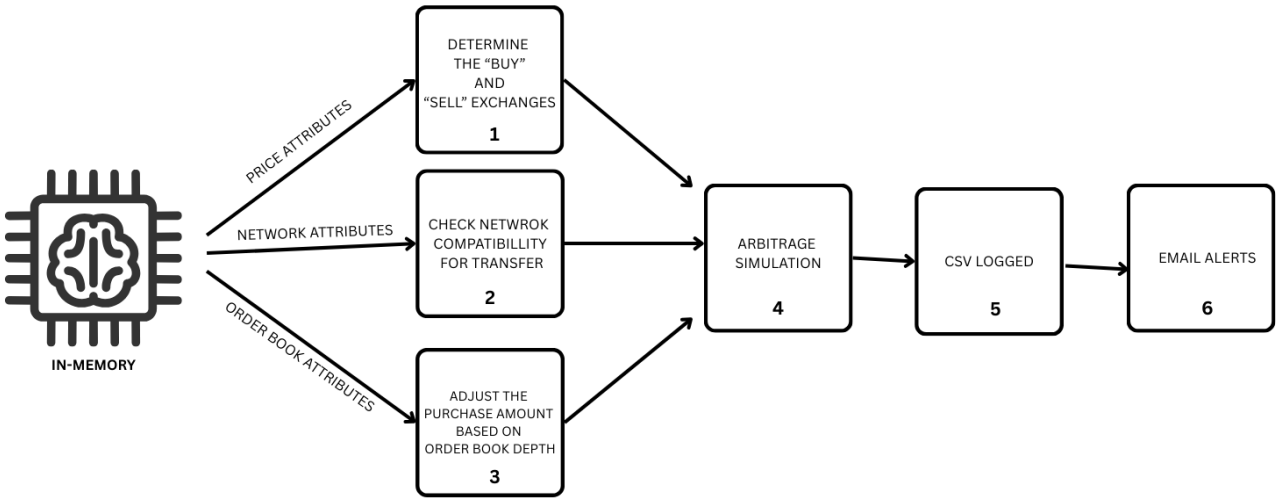


Figure 5.3 Arbitrage Engine Workflow

5.3.1 Arbitrage Strategies

Spatial arbitrage was selected for its practicality, transparency, and suitability for modelling asset transfers (Makarov & Schoar, 2020; Xiong & Luo, 2024), as it reflects real-world conditions where price differences persist due to fragmentation and regional inefficiencies. In this strategy, the bot compares real-time price differences between two centralized exchanges—Binance and OKX—and evaluates whether a profitable opportunity exists after accounting for trading fees, withdrawal costs, and network compatibility.

Spatial arbitrage is particularly suitable for simulation environments because it captures the logistical and economic realities of moving assets across platforms. The approach also aligns with the bot's modular architecture, which supports continuous monitoring, low-latency decision-making, and fee-sensitive trade evaluation across multiple blockchain networks.

5.3.2 Arbitrage Engine Workflow

The arbitrage engine operates entirely on in-memory data, which is continuously updated by both WebSocket and REST components. This allows the bot to run efficiently without having to fetch external data during each cycle. The main focus of the engine is to simulate and evaluate arbitrage opportunities using the most up-to-date order books, price information, network fees, and exchange metadata already stored in memory.

The engine runs on a loop, scanning through all asset pairs that are available on both Binance and OKX. For each asset, it gathers necessary data such as bid/ask prices, trading fees, available blockchain networks, and liquidity depth. Before any simulation can take place, the bot determines the most favorable direction to trade—whether to buy on Binance and sell on OKX, or the other way around—based on live price differences. This direction is important, as it directly affects which side of the order book will be used during the simulation.

Next, the bot checks for a shared network between the two exchanges that can be used to transfer the asset. Since many assets support multiple blockchain networks, it compares the options and selects the one with the lowest total fee. If no common network exists, the asset is skipped. Once direction and network are selected, the bot evaluates the order

book's liquidity to ensure the trade size is realistic. If the current market cannot fulfill the full budget, the bot dynamically adjusts the size to match available volume.

At this point, the bot has everything it needs to simulate a full arbitrage cycle: the buy and sell exchanges, the selected network, an adjusted budget, and the live order book data. This information is then passed into the simulation, where the actual profit potential is evaluated under realistic market conditions.

5.3.3 Spatial Arbitrage Simulation Logic

This project does not execute real trades but instead performs full-cycle simulations to estimate whether an arbitrage opportunity is profitable. The simulation is critical, as it goes beyond simply comparing price differences. It incorporates real-world trading elements such as price impact, slippage, trading fees, and withdrawal costs—factors that significantly influence whether a theoretical arbitrage can actually be executed with profit.

The simulation begins by executing a virtual market buy on the selected exchange. The bot walks through the ask side of the order book using the adjusted budget, calculates how much of the asset would be received, applies the maker fee, and deducts the withdrawal fee based on the selected blockchain network. This step provides a realistic view of how much of the asset would be available for transfer. The transfer itself is simulated by reducing the received quantity according to the network fee—no time delay is applied, but the cost impact is accounted for.

Next, the bot simulates selling the asset on the destination exchange by walking through the bid levels. The logic is similar to the buy phase, except taker fees are applied to the USDT proceeds instead. After completing both buy and sell simulations, the engine calculates the net profit and profit percentage. Because all of this is done using real order book data, the simulation closely reflects what would happen in a live market scenario and allows the bot to assess whether the opportunity is truly profitable.

5.3.4 Logging and Filtering

After completing the full simulation cycle for all eligible asset pairs, the arbitrage engine enters its final stage: filtering and recording results. Each simulated trade is evaluated based on its profit percentage, and only those that exceed a predefined threshold, set by the user, are considered valid opportunities. This filtering mechanism is important to eliminate low-margin or borderline trades that may not be practical after accounting for real-world execution delays or sudden price shifts. By enforcing a minimum profit percentage, the bot focuses only on the most promising and meaningful trades.

Once filtered, any profitable opportunities are saved and logged for tracking purposes. The results are appended to a CSV file, which acts as a historical record of all arbitrage events detected during runtime. In addition to logging, the most profitable opportunity from the current cycle is sent via email alert to the user, allowing for real-time updates without manual monitoring. This combination of structured logging and automated notifications helps make the arbitrage engine both practical and transparent for users looking to evaluate or act on high-value trade signals.

5.4 Key Design Decisions

This section highlights several design choices that distinguish this bot from simpler or less performant implementations.

5.4.1 In-memory Over Database

In the context of high-frequency trading and real-time arbitrage detection, the choice of data storage architecture is pivotal. This project employs an in-memory data structure instead of a traditional database to manage live market data and fee information. This design decision is driven by the need for ultra-low latency and high throughput, which are essential for timely identification and execution of arbitrage opportunities. Research has shown that in-memory processing is highly effective for reducing latency and supporting real-time responsiveness in data-intensive financial systems (Hassani, Huang, & Silva, 2019).

In-memory data structures facilitate concurrent access and updates, which is beneficial for handling multiple data streams from different exchanges simultaneously. This concurrency ensures that the bot can process and react to new information in real-time, maintaining an up-to-date view of the market.

While traditional databases offer advantages in terms of data persistence and complex querying capabilities, they introduce latency that is unacceptable in a real-time trading environment. By prioritizing speed and responsiveness, the in-memory approach aligns with the performance requirements of an effective arbitrage system.

5.4.2 Concurrent Asynchronous Execution

This approach is particularly beneficial for handling the bot's two primary pipelines: data retrieval and data processing. The data retrieval pipeline continuously gathers real-time market data and fee information from multiple exchanges, while the data processing pipeline analyses this information to identify potential arbitrage opportunities. By executing these pipelines concurrently, the bot ensures that the most recent data is always available for analysis, minimizing latency and maximizing the chances of capitalizing on fleeting market inefficiencies.

Chapter 6: Implementation

This chapter presents the practical implementation of the cryptocurrency arbitrage bot, translating the system architecture and strategies defined in the previous chapter into functional code. It details how real-time data is collected from Binance and OKX using both WebSocket and REST APIs, and how this information is stored and processed in an efficient in-memory structure. The chapter then explains how the bot simulates full arbitrage cycles using live market data, accurately modeling fees, order book depth, and liquidity constraints. In addition, it describes how simulation results are logged and communicated via email notifications, enabling both transparency and real-time user engagement. Supporting libraries, technologies, and data structures used throughout the system are also introduced, demonstrating how the implementation aligns with the design principles established earlier in the project.

6.1 Real-Time Data retrieval

The effectiveness of an arbitrage system depends heavily on the freshness and accuracy of market data. To maximize responsiveness and minimize latency, this project avoids third-party aggregators like CCXT and instead integrates directly with the native APIs of Binance and OKX. This approach ensures that all price, order book, and fee data is retrieved and updated in near real time.

6.1.1 WebSocket Integration

The bot establishes separate WebSocket connections to Binance and OKX and subscribes to two primary data streams from each exchange: one for ticker updates and another for the top five levels of the order book. These feeds supply the current price and market depth information required for precise trade simulation and liquidity assessment.

Each exchange offers distinct message formats and subscription methods. Therefore, the implementation includes custom message-handling routines for each platform. Binance provides a combined stream with both ticker and depth updates, while OKX requires separate subscriptions and delivers data in a different structure. All messages are parsed asynchronously and standardized into a unified format. Symbol names are also normalized (e.g., converting "BTC-USDT" to "BTCUSDT") to ensure consistency across the system.

WebSocket technology is especially advantageous in this context because it enables low-latency, real-time data streaming, making it ideal for market applications that require continuous updates (Zheng et al., 2018). The parsed data is stored in a shared in-memory dictionary, which serves as the central access point for the arbitrage simulation engine. This approach minimizes latency by avoiding repeated API requests and ensures that simulations are based on the freshest available market data.

6.1.2 Rest API Integration

While WebSocket provides the bot with high-frequency price and order book data, REST API integration is used to retrieve critical metadata from Binance and OKX—specifically, information that does not require real-time updates. These include the list of supported trading pairs, withdrawal network details, minimum transfer limits, and withdrawal fees. This metadata is essential for assessing whether a valid arbitrage cycle can occur between two exchanges using compatible transfer networks.

REST API calls are executed using HTTP GET requests. Although both exchanges expose similar data, their APIs differ in structure and authentication. Binance's endpoint returns asset configuration data including withdrawal fees and requires HMAC SHA256-based signing of query strings. OKX, on the other hand, requires a more complex signature including a timestamp, HTTP method, and request path, encoded using HMAC and base64, along with a required API passphrase.

Due to these differences, the system implements exchange-specific logic to retrieve and authenticate each request. The data is then parsed, normalized, and stored in a central in-memory dictionary shared across the system. This repository includes standardized network names, fee amounts, and trading symbols to ensure compatibility in further processing.

6.1.3 In-Memory Data Handling

To support real-time responsiveness and reduce processing latency, the bot uses an in-memory data structure rather than a traditional database for storing market and fee data. This design enables the system to rapidly access, update, and share information across all core components—including data retrieval, simulation, logging, and notification.

The primary structure is a global Python dictionary referred to as `symbol_prices`. Each key represents a normalized trading pair symbol (e.g., "BTCUSDT"), and each value is a nested dictionary containing exchange-specific data under keys like "Binance" and "OKX." These nested dictionaries include fields such as:

- **price**: The latest market price
- **bid** and **ask**: Lists representing top levels of the order book
- **network**: Information about supported networks, fees, and withdrawal limits

```
symbol_prices = {
    "BTCUSDT": {
        "Binance": {"price": -1, "bid": [], "ask": [], "network": {}},
        "OKX":     {"price": -1, "bid": [], "ask": [], "network": {}}
    },
    ...  # Other symbols available on both exchanges
}
```

Figure 6.1 In-memory visualization

This shared data structure is updated asynchronously using Python's `asyncio` event loop, allowing multiple coroutines to operate concurrently without explicit locking mechanisms. For instance:

- WebSocket handlers update market data in real time.
- Scheduled REST calls refresh withdrawal fees and network info.
- The arbitrage engine reads from this structure to simulate trades.
- The logging and notification subsystems access it for output.

The in-memory model ensures minimal latency in accessing the freshest available data, a key requirement for exploiting short-lived arbitrage opportunities. To mitigate the volatility of in-memory storage (e.g., data loss on crash), the bot periodically snapshots the current state of `symbol_prices` and writes it to a JSON file. These backups provide valuable debugging and recovery support without compromising runtime speed.

Ultimately, the in-memory dictionary functions as the central data hub of the system—bridging the retrieval and processing pipelines and enabling the bot to respond to market shifts with speed and precision.

6.2 Data Retrieval Workflow

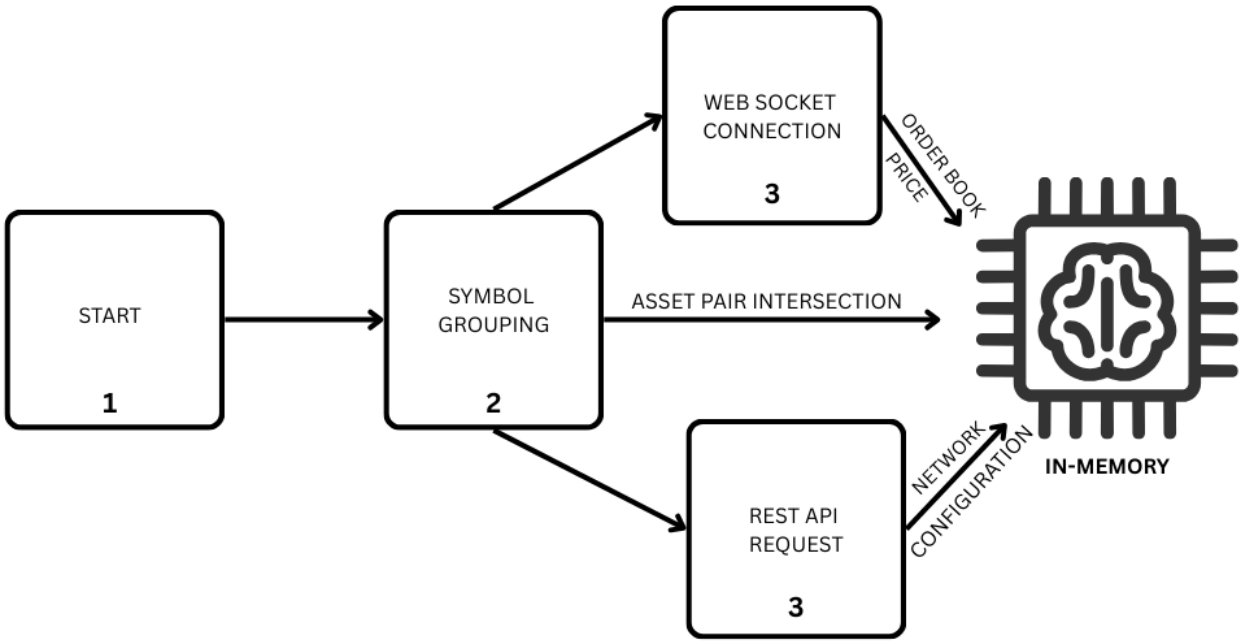


Figure 6.2 Data Retrieval Workflow

Following the integration of WebSocket and REST APIs outlined in the previous section, the data retrieval pipeline is responsible for continuously feeding fresh market data and metadata into the system. While Section 6.1.3 introduced the role and structure of the shared in-memory dictionary, this section focuses on the workflow that actively populates and updates it in real time.

At startup, the bot initiates its groundwork by querying each exchange via REST API to obtain the complete list of available trading pairs. It then performs an intersection to identify only those trading pairs that are supported on both Binance and OKX. These pairs are normalized into a consistent format and used to initialize entries within the in-memory store. This selection process is repeated periodically to adapt to listing or delisting events, ensuring that only actively tradable assets are considered.

In parallel, the bot collects supporting metadata for each asset, including withdrawal fees, minimum transfer limits, and the list of supported blockchain networks. This data is essential for determining whether an arbitrage opportunity can be completed using a

shared network between exchanges. Once this foundational information is gathered and structured, the system transitions into live data collection.

Real-time price and order book data is streamed via WebSocket connections to each exchange. For every tracked trading pair, the bot subscribes to ticker updates and the top levels of the order book. These streams are handled asynchronously, parsed according to each exchange's message format, and used to update the respective fields in the in-memory structure.

To support transparency and debugging, the system periodically writes snapshots of the in-memory state to JSON files. These dumps serve as diagnostic tools and can also be used to analyze the market's behavior retrospectively.

Overall, this pipeline forms the live data backbone of the bot, ensuring that all downstream components—including the arbitrage engine and result logger—operate on the most current and accurate market information available. By separating the retrieval and processing responsibilities, the system maintains both low latency and modularity, supporting scalable and reliable arbitrage detection.

6.3 Arbitrage Opportunity Result and Analysis

Building on the design principles outlined in Chapter 5, this section describes the actual implementation of the arbitrage engine—responsible for analysing opportunities between exchanges using the live data stored in memory—and how it produces structured simulation results for logging and notification.

The arbitrage engine operates as an asynchronous loop that continuously evaluates all eligible trading pairs previously identified by the data retrieval pipeline. Each cycle begins by iterating through the `symbol_prices` dictionary, which contains up-to-date market data for each overlapping pair across Binance and OKX.

Each completed simulation is stored in a structured `TradeSimulation` dataclass, which aggregates all key metrics related to the trade: timestamp, selected exchanges, budget used, final net profit, profit percentage, fee breakdowns, executed prices, and quantity movement before and after deductions. Supporting dataclasses such as `FeeDetails`, `QuantityDetails`, and `PriceDetails` provide a clean internal format for downstream modules to consume.

Field Name	Description
`timestamp`	Time when the simulation was executed; used for chronological tracking.
`asset_pair`	Trading pair under evaluation (e.g., BTCUSDT) in normalized format.
`initial_budget`	Starting USDT allocation used for the trade simulation.
`adjusted_budget`	Final budget after dynamic scaling based on available order book liquidity.
`buy_exchange`	Exchange selected for the buy side of the trade.
`sell_exchange`	Exchange selected for the sell side of the trade.
`network_used`	The blockchain network chosen for asset transfer (e.g., ERC20, TRC20).
`fee_details`	Includes maker/taker trading fees and withdrawal fees (from FeeDetails).
`quantity_details`	Asset quantities before and after deductions (from QuantityDetails).
`price_details`	Execution-level prices used in simulation (from PriceDetails).
net_profit	Final profit in USDT after all costs.
`profit_percentage`	Profit relative to the initial budget, expressed as a percentage.

6.3 TradeSimulation dataclass definition

By encapsulating simulation logic, result evaluation, and output handling within this cohesive module, the bot is able to continuously monitor market conditions, simulate realistic trades, and generate actionable insights for users—without requiring any live trade execution. This simulation-first approach provides a controlled environment to assess strategy performance while maintaining full architectural separation between analysis and execution.

6.4 Email Notification and Logging System

The notification and logging components represent the final step of the arbitrage pipeline, transforming simulated trade data into actionable and traceable outputs. After the arbitrage engine completes its evaluation cycle, all profitable results—those exceeding a defined profit threshold—are filtered and ranked. The most promising opportunity is passed to the notification system for immediate user alert, while all qualifying simulations are logged to a CSV file for later analysis.

To deliver real-time alerts, the system uses Python's email libraries to send formatted messages via Yahoo's SMTP server. Each email contains a concise summary of the trade opportunity, including the asset pair, exchange roles, selected network, trade size, applied fees, and profit metrics. This ensures the user is quickly informed about high-potential opportunities without needing to monitor the bot continuously. To avoid overload, only one email—based on the top trade per cycle—is sent during each evaluation loop.

Simultaneously, the bot appends all valid simulation results to a CSV log. Each row contains detailed metrics pulled directly from the `TradeSimulation` object, enabling users to track historical performance, verify simulation accuracy, or conduct further analysis. Together, the notification and logging subsystems provide both short-term responsiveness and long-term traceability, completing the bot's data lifecycle from detection to documentation.

6.5 Supporting Libraries and Tools

This section outlines the key Python libraries and standard modules employed throughout the project. These tools were selected to enable efficient concurrency, real-time communication, secure API access, and structured data handling, each essential to the bot's performance and reliability.

- **`websockets, requests`**: Used for real-time and REST-based data retrieval from Binance and OKX. WebSocket supports continuous streaming via WebSocket, while requests handle authenticated HTTP requests for network and fee metadata.

- **`smtplib, email.mime`**: Standard libraries used to format and send email alerts via Yahoo's SMTP service. These tools enable real-time notification delivery when a profitable arbitrage opportunity is detected.

- **`json, csv`**: json is used for parsing and storing API responses and in-memory snapshots, while csv is employed to log structured simulation outputs for later analysis.

- **`hmac, hashlib, base64:`** These libraries are combined to generate secure HMAC-SHA256 signatures required for authenticated REST API calls, particularly for OKX's API.

- **`time, datetime, pytz`**: Handle time-based operations including scheduling, timestamping logs, and managing time zones consistently across the system.

- **`asyncio`**: Enables asynchronous programming, allowing concurrent execution of WebSocket listening, periodic REST calls, and simulation cycles without blocking the event loop.

- **`dataclasses`**: Used to structure simulation output and internal data (e.g., TradeSimulation, FeeDetails), improving readability and maintainability of the code.

Chapter 7: Results

This chapter presents the evaluation and analysis of the arbitrage bot's performance based on real-time simulated trading data. The results are generated from a series of arbitrage opportunities identified and logged by the system between April 9th and April 14th, 2025. During this period, the bot continuously monitored price discrepancies across two major cryptocurrency exchanges Binance and OKX .

Each arbitrage opportunity was recorded using a structured format defined by the `TradeSimulation` dataclass. This includes comprehensive metadata such as the trading pair, the direction of the trade (buy and sell exchanges), the blockchain network used for transfer, fee breakdowns, quantities handled, and both the absolute and percentage profit simulated. All simulations account for realistic trading conditions, including exchange fees, withdrawal costs, and available liquidity based on top-level order book depth.

To ensure the detection of only meaningful and executable arbitrage scenarios, a minimum profit threshold of 0.2% was set. This margin serves as a buffer against potential slippage, especially under rapidly changing market conditions. Additionally, each simulation was conducted using a maximum capital allocation of 10,000 USDT, although this budget was subject to dynamic adjustment based on the depth of the order book and available liquidity, thereby reflecting real-world constraints on trade execution size.

The purpose of this results chapter is to assess the practical viability of the bot’s arbitrage logic. It explores patterns in exchange activity, asset distribution, profit margins, and network selection strategies, with the aim of identifying key factors that contribute to successful arbitrage trades. This analysis helps highlight strengths, uncover inefficiencies, and lay the groundwork for further improvements to the bot’s strategy and architecture.

7.1 Exchange Activity Analysis

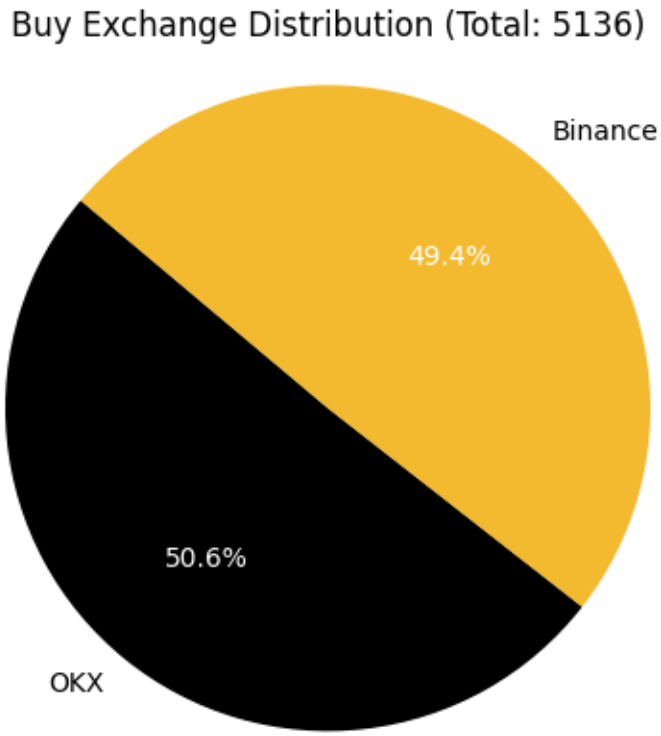


Figure 7.1 Frequency of Arbitrage Routes by Exchanges Buying Direction

The pie chart above visualizes the directional distribution of arbitrage trades executed by the bot between Binance and OKX. The data is grouped by trade flow — whether the bot bought from Binance and sold on OKX, or vice versa.

- OKX → Binance: 2597 trades
- Binance → OKX: 2539 trades

This relatively balanced distribution indicates that arbitrage opportunities were almost equally likely in both directions, suggesting that price discrepancies between Binance and OKX were not consistently biased toward one exchange. This balance reinforces the bot's versatility, showing that it did not rely on a specific directional strategy but rather dynamically reacted to whichever side offered a better price differential after considering fees and liquidity.

The slight edge in the OKX → Binance direction may hint at marginally better conditions — such as tighter spreads, lower fees, or faster withdrawal processing — on OKX when initiating a trade. This is something that could be explored further in fee analysis or network performance evaluation.

7.2 Asset Pair Distribution

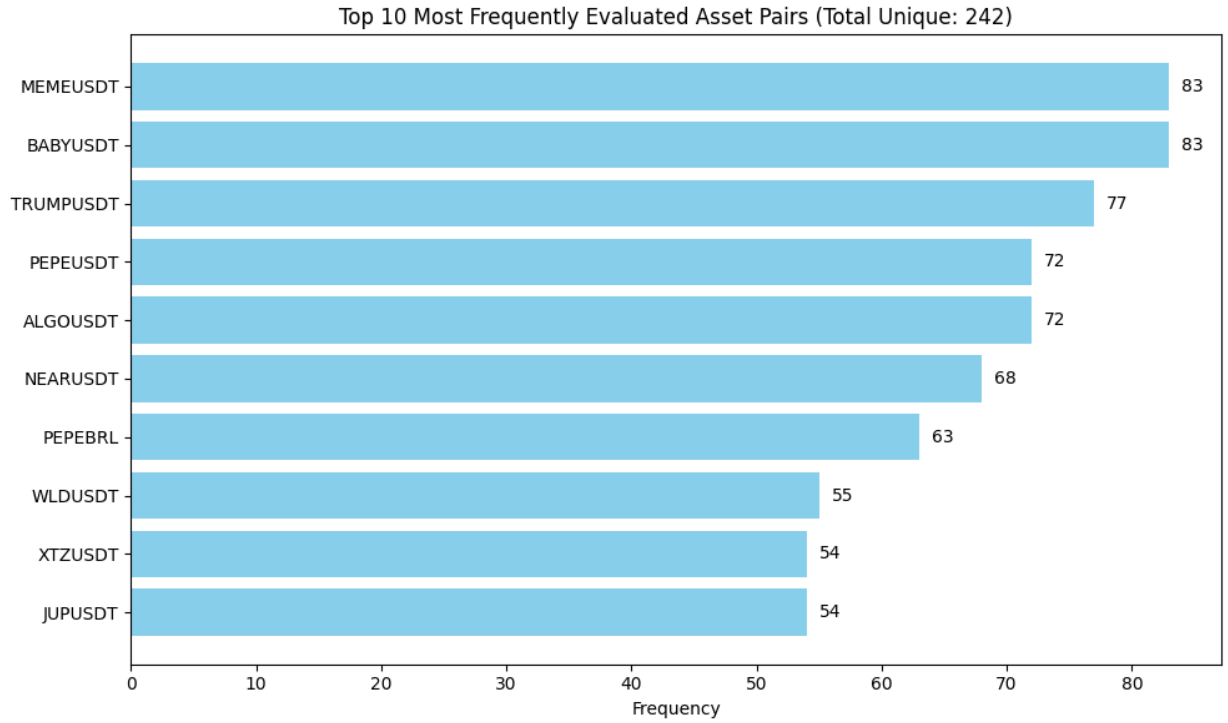


Figure 7.2 Top 10 Most Frequently Arbitraged Asset Pairs

The bar chart above highlights the top 10 most frequently arbitraged asset pairs detected by the bot during its operation. Notably, meme and trending coins dominate the list:

- BABYUSDT and MEMEUSDT lead the chart, each with 83 opportunities.
- Closely following are TRUMPUSDT, ALGOUSDT, and PEPEUSDT, all of which are relatively volatile or community-driven assets.

This distribution indicates that high-volatility and hype-driven tokens often present more frequent arbitrage opportunities. These assets are usually characterized by rapid price shifts, speculative trading behaviour, and inconsistent liquidity across exchanges — all of which contribute to short-lived price discrepancies.

Another observation is that stable-to-stable pairs (e.g., major assets like BTC/USDT or ETH/USDT) are absent from the top spots. This aligns with expectations, as larger cap assets tend to have tighter spreads and higher efficiency between exchanges.

7.3 Profitability Analysis

timestamp	asset_pair	buy_exchange	sell_exchange	net_profit_percentage
2025-04-14T13:26:58.753883	ANIMEUSDT	OKX	Binance	13.65868049974199
2025-04-13T20:41:10.737623	OMUSDT	OKX	Binance	11.991413698137158
2025-04-14T12:37:19.998053	ANIMEUSDT	OKX	Binance	11.146074068067453
2025-04-14T12:42:21.893995	ANIMEUSDT	OKX	Binance	10.495452842460198
2025-04-13T20:45:45.888844	OMUSDT	OKX	Binance	9.569214326860545
2025-04-13T19:46:18.521447	OMUSDT	OKX	Binance	8.659376045563201
2025-04-13T20:38:46.954312	OMUSDT	OKX	Binance	8.502290405911934
2025-04-14T12:32:35.087389	ANIMEUSDT	OKX	Binance	7.948254193732279
2025-04-13T21:03:47.629321	OMUSDT	OKX	Binance	5.936134994300641
2025-04-13T21:00:31.784968	OMUSDT	OKX	Binance	5.700750398979698

Figure 7.3 Top 10 Most Profitable Arbitrage Opportunities by Net Profit Percentage

In addition to analysing average profitability trends, a closer examination of the top 10 most profitable arbitrage simulations reveals the bot's ability to detect highly lucrative opportunities. These are ranked by net profit percentage, which reflects the relative gain compared to the initial budget allocated in the simulation.

Notably, the most profitable simulations consistently involved altcoins or low-cap assets such as ANIME and OM, often paired with networks offering lower withdrawal fees and faster transaction finality. These trades likely reflect short-lived price imbalances during periods of market volatility or shallow liquidity. The results underscore the importance of real-time responsiveness and network selection in capturing high-yield opportunities.

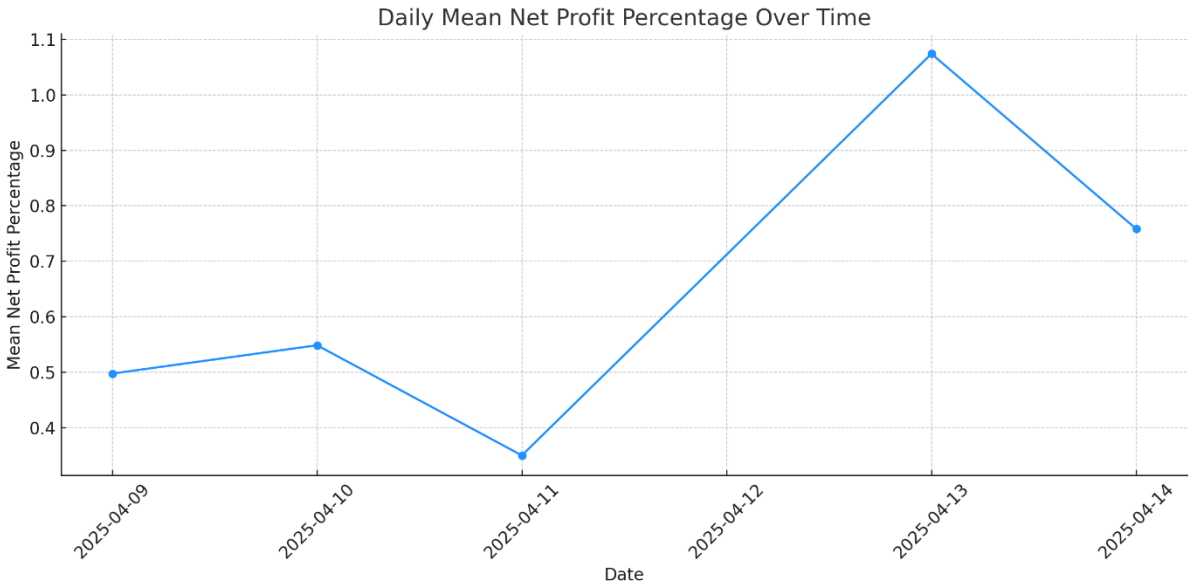


Figure 7.4 Daily Mean Net Profit Percentage Over Time

The distribution of net profit percentages provides insight into the typical efficiency and effectiveness of the arbitrage opportunities identified by the bot. As shown in the corresponding histogram, the majority of trades yielded profit percentages clustered around the 0.5% to 1.0% range, with the average sitting at approximately 0.77%. This concentration suggests that the bot predominantly identified small, low-risk opportunities — a characteristic expected in real-world arbitrage, where price discrepancies across reputable exchanges are generally narrow and short-lived.

7.4 Network Utilization

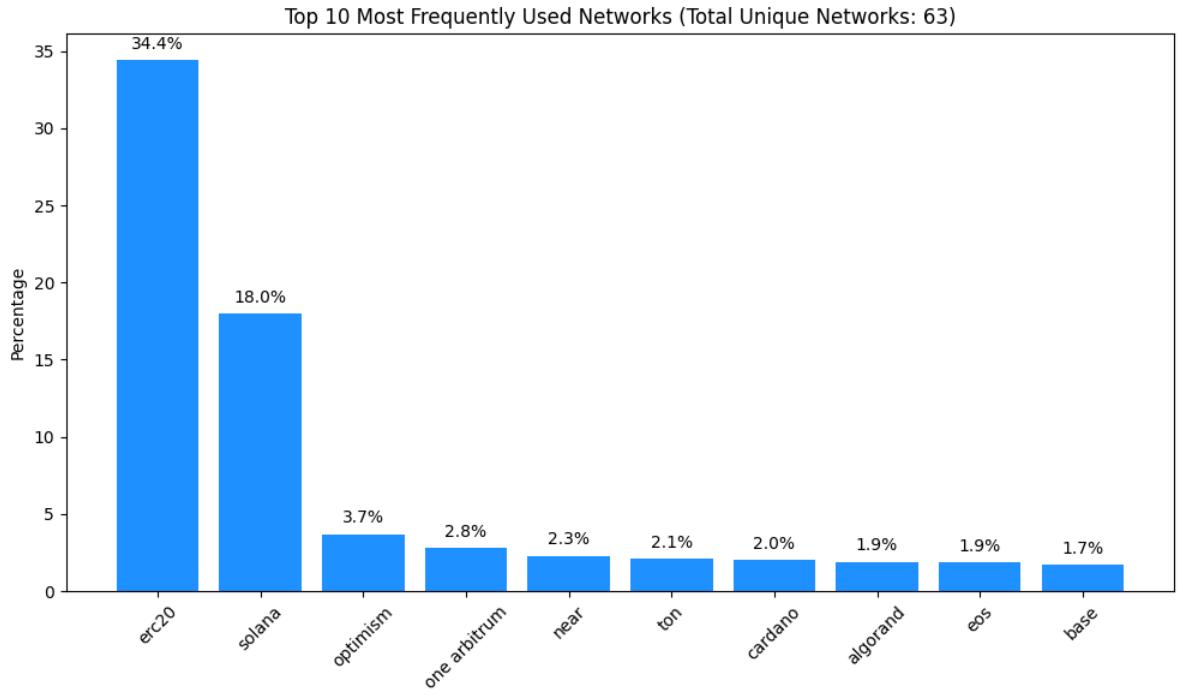


Figure 7.5 Frequency of Blockchain Network Usage in Arbitrage Simulations

The analysis reveals that the most frequently used network was ERC20, which accounted for 1,766 trades, followed by Solana with 923 occurrences. While ERC20 is known for its broad asset support and compatibility across exchanges, its relatively high fees may make it less optimal in low-margin scenarios. Nevertheless, its dominance in the results suggests that many assets default to this network due to availability and liquidity, despite the cost.

Solana, the second most utilized network, is known for its high throughput and low fees. Its popularity within the dataset suggests it offered a good balance between accessibility and cost-efficiency, especially for altcoins with Solana-native implementations. Other notable networks include Optimism, Arbitrum One, and TON, all of which are layer-2 or high-performance alternatives that provide low-fee and fast-settlement options which is ideal for arbitrage scenarios requiring quick asset transfers.

7.5 Summary and Key Observations

The results presented in this chapter demonstrate the practical viability of the cryptocurrency arbitrage bot in identifying profitable trading opportunities across major exchanges. Throughout the analysed period, the bot logged over 5,000 arbitrage simulations, with an average net profit of 36.93 USDT and an average profit percentage of 0.77% per trade. These findings highlight the bot's capability to exploit market inefficiencies even under realistic trading conditions, including network and fee constraints.

A key observation is the balanced directionality of arbitrage trades between Binance and OKX, indicating that price discrepancies occurred in both directions with comparable frequency. The most frequently arbitraged asset pairs were dominated by high-volatility altcoins and meme tokens, such as BABYUSDT, MEMEUSDT, and PEPEUSDT. These assets are often subject to rapid price fluctuations, providing fertile ground for short-term arbitrage opportunities.

In terms of profitability, while most trades yielded modest returns within the 0.5–1.0% range, the bot also identified outlier opportunities with profit margins exceeding 10%, particularly involving lesser-known assets and efficient transfer networks like Arbitrum One. This supports the hypothesis that significant arbitrage windows occasionally emerge in low-liquidity or high-volatility market conditions.

The analysis of network utilization further emphasizes the importance of adaptive network selection. Although ERC20 was the most commonly used network, likely due to its broad compatibility, networks such as Solana, Optimism, and TON played an important role in optimizing profitability by minimizing withdrawal fees and reducing latency.

Overall, the results confirm that the bot not only functions correctly under live market conditions but also intelligently adapts to network and asset-specific dynamics. These findings support its effectiveness as a tool for arbitrage-based strategies and lay the foundation for future enhancements such as real trade execution, more advanced route optimization, or multi-hop arbitrage.

Chapter 8: Discussion

This chapter provides a critical reflection on the performance and broader implications of the arbitrage bot, based on the evaluation of simulation results presented in Chapter 7. It begins by assessing how effectively the system met its original objectives and the extent to which it demonstrated the viability of automated arbitrage under realistic market conditions. The chapter also acknowledges key limitations encountered during development and testing, including technical constraints, regulatory boundaries, and the absence of live trade execution. Based on these insights, potential directions for future improvement are proposed, with a focus on enhancing execution realism, expanding market coverage, and exploring alternative arbitrage strategies. Finally, the chapter concludes with a summary of the project's overall contribution to the field of automated trading and its relevance within the evolving landscape of cryptocurrency markets.

8.1 Evaluation

The primary objective of this project was to design and implement a cryptocurrency arbitrage bot capable of autonomously identifying and simulating profitable trading opportunities across two major exchanges—Binance and OKX. The system was expected to retrieve real-time market data, account for trading and network fees, and simulate full arbitrage cycles with sufficient realism to evaluate profitability under practical conditions. Based on the results presented in Chapter 7, it is evident that these objectives were successfully met.

Throughout the testing period, the bot operated continuously, demonstrating strong system reliability and responsiveness under live data conditions. Its ability to detect and log over 5,000 arbitrage opportunities, each processed using real-time price data and order book depth, indicates that the core architecture performed as intended. The decision to rely on in-memory data structures significantly reduced latency and enabled efficient access to price, liquidity, and fee information across both exchanges. This, coupled with asynchronous execution using Python's `asyncio` framework, allowed for smooth, concurrent processing of WebSocket feeds, REST API data, and simulation logic.

The simulation methodology also proved robust. By accounting for slippage, trading fees, withdrawal costs, and liquidity limits, the bot generated profit estimates that closely resemble real-world execution outcomes. The inclusion of a 0.2% profit threshold further improved the system's credibility, effectively filtering out false positives and margin-neutral trades. Additionally, the use of a dynamically adjusted budget—capped at 10,000

USDT—ensured that simulations remained feasible and respected the liquidity available at the time of trade.

These results align with recent academic findings suggesting that although crypto markets are becoming more efficient, price discrepancies still persist in certain market segments (Jalan & Matkovskyy, 2023). Furthermore, the balanced trade directionality between exchanges and consistent profit detection across multiple asset pairs reflect the system's adaptability to diverse market conditions and real-time changes in spread dynamics.

In summary, the bot met its design objectives by effectively simulating arbitrage opportunities in real time, producing structured and actionable insights without the need for actual trade execution. The system's performance supports its potential use as a foundational framework for live arbitrage strategies and reinforces the utility of automation in identifying and exploiting inefficiencies in evolving cryptocurrency markets (Foley, Karlsen, & Putniņš, 2022).

8.2 Limitations

While the arbitrage bot successfully achieved its primary objectives, several limitations must be acknowledged—both in terms of system scope and external constraints that affected the realism and generalizability of the results.

Firstly, the most significant limitation is the absence of live trade execution. All arbitrage opportunities were processed within a simulation environment, using real-time data but without executing actual buy or sell orders on the exchanges. As a result, the system could not account for real-world variables such as trade confirmation delays, unexpected network congestion, or slippage due to competing traders. While the simulation logic attempted to realistically replicate these conditions by incorporating fees and liquidity depth, the lack of true execution inherently limits the practical validation of the strategy.

Secondly, regulatory constraints in the United Kingdom restricted access to certain exchange features and services, especially around advanced trading tools or leveraged positions. These limitations not only affected the selection of exchanges but also prevented integration with more advanced market-neutral strategies that require features such as short selling or margin trading. The scope of the project was therefore narrowed to spatial arbitrage between Binance and OKX, excluding potentially profitable routes on other platforms.

The project was also impacted by data limitations, particularly regarding order book depth. OKX, for instance, only provides access to the top five levels of its order book, limiting the accuracy of liquidity estimation. This restricted the system's ability to simulate large-volume trades or properly model the effects of price impact. Although the budget was dynamically adjusted to avoid unrealistic assumptions, this constraint may still have skewed profitability estimates in certain scenarios.

Another technical constraint lies in the fee data granularity and network inconsistencies. Withdrawal fees are not always consistent across time or network conditions, and the bot's logic assumes that the most recently retrieved fee applies at the time of trade. This simplification may not hold during periods of network congestion or fee volatility, particularly on chains like Ethereum (ERC20), where gas prices can spike suddenly.

Finally, the limited diversity of exchanges and trading pairs restricts the generalizability of the results. While Binance and OKX provide a solid foundation for cross-exchange comparison, expanding the system to include more exchanges would allow for broader coverage and potentially reveal more frequent or profitable arbitrage windows. Similarly, access to decentralized exchanges (DEXs) or cross-chain bridges could further enhance the bot's adaptability in an increasingly fragmented market.

In sum, while the project successfully demonstrates a robust arbitrage detection framework, its current scope is bounded by simulation-only testing, exchange-level constraints, partial data access, and regulatory considerations. These limitations highlight areas for improvement and provide clear direction for future enhancements.

8.3 Future work

Building upon the foundation established in this project, several areas have been identified for future development to enhance the effectiveness, scope, and realism of the arbitrage bot.

The most immediate and impactful improvement would be to extend the system from simulation-based evaluation to live trade execution. This transition would require the integration of secure exchange account access, order placement logic, and real-time error handling to manage partial fills, transaction delays, or execution failures. Implementing this functionality would allow for real-world validation of arbitrage performance and enable the development of a true profit-generating system.

Additionally, incorporating more exchanges would significantly broaden the opportunity landscape. Currently, the bot is limited to Binance and OKX; expanding to include other

centralized or even decentralized exchanges could increase the volume and diversity of detected arbitrage windows. This would not only improve trade frequency but also allow the bot to exploit price inefficiencies across exchanges with varying regional, regulatory, or liquidity profiles.

There is also substantial potential to explore alternative arbitrage strategies beyond the spatial (cross-exchange) model implemented here. For instance, the architecture could be adapted to support more complex strategies such as triangular arbitrage, which involves identifying price inefficiencies across three trading pairs within a single exchange. While this project did not define or implement such strategies, they represent a promising direction for future work due to their ability to exploit internal inconsistencies without requiring inter-exchange transfers.

Another area for enhancement is the improvement of order book modeling, particularly in simulating more realistic slippage and partial fills for larger trade volumes. This could involve deeper order book analysis or interpolation of missing depth levels on exchanges like OKX, which currently expose limited data. A more nuanced model of liquidity would help produce even more accurate estimates of trade feasibility and net profitability.

Collectively, these improvements would evolve the current system into a more sophisticated and production-ready arbitrage platform capable of not only detecting, but also executing and managing live trades across a diverse and rapidly changing crypto ecosystem.

8.4 Conclusion

This project set out to design and implement a cryptocurrency arbitrage bot capable of identifying and simulating profitable opportunities across multiple exchanges in real time. By combining WebSocket and REST API integrations, a memory-efficient architecture, and fee-aware trading logic, the bot successfully detected thousands of viable arbitrage scenarios between Binance and OKX. Each simulation accounted for practical constraints such as trading fees, withdrawal costs, and limited liquidity, providing a realistic measure of net profitability.

The motivation behind this project stemmed from real-world inefficiencies like the Kimchi Premium, which historically highlighted persistent price disparities between geographically isolated exchanges. While such extremes are now rare, the results from this bot suggest that similar inefficiencies—albeit smaller and shorter-lived—continue to exist in today's fragmented and fast-moving crypto markets. Through continuous monitoring

and adaptive simulation, the bot demonstrated the ability to capture these fleeting opportunities with measurable consistency.

Compared to earlier arbitrage systems like Blackbird, which relied solely on market-neutral strategies and REST-based polling, this project advances the state of practice by incorporating spatial arbitrage with live order book data, multi-network fee calculations, and asynchronous data processing. This allowed for more realistic simulations and broader coverage across volatile, low-cap assets and alternative blockchain networks.

In light of recent studies that suggest increased market efficiency and institutional convergence are diminishing arbitrage potential, these findings provide a nuanced perspective. The bot's detection of over 5,000 filtered opportunities across just two exchanges indicates that arbitrage remains viable—particularly in less liquid markets or during periods of volatility. Rather than being eliminated, inefficiencies appear to be evolving, requiring more agile and intelligent systems to exploit them.

Although several limitations remain, such as the absence of real trade execution and regulatory constraints, the project demonstrates that a modular, asynchronous, and fee-conscious arbitrage bot can serve as both a research tool and a foundation for further development. With continued enhancement—such as integration with live execution, broader exchange coverage, and support for alternative strategies—this system holds significant potential as a practical contributor to both profit generation and market efficiency in the crypto ecosystem.

References

Antonopoulos, A. M., & Wood, G. (2018). *Mastering Ethereum: Building smart contracts and DApps*. O'Reilly Media.

Burns, A. (2019). Exploring arbitrage in cryptocurrency markets. *Journal of Financial Markets and Institutions*.

Catalini, C., & Gans, J. S. (2016). Some simple economics of the blockchain. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.2874598

CCXT. (n.d.). *A JavaScript / Python / PHP cryptocurrency trading library with support for many crypto exchange markets and merchant APIs*. https://github.com/ccxt/ccxt

Chowdhury, A. (2024). Global cryptocurrency market cap surpasses $2 trillion. *Financial Technology Reports*. https://www.financialtechreports.com/crypto-market-cap-2024

CoinMarketCap. (2023). *Cryptocurrency market capitalization data*. https://coinmarketcap.com

De Filippi, P., & Wright, A. (2018). *Blockchain and the law: The rule of code*. Harvard University Press.

Foley, S., Karlsen, J. R., & Putniņš, T. J. (2019). Sex, drugs, and bitcoin: How much illegal activity is financed through cryptocurrencies? *The Review of Financial Studies, 32*(5), 1798–1853. https://doi.org/10.1093/rfs/hhz015

Foley, S., Karlsen, J. R., & Putniņš, T. J. (2022). Crypto prices and exchange characteristics: Price dispersion and the role of regulation. *Journal of Financial Markets, 63*, 100715. https://doi.org/10.1016/j.finmar.2021.100715

Gandal, N., & Halaburda, H. (2014). Can we predict the bitcoin bubble? A multivariate bubble test. *Technological Forecasting and Social Change, 99*, 275–287.

Gandal, N., & Halaburda, H. (2016). Can we predict the winner in a market with network effects? Competition in cryptocurrency market. *Games, 7*(3), 16. https://doi.org/10.3390/g7030016

Harvey, C. R., & Ramachandran, A. (2018). Cryptocurrency and blockchain: Technical overview. *SSRN Electronic Journal*.

Hassani, H., Huang, X., & Silva, E. S. (2019). Digitalisation and big data mining in banking. *Economies, 8*(20), 1–18. https://doi.org/10.3390/economies8010020

Houben, R., & Snyers, A. (2018). *Cryptocurrencies and blockchain: Legal context and implications for financial crime, money laundering, and tax evasion*. European Parliament.

Jalan, A., & Matkovskyy, R. (2023). Systemic risks in the cryptocurrency market: Evidence from the FTX collapse. *Finance Research Letters, 53*, 103670. https://doi.org/10.1016/j.frl.2023.103670

Kabašinskas, A., & Šutienė, K. (2021). Key roles of crypto-exchanges in generating arbitrage opportunities. *Entropy, 23*(4), 455. https://doi.org/10.3390/e23040455

Kyriazis, N. (2020). Market efficiency and arbitrage opportunities in cryptocurrency markets. *Journal of Financial Economics*.

Liu, Y., & Tsyvinski, A. (2018). The cross-section of cryptocurrency returns. *National Bureau of Economic Research*.

Makarov, I., & Schoar, A. (2020). Trading and arbitrage in cryptocurrency markets. *Journal of Financial Economics, 135*(2), 293–318. https://doi.org/10.1016/j.jfineco.2019.07.001

Mattke, J., Maier, C., Reis, L., & Weitzel, T. (2021). Bitcoin investment: A mixed methods study of investment motivations. *European Journal of Information Systems, 30*(3), 246–267.

Moore, T., & Christin, N. (2013). Beware the middleman: Empirical analysis of bitcoin-exchange risk. In *Financial Cryptography and Data Security* (pp. 25–33).

Nakamoto, S. (2008). *Bitcoin: A peer-to-peer electronic cash system*. https://bitcoin.org/bitcoin.pdf

Pagnotta, E. S., & Buraschi, A. (2020). An equilibrium valuation of Bitcoin and decentralized network assets. *Journal of Financial Economics*.

Park, S., Tian, S., & Zhao, H. (2019). Global Bitcoin markets and local regulations. *Asia-Pacific Journal of Financial Studies, 48*(5), 675–705. https://doi.org/10.1111/ajfs.12275

Petukhina, A., Trimborn, S., & Härdle, W. K. (2021). Investing with cryptocurrencies – A liquidity constrained investment approach. *Journal of Empirical Finance, 63*, 75–96. https://doi.org/10.1016/j.jempfin.2021.04.005

Python Software Foundation. (2023). *smtplib — SMTP protocol client*. Python 3 Documentation. https://docs.python.org/3/library/smtplib.html

Qin, K., Zhou, L., & Gervais, A. (2020). Quantifying blockchain extractable value: How dark is the forest? In *Proceedings of the IEEE Symposium on Security and Privacy* (pp. 1399–1415).

Reference: FIX Protocol Ltd. (n.d.). What is FIX? Retrieved March 31, 2025, from https://www.fixtrading.org/what-is-fix/

Shadab, H. B. (2014). Regulating Bitcoin and blockchains. *Cardozo Legal Studies Research Paper, No. 478*.

Silver Institute. (2024). *World Silver Survey 2024*. https://www.silverinstitute.org/world-silver-survey-2024/

Teti, E. (2017). *Blackbird Bitcoin Arbitrage* [GitHub repository]. https://github.com/butor/blackbird

Wang, G., Peng, Y., Huang, S., & Cheng, W. (2022). Crypto triangular arbitrage: A path-dependent approach for trading strategy generation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22)*, 1340–1350. https://doi.org/10.1145/3534678.3537553

Xiong, W., & Luo, J. (2024). Cryptocurrency arbitrage and market efficiency. *Financial Innovation, 10*(2), 45–58. https://doi.org/10.1186/s40854-024-00525-8

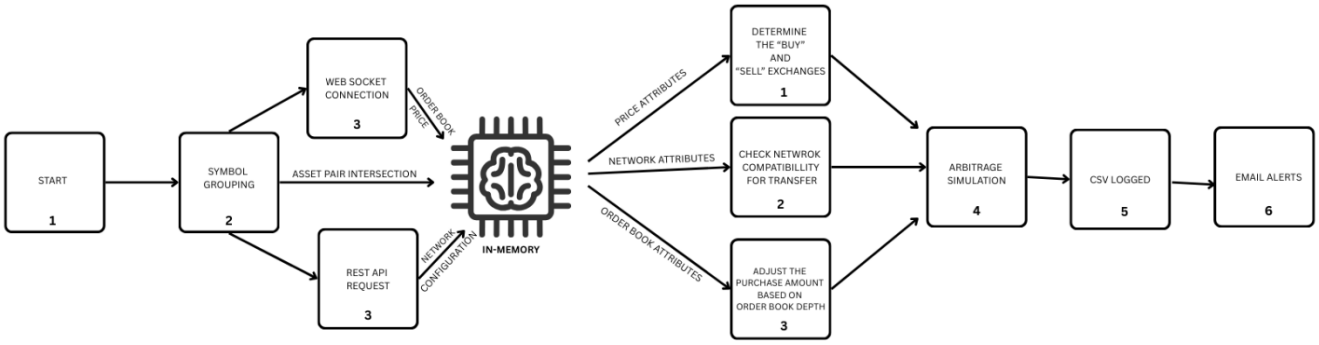
Zheng, Z., Xie, S., Dai, H., Chen, X., & Wang, H. (2018). An overview of blockchain technology: Architecture, consensus, and future trends. In *Proceedings of the IEEE International Conference on Big Data* (pp. 557–564).

Zohar, A. (2015). Bitcoin: Under the hood. *Communications of the ACM, 58*(9), 104–113.

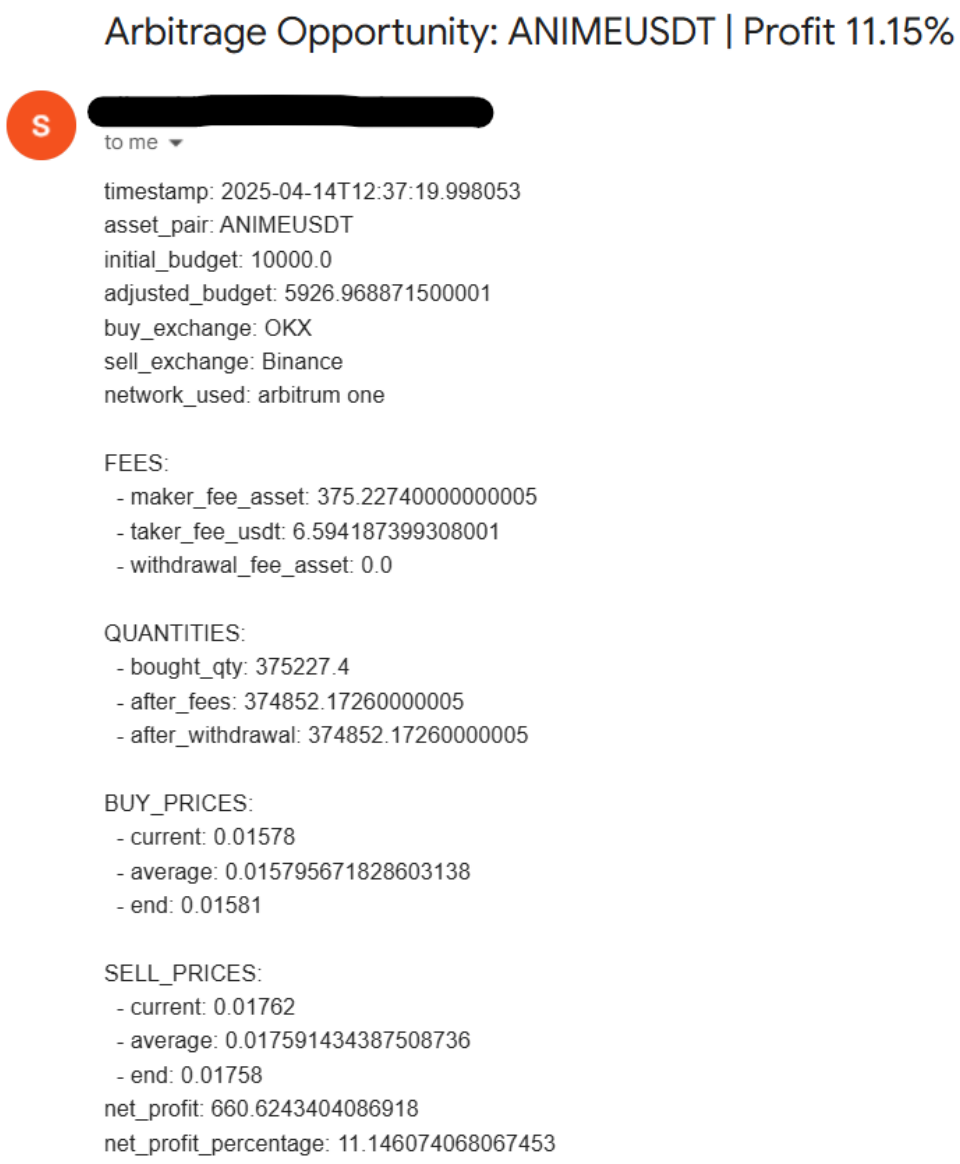
Appendix A
Sample Arbitrage Simulation CSV Log

Field	Value
timestamp	2025-04-09T19:58:00.567680
asset_pair	NULSUSDT
initial_budget	10000.0
adjusted_budget	550.5005729155
buy_exchange	OKX
sell_exchange	Binance
network_used	nuls
fees	{'maker_fee_asset': 10.739938565, 'taker_fee_usdt': 0.553276
quantities	{'bought_qty': 10739.938564999999, 'after_fees': 10729.1986
buy_prices	{'current': 0.051, 'average': 0.05125732978673701, 'end': 0.0
sell_prices	{'current': 0.0516, 'average': 0.051567432859121486, 'end': 0
net_profit	2.222967528640993
net_profit_percentage	0.40380839512445177

Appendix B
System Architecture Diagram



Appendix C
Sample Email Notification



Arbitrage Opportunity: ANIMEUSDT | Profit 11.15%

S [redacted]

to me

timestamp: 2025-04-14T12:37:19.998053
asset_pair: ANIMEUSDT
initial_budget: 10000.0
adjusted_budget: 5926.968871500001
buy_exchange: OKX
sell_exchange: Binance
network_used: arbitrum one

FEES:
- maker_fee_asset: 375.22740000000005
- taker_fee_usdt: 6.594187399308001
- withdrawal_fee_asset: 0.0

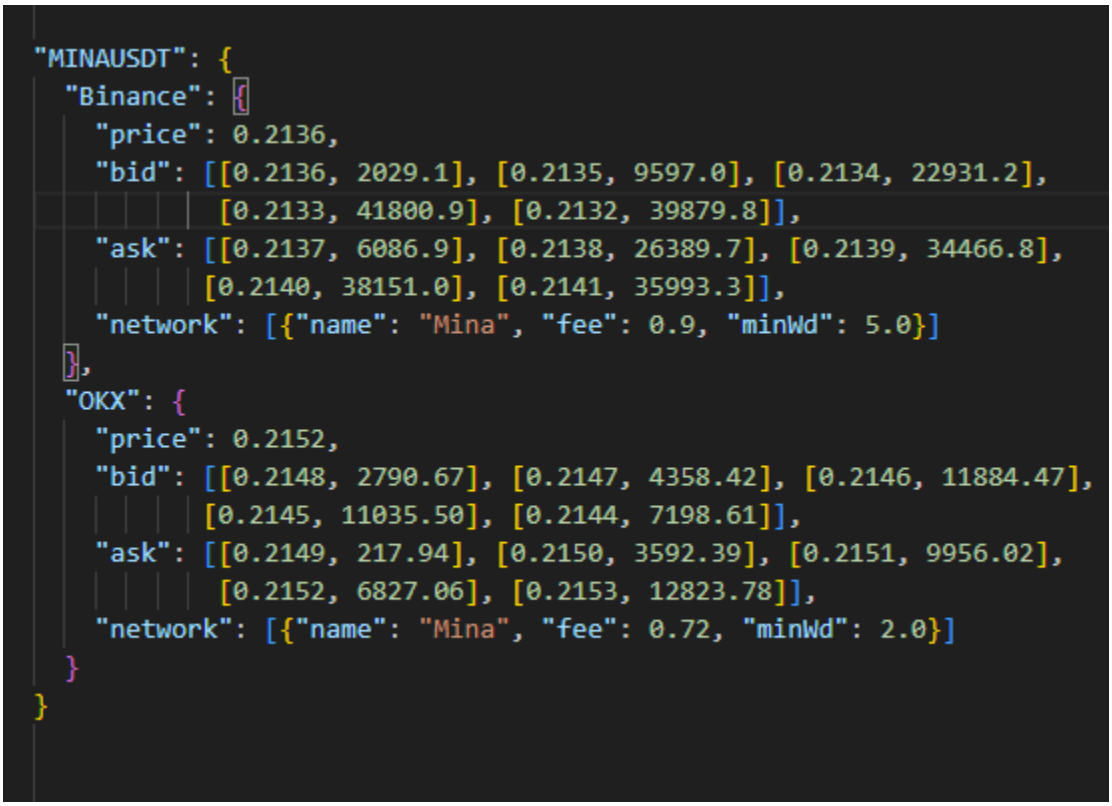
QUANTITIES:
- bought_qty: 375227.4
- after_fees: 374852.17260000005
- after_withdrawal: 374852.17260000005

BUY_PRICES:
- current: 0.01578
- average: 0.015795671828603138
- end: 0.01581

SELL_PRICES:
- current: 0.01762
- average: 0.017591434387508736
- end: 0.01758

net_profit: 660.6243404086918
net_profit_percentage: 11.146074068067453

Appendix D
Sample Updated In memory symbol_price




```
"MINAUSDT": {
  "Binance": {
    "price": 0.2136,
    "bid": [[0.2136, 2029.1], [0.2135, 9597.0], [0.2134, 22931.2],
            [0.2133, 41800.9], [0.2132, 39879.8]],
    "ask": [[0.2137, 6086.9], [0.2138, 26389.7], [0.2139, 34466.8],
            [0.2140, 38151.0], [0.2141, 35993.3]],
    "network": [{"name": "Mina", "fee": 0.9, "minWd": 5.0}]
  },
  "OKX": {
    "price": 0.2152,
    "bid": [[0.2148, 2790.67], [0.2147, 4358.42], [0.2146, 11884.47],
            [0.2145, 11035.50], [0.2144, 7198.61]],
    "ask": [[0.2149, 217.94], [0.2150, 3592.39], [0.2151, 9956.02],
            [0.2152, 6827.06], [0.2153, 12823.78]],
    "network": [{"name": "Mina", "fee": 0.72, "minWd": 2.0}]
  }
}
```

Appendix E
Data Retrieval Pipeline & Data Processing Pipeline Pseudo Concurrent Code

```python
async def arbitrage_loop(symbol_prices, budget, profit_threshold, check_interval=10):
    """
    Arbitrage Logic. This function will retrive the information from in-memory metadata symbol_price,
    and finding profitable arbitrage by simulating the trade. It will send email notification alert
    to the user for the best profitbale percentage in that cycle. This function also add random
    sleep time between 2-5 minutes for the network congestion simulation.
    """
    while True:
        # all details about profitable arbitrage in one cycle
        opportunities = arbitrage_checker(symbol_prices, budget, profit_threshold)

        if opportunities:
            # write to csv file
            log_arbitrage_opportunities(opportunities)

            # Send Email to the user about the best Arbitrage Execution
            best = max(opportunities, key=lambda x: x.net_profit_percentage)
            email_alert(best)

            # Simulate transfer delay
            wait_time = random.randint(120, 300)
            print(f"⏳ Sleeping {wait_time} seconds to simulate transfer...")
            await asyncio.sleep(wait_time)
        else:
            await asyncio.sleep(check_interval)

async def main():
    """
    Run all components concurrently:
      - Binance and OKX WebSocket connections for live price updates.
      - REST-based withdrawal fee updaters.
      - Periodic saving of symbol_prices and USDT_Network.
      - Daily recalculation of symbol grouping.
      - Periodic arbitrage checking.
    """

    await asyncio.gather(

        # Updating all of the atributes for the in-memory metadata
        binance_combined_connection(),
        okx_combined_connection(),
        update_binance_withdrawal_fees(),
        update_okx_withdrawal_fees(),
        periodic_save(),
        recalc_symbols_daily(),

        # retriving attributes and cycle the arbitrage
        arbitrage_loop(symbol_prices, ARBITRAGE_BUDGET, ARBITRAGE_PROFIT_THRESHOLD)
    )

if __name__ == "__main__":
    try:
        asyncio.run(main())
```