UDC 004.94 https://doi.org/10.20535.2786-8729.5.2024/316563

THE AUTOMATIC CRYPTOCURRENCY TRADING SYSTEM USING A SCALPING STRATEGY

Elisa Beraudo *
National Technical University of Ukraine
"Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine
https://orcid.org/0000-0001-7550-3620

Yurii Oliinyk
National Technical University of Ukraine
"Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine
https://orcid.org/ 0000-0002-7408-4927

*Corresponding author: elisa.beraudolive@gmail.com

The study focuses on the development and implementation of an automated system for scalping strategies in cryptocurrency markets. Scalping, a high-frequency trading strategy, aims to generate profits from small price fluctuations. The primary goal of the research is to create an automated trading bot that addresses critical issues such as latency, risk management, scalability, and reliability in real-world market conditions. To achieve this, the following objectives were defined: develop a novel scalping method, implement a software solution to integrate the method into an automated trading system, and evaluate its effectiveness through experimental testing.

The research methodology utilized technical indicators, including the Exponential Moving Average (EMA) and Volume Weighted Average Price (VWAP). Pseudocode was created to illustrate the decision-making process, incorporating key parameters such as smoothing factors, time periods, and thresholds for trade execution. The software architecture consists of modules: Binance exchange integration, data collection and management, strategy analysis, trade execution, and historical data storage. Technologies such as PostgreSQL, Redis, WebSocket, and Python libraries (Pandas, NumPy, TA-Lib) were employed to ensure the robustness and efficiency of the system.

Experiments were conducted using the BTC/USDT trading pair, known for its high liquidity and volatility. The system was tested on hardware featuring an Intel Core i7-10700K processor, 32 GB of RAM, and a 1 Gbps network connection. A comparative analysis between the scalping strategy and a trend-following strategy demonstrated the advantages of scalping in volatile markets. The scalping bot executed 15 trades (13 successful) within two hours, achieving a total profit of 120 USDT.

Performance metrics, including latency (15–50 ms), signal processing time, CPU utilization (5–55%), and memory usage (120–2100 MB), were measured. The results confirmed the system's modular architecture and its ability to scale linearly with increasing trading volumes.

The findings validate the effectiveness of the proposed method and the reliability of the developed system in real-world conditions. Future research may focus on optimizing algorithms to reduce resource consumption and integrating advanced risk management techniques to enhance performance.

Key words: automated trading, scalping, cryptocurrency, Binance API, algorithmic trading.

1. Introduction

The rapid development of the cryptocurrency market has created new challenges and opportunities in the field of algorithmic trading. Among these challenges are the high volatility of cryptocurrencies, the necessity of real-time data analysis, and the need for efficient strategies to maximize profit in highly dynamic environments. Scalping strategies, which focus on generating profit from small price fluctuations, are among the most promising approaches. However, their implementation requires advanced algorithm optimization and seamless integration with trading platforms.

This study addresses the scientific problem of developing efficient methods and tools for automated cryptocurrency trading, focusing on improving the speed and accuracy of executing

trading operations. The relevance of this topic lies in its potential to enhance the effectiveness of trading systems in a market that is increasingly dominated by automation and algorithmic strategies.

2. Literature review and problem statement

The implementation of automated trading systems using scalping strategies has been widely studied in recent years, as the cryptocurrency market presents unique challenges and opportunities. Existing literature provides valuable insights into the tools, algorithms, and methodologies employed to address the core challenges of latency, risk management, scalability, and reliability in high-frequency trading.

One of the foundational areas of research involves addressing latency in automated trading systems. Studies such as those by Miles and Cliff [2] highlight the importance of real-time data streaming technologies, like WebSocket, for minimizing delays in data retrieval and trade execution. WebSocket's ability to facilitate low-latency, bidirectional communication between clients and servers has made it a popular choice for trading platforms. Additionally, the use of in-memory data structures, such as those provided by Redis, has been identified as a key mechanism for speeding up temporary data storage and access during high-frequency trading operations. Yuxin Fan et al. [3] extend this discussion by exploring machine learning-based approaches for optimizing real-time data processing in high-frequency trading algorithms. Their study highlights the use of dynamic feature selection mechanisms and lightweight neural networks, which not only reduce computational complexity but also improve adaptability and efficiency in varying market conditions.

The challenge of risk management in scalping strategies has been explored extensively. Research by Liu et al. [4] emphasizes the importance of leveraging technical indicators, such as the Simple Moving Average (SMA) and Relative Strength Index (RSI), to make informed trading decisions. These indicators allow for the evaluation of market trends and momentum, enabling traders to identify optimal entry and exit points. However, the literature also notes the limitations of traditional indicators in highly volatile markets. Abdelatif Hafid et al. [5] address these limitations by presenting a machine learning-based classification model that integrates enhanced technical indicators such as MACD, RSI, and Bollinger Bands. Their study demonstrates a buy/sell signal accuracy of over 92%, providing a significant advantage in decision-making under volatile conditions. Similarly, Anika Tahsin Meem [6] proposes a novel deep learning approach that applies Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), Bidirectional Long Short-Term Memory (BiLSTM), and Gated Recurrent Unit (GRU) architectures to predict market risk (volatility). This model demonstrates improved accuracy and generalization, making it particularly useful for navigating unpredictable market scenarios.

Another critical aspect discussed in the literature is scalability, particularly in systems designed to handle multiple cryptocurrency pairs and large volumes of data. Modular software architectures have been a recurring theme in studies, with researchers advocating for the use of distributed systems to improve scalability. For instance, Jericevic et al. [7] propose a modular approach where separate components handle data collection, analysis, and trade execution, ensuring efficient distribution of computational workloads. The use of cloud-based infrastructure is also frequently cited as a solution for scaling trading systems to accommodate growing market demands.

Reliability is a recurring concern in automated trading systems, as technical disruptions can lead to significant financial losses. Existing research underscores the importance of implementing robust error-handling and recovery mechanisms. For example, the work of Byrd et al. [8] focuses on strategies for ensuring system continuity during API outages or connection losses. Techniques such as retry mechanisms and failover protocols have been highlighted as critical for maintaining uninterrupted operations.

Blockchain metrics have also gained attention as valuable indicators in cryptocurrency trading. Juan C. King et al. [9] propose the use of blockchain-derived metrics such as hash rate and mining difficulty to identify trading signals. These metrics, inspired by the "Hash Ribbon," provide statistically significant advantages in highly volatile markets, further enriching the decision-making process.

The comparison of scalping strategies with other trading approaches, such as trend-following, has also been a focus of academic and industry research. Scalping is often described as more profitable in volatile markets due to its high-frequency nature, as demonstrated by experiments conducted by Sood et al. [10]. However, these studies also acknowledge the trade-offs, noting that scalping requires higher computational resources and carries increased exposure to short-term market risks compared to trend-following strategies. In this context, Shamima Nasrin Tumpa et al. [11] explore Recurrent Neural Networks (RNN) for real-time price prediction and trading strategy optimization, demonstrating their effectiveness in addressing the challenges posed by high market volatility.

Despite the extensive research available, there are notable gaps in the literature. For instance, while many studies address latency and risk management independently, there is limited work on integrated solutions that simultaneously optimize multiple system aspects. Additionally, while the advantages of using specific technical indicators are well-documented, there is a need for comparative studies to evaluate their relative effectiveness in scalping versus other strategies.

This study builds upon the existing body of knowledge by addressing these gaps and proposing a comprehensive solution that integrates real-time data processing, advanced risk management techniques, modular architecture for scalability and reliability, and innovative blockchain metrics. By combining insights from previous research with advanced methodologies, this study aims to advance the field of automated cryptocurrency trading and provide a practical framework for implementing efficient scalping strategies.

The core challenge in implementing scalping strategies lies in addressing latency issues. Scalping requires systems to process market data and execute trades within milliseconds. Delays in data collection, processing, or trade execution can result in significant losses. Furthermore, the high frequency of trades exposes the system to increased risks due to rapid market fluctuations. Effective mechanisms are necessary to mitigate these risks while maintaining the profitability of the strategy.

Another significant challenge is ensuring scalability. Scalping systems must be capable of handling large volumes of market data and supporting multiple cryptocurrency pairs without compromising performance. This necessitates a modular software architecture that can efficiently distribute processing tasks and scale as trading volumes increase. Scalability is critical for maintaining system reliability and ensuring that it can adapt to the growing demands of the cryptocurrency market.

Reliability is also a key consideration. Automatic trading systems must operate continuously, even during technical disruptions such as connection losses or API failures. Ensuring uninterrupted operation requires robust error-handling mechanisms and automatic recovery features. Without these capabilities, the system risks downtime, missed trading opportunities, and potential financial losses.

To address these challenges, this study focuses on developing an automatic trading system optimized for scalping strategies. The proposed system utilizes advanced technologies such as WebSocket for real-time data streaming, Redis for temporary data caching, and PostgreSQL for reliable storage of historical market data. The system integrates technical indicators, including SMA, RSI, and Bollinger Bands [12], to enable informed decision-making and improve trade execution efficiency.

A review of existing literature reveals that while significant advancements have been made in each of these areas, few studies have addressed the need for a comprehensive system that integrates low-latency data processing, advanced risk management techniques, scalability, and reliability in the context of cryptocurrency scalping.

This gap in the literature highlights the need for research dedicated to the development of an integrated automatic cryptocurrency trading system optimized for scalping strategies. This study aims to fill this gap by proposing a system that combines real-time data processing, advanced risk management techniques, and modular scalability to address the unique challenges of high-frequency trading in volatile cryptocurrency markets.

3. The aim and objectives of the study

The cryptocurrency market is rapidly evolving, offering unique opportunities for implementing innovative trading strategies. Scalping, a high-frequency trading method focused on small price fluctuations, is one of the most effective approaches for maximizing profits in volatile markets. However, its implementation poses significant challenges, including latency, risk management, scalability, and reliability, which require advanced automatic trading systems.

The aim of the study is to develop and implement an automatic cryptocurrency trading system optimized for scalping strategies. The proposed system will address key limitations of existing solutions, such as latency and scalability issues, while ensuring robust risk management and reliable operation. Additionally, the study aims to evaluate the performance of the developed system through experimental testing and comparison with alternative trading approaches.

To achieve this aim, the following objectives are set:

1. Develop a novel scalping method that optimizes trade execution, risk management, and scalability in dynamic cryptocurrency market conditions.

2. Implement a software solution integrating the proposed scalping method into an automatic trading system, ensuring real-time data processing and seamless operation within cryptocurrency exchanges.

3. Evaluate the effectiveness of the developed system through experimental testing, comparing its performance with other trading strategies to provide practical recommendations for its application.

4. The study materials and development of a scalping method and software solution

4.1. Development of a scalping method

To better understand the decision-making process, pseudocode was developed to implement the method. Below is an example of pseudocode for implementing a strategy based on Exponential Moving Average (EMA) and Volume Weighted Average Price (VWAP):

```
Initialize parameters: EMA_period, VWAP_period, smoothing_factor, threshold

While market is open:
    Get current price P_t and volume V_t
    Calculate EMA using smoothing_factor and previous EMA value
    Calculate VWAP based on cumulative price and volume

    If (P_t > EMA and VWAP indicates strong buy signal):
        Place Buy Order
    Else If (P_t < EMA and VWAP indicates strong sell signal):
        Place Sell Order
    End If

    Update historical price and volume data
End While
```

When modeling trading strategies, it is crucial to identify and adjust key parameters that influence algorithm efficiency. The number of periods for the EMA determines the indicator's sensitivity to price changes, with higher periods reducing sensitivity and lower periods increasing it. The time period for VWAP calculation affects the accuracy of average price estimation, where shorter periods are typically used in scalping to enhance entry and exit signal precision. The threshold for entering or exiting positions defines the levels at which the bot executes buy or sell decisions, typically fine-tuned through historical data analysis and backtesting. Additionally, the smoothing factor (α) controls how quickly the EMA responds to price changes, enabling the bot to adapt to varying market conditions effectively.

All these parameters require tuning based on backtesting results to ensure optimal bot performance. Selecting the optimal values for each parameter can significantly impact trading efficiency, risk reduction, and profit maximization.

4.2. Development of a scalping method and software solution

The software architecture is based on a modular approach, enabling effective distribution of functionality across various components as shown in Figure 1.

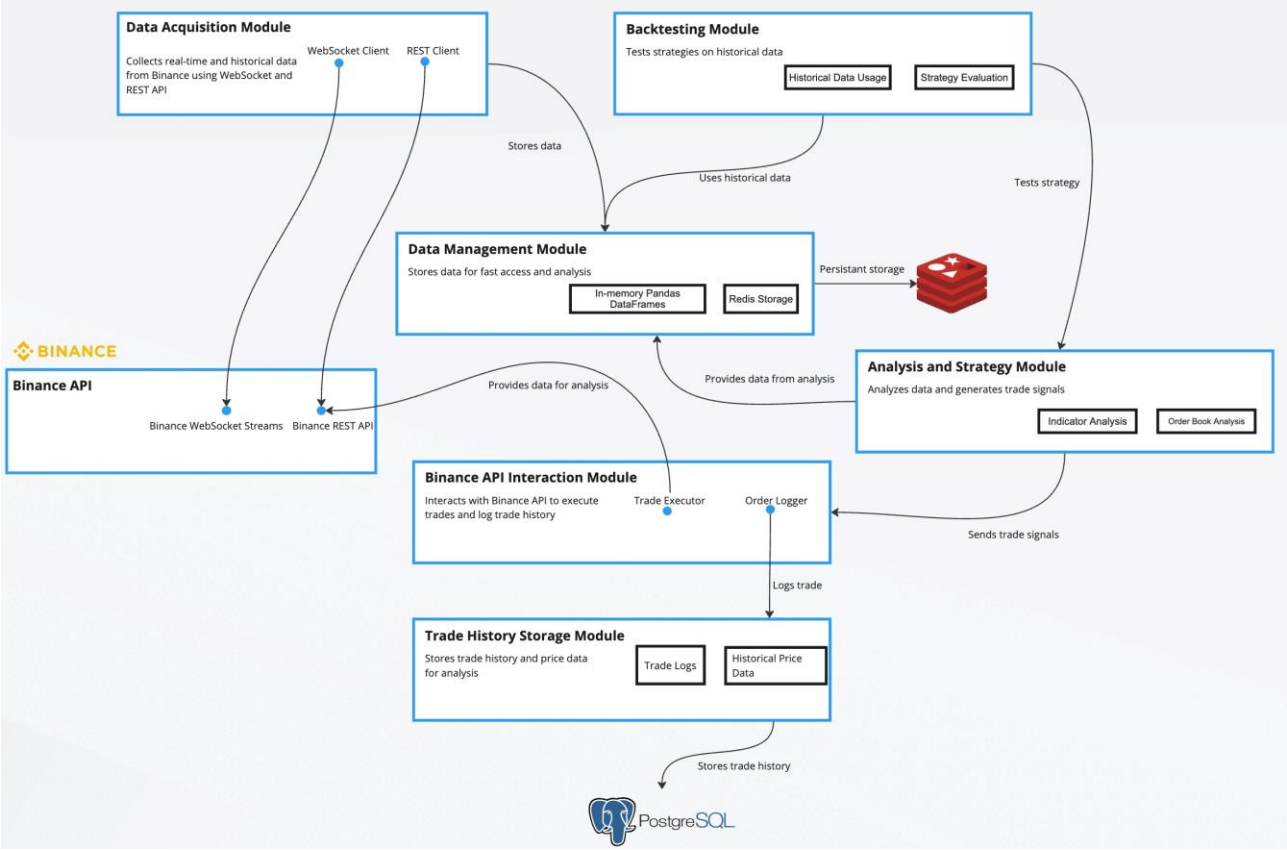

Fig.1. Software Architecture

The main modules of the system include: the Binance interaction module, the data collection module, the data management module, the analysis and strategy module, the trading execution module, and the historical data storage module. Each module performs a specific task, ensuring high flexibility, scalability, and adaptability of the system.

– The Binance interaction module manages integration with the cryptocurrency exchange, providing access to streaming and historical data via WebSocket and REST API.

– The data collection module structures the received information for further use in analytical and trading processes.

– The data management module implements temporary storage via Redis and processes current data using Pandas.

– The analysis and strategy module calculates technical indicators such as SMA, RSI, and Bollinger Bands, making decisions on trading operations.

– The trading execution module interacts with the Binance API to place orders and monitor trades.

– The historical data storage module uses PostgreSQL for reliable and structured data storage.

The bot's lifecycle consists of the stages of initialization, data collection, analysis, strategy execution, and error handling as shown in Figure 2. During the initialization stage, API keys are configured, a WebSocket connection to Binance is established, and necessary components like Redis are prepared. Data collection ensures the retrieval of real-time and historical market data for analysis. Market analysis is performed using indicators, enabling well-reasoned decisions on trade entry or exit.

The strategy execution stage involves interaction with the exchange API for order placement and logging of all operations. In the event of technical failures, such as connection loss, the bot automatically resumes its operation, ensuring system stability.

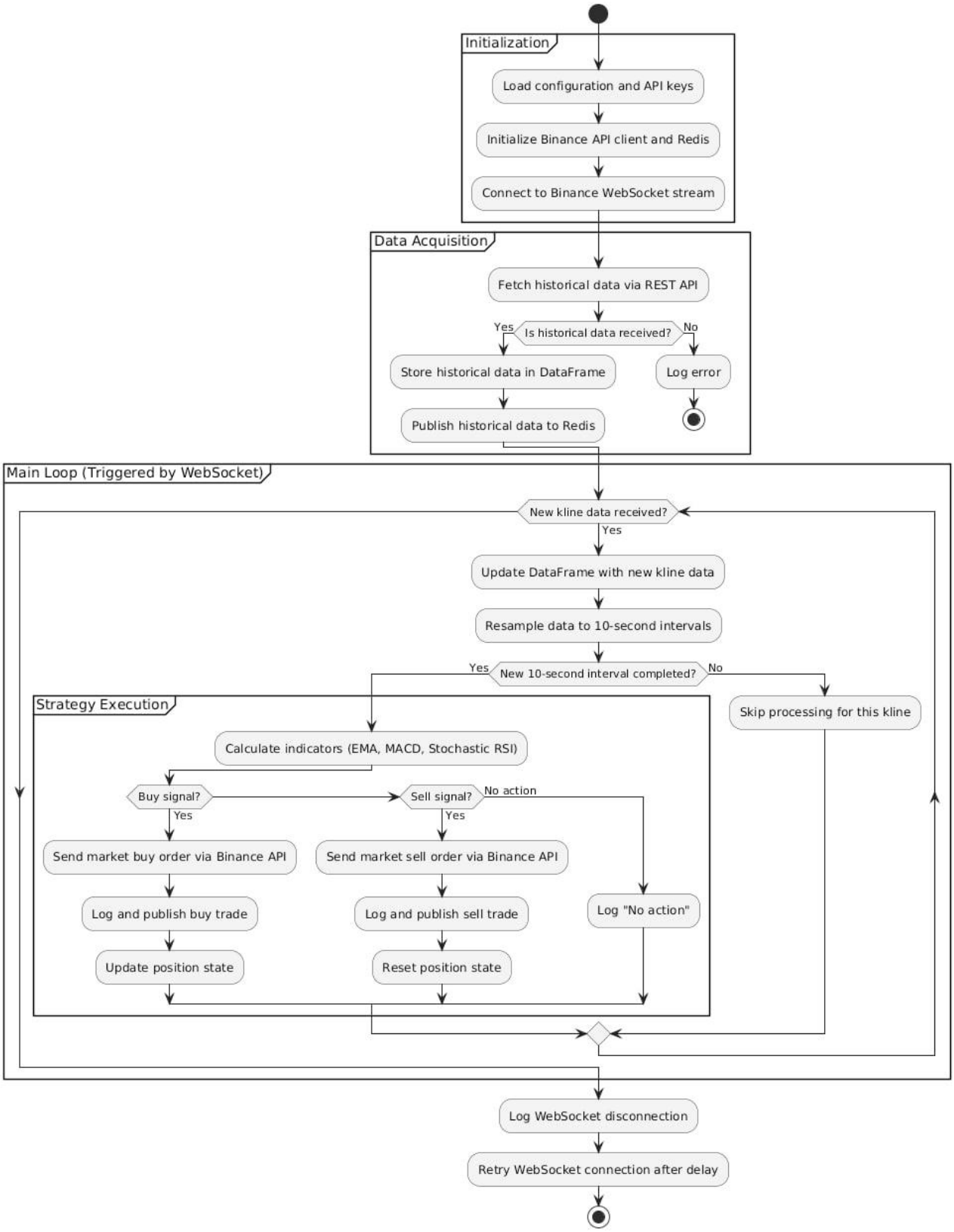

Fig.2. UML Activity Diagram of the Bot's Lifecycle

The technologies employed during development include PostgreSQL for historical data storage, Redis for fast access to temporary data, WebSocket and REST API Binance for exchange integration, and Python libraries (Pandas, NumPy, TA-Lib) ensures a robust and efficient architecture. for market data processing. PostgreSQL provides reliable storage of large volumes of information necessary for strategy analysis and optimization. Redis is used for data caching, significantly improving access speed.

This modular, technologically advanced approach ensures the scalability, reliability, and efficiency of the trading system under real market conditions.

The proposed trading method based on a scalping strategy was implemented through software that enables real-time market data analysis, decision-making regarding trade entry and exit, and automatic trade execution. The bot operates on WebSocket technology, allowing data to be received with minimal delay, which is critical for the successful application of scalping strategies. These strategies rely on frequent trades with small profits, and their success depends on rapid responses to market fluctuations.

5. Results of investigating the proposed method and software – optimized Scalping method using EMA and VWAP Indicators

The experiment was conducted using the Bitcoin to Tether trading pair (BTC/USDT) cryptocurrency pair, selected for its high liquidity, popularity, and significant volatility, which created favorable conditions for testing the scalping bot. The collected data was analyzed using key indicators such as the SMA, RSI, and Bollinger Bands. This ensured comprehensive market condition analysis and allowed well-reasoned decisions regarding trade entry and exit.

The experiments were executed on a computer system with the following specifications:

Processor: Intel Core i7-10700K (8 cores, 16 threads, 3.8 GHz base clock, 5.1 GHz turbo boost).

– Memory: 32 GB DDR4 RAM (3200 MHz).
– Storage: 1 TB NVMe SSD (read/write speed: 3500/3000 MB/s).
– Operating System: Ubuntu 22.04 LTS.
– Network: Wired Ethernet connection with a speed of 1 Gbps.

This configuration ensured the system's ability to handle high-frequency trading operations, process large volumes of market data, and execute trades with minimal latency.

To assess the effectiveness of the scalping method, it was compared with a trend-following strategy. The trend bot focused on long-term trades and used indicators like SMA and Moving Average Convergence Divergence (MACD). Unlike the scalping bot, the trend bot conducted fewer trades, focusing on stable market trends. Experiments showed that the scalping bot achieved higher overall profits due to more frequent trades, while the trend bot provided more stable profits per trade under steady market conditions as shown in Table 1.

Table 1. Comparison of Scalping and Trend Bot Performance

Parameter	Scalping Bot	Trend Bot
Test Duration	2 hours	2 hours
Number of Trades	15	7
Number of Successful Trades	13	5
Total Profit (USDT)	120	85
Average Profit (%)	1.2%	0.85%
Maximum Profit (USDT)	25	40
Minimum Profit (USDT)	5	10
Maximum Loss (USDT)	–8	–12
Position Holding Time (min)	4	20

The scalping bot’s higher trade frequency allowed it to capitalize on rapid price fluctuations, demonstrating its strength in volatile markets. Its ability to adapt quickly to changing conditions was evident in its low average position holding time and high success rate. The trend bot, on the other hand, was more conservative, aiming for fewer but more significant trades, which is reflected in its higher maximum profit per trade.

An evaluation of memory usage during the experiment showed that the scalping bot's resource requirements scaled linearly with the number of cryptocurrency pairs processed. This was an essential observation as it underlined the system's scalability. For instance, the bot consumed 120 MB of memory for a single pair and scaled up to 2100 MB when handling 20 pairs, as shown in Table 2.

Table 2. Memory Usage During Scalping Bot Operation

Number of Pairs	Memory During Peak Load (MB)	Base Memory Usage (MB)	Cache Memory Usage (MB)	Total Memory (MB)
1	100	80	20	120
5	450	350	100	650
10	900	700	200	1300
15	1400	1100	300	1700
20	1900	1400	500	2100

For a single cryptocurrency pair, the scalping bot requires a total of 120 MB of memory. This includes 80 MB allocated to core operations, such as maintaining the bot's functionality and processing real-time data streams, and 20 MB for cache memory, which is utilized for storing frequently accessed data temporarily. The efficient allocation of memory resources ensures the bot operates smoothly without unnecessary overhead, making it well-suited for low-volume trading scenarios where only a single pair is analyzed.

As the number of cryptocurrency pairs being processed increases, the bot's memory consumption grows linearly. This consistent scaling demonstrates the modular design of the system, allowing for predictable increases in resource usage with each additional pair. For example, when managing 5 pairs, the total memory usage rises to 650 MB, comprising 350 MB for base memory and 100 MB for cache. At 10 pairs, memory consumption doubles to 1300 MB, with 700 MB dedicated to base operations and 200 MB allocated for cache. At its maximum tested capacity of 20 pairs, the bot requires 2100 MB of memory, including 1400 MB for base functions and 500 MB for cache. This linear growth pattern highlights the bot's ability to handle increasingly complex workloads while maintaining stable performance.

The predictable nature of memory consumption is a key factor in the bot's scalability. Linear scaling allows traders to effectively plan their system resources based on expected trading volumes, ensuring smooth operations across varying scales. Whether used for single-pair trading or high-volume scenarios involving multiple pairs, the bot demonstrates an ability to adapt without significant performance degradation.

To further evaluate the scalability of the bot, several key performance metrics were measured, including latency, average signal processing time, Computational tasks were executed using a Central Processing Unit (CPU) usage, and memory consumption. Latency measures the delay between receiving a market signal and the bot's response, a critical metric for scalping strategies that rely on rapid execution to capture small price fluctuations. Average Signal Processing Time reflects the bot's efficiency in analyzing incoming data and making trading decisions, ensuring responsiveness to dynamic market conditions.

CPU usage was another important factor, as it indicates the computational load on the processor during operation. Efficient CPU utilization ensures the bot can maintain high performance without overburdening the system, even when processing multiple pairs. Finally, memory consumption provides insight into the bot's resource requirements, highlighting its capacity to scale smoothly as trading activity intensifies.

These metrics, as summarized in Table 3, underscore the bot's robust architecture and its ability to maintain performance under varying workloads. By effectively managing memory, processing time, and computational resources, the scalping bot proves to be a scalable and reliable tool for

traders, whether operating in low or high-volume trading environments. This adaptability ensures that the bot is suitable for real-world applications, providing both efficiency and stability in volatile cryptocurrency markets.

Table 3. Scalability Metrics of the Scalping Bot

Number of Pairs	**Latency (ms)**	**Average Signal Processing Time (ms)**	**CPU Usage (%)**	**Memory (MB)**
1	15	30	5	120
5	20	40	15	650
10	25	50	25	1300
15	35	60	40	1700
20	50	75	55	2100

The scalability of the scalping bot was evaluated using key performance metrics, including latency, average signal processing time, CPU usage, and memory consumption. These metrics, detailed in Table 3, were measured for varying loads, starting from a single cryptocurrency pair up to a maximum of 20 pairs. This assessment provided valuable insights into the bot's ability to maintain stable performance as the operational load increased.

Latency, a critical metric for scalping strategies, measures the time delay between receiving a market signal and the bot's response. For a single pair, latency was recorded at 15 ms. As the number of pairs increased, latency grew steadily, reaching 50 ms when processing 20 pairs. Despite the incremental increase, latency remained within acceptable limits, ensuring the bot could respond quickly to market changes—a necessity for high-frequency trading.

Average signal processing time reflects the time taken to analyze incoming market data and make trading decisions. For a single pair, the bot required 30 ms, and this time increased proportionally to 75 ms when handling 20 pairs. The gradual rise in processing time demonstrates efficient computational resource allocation, enabling the bot to handle complex data streams without significant delays. This consistent performance underscores the bot’s ability to operate effectively in fast-paced trading environments.

CPU usage was also analyzed to understand the bot’s computational demands. At minimal load, the bot used only 5% of the CPU, demonstrating its optimized design for low-volume trading. As the number of pairs increased, CPU usage rose proportionally, peaking at 55% for 20 pairs. This efficient scaling ensured that the bot remained within safe operational thresholds, avoiding system overload while maintaining stable performance.

Memory consumption increased linearly with the number of pairs, as previously noted, growing from 120 MB for a single pair to 2100 MB for 20 pairs. This predictable scaling of memory usage highlights the bot's modular architecture, allowing for effective resource management and planning as trading volumes expand. The bot’s ability to maintain linear growth in resource consumption further validates its scalability.

To analyze real-time performance, Table 4 breaks down response times into signal receiving, data processing, and order placement. For a single pair, the total response time was 50 ms, divided into 10 ms for signal receiving, 20 ms for processing, and 20 ms for order placement. At maximum capacity with 20 pairs, the total response time increased to 105 ms, with 20 ms for signal receiving, 50 ms for processing, and 35 ms for order placement. These results indicate that the bot maintained rapid response times even under maximum load, ensuring minimal delays in trade execution.

In summary, the scalping bot demonstrated strong scalability and responsiveness, maintaining efficient performance across varying levels of trading activity. The latency, processing time, CPU usage, and memory consumption metrics all showed proportional scaling, ensuring the bot’s adaptability to different trading demands. This robust performance makes the bot a reliable tool for

traders, capable of handling both small-scale and large-scale operations in volatile cryptocurrency markets.

Table 4. Response Time for Scalping Bot Operations

Number of Pairs	Signal Receiving Time (ms)	Processing Time (ms)	Order Placement Time (ms)	Total Response Time (ms)
1	10	20	20	50
5	12	28	20	60
10	15	35	25	75
15	18	40	30	88
20	20	50	35	105

Overall data processing speed remains stable even under maximum load conditions.

Thus, the scalping bot demonstrates high efficiency and stability, even under significant loads. This makes it a versatile tool suitable for real-world market conditions.

6. Analysis of the obtained results

The results of the study provide strong evidence of the scalability, efficiency, and adaptability of the scalping bot in handling high-frequency cryptocurrency trading. These findings are explained by the system's modular architecture and its ability to efficiently allocate computational resources, even under significant workloads. The analysis of performance metrics, such as memory consumption, CPU usage, latency, and response times, highlights the bot's ability to maintain stable operation and reliable performance across various trading volumes.

One of the critical findings is the bot's scalability, which is explained by the linear nature of resource consumption observed during the experiments. The memory usage, for example, increased predictably from 120 MB for a single cryptocurrency pair to 2100 MB for 20 pairs. This predictable scaling underscores the effectiveness of the bot’s design in managing growing workloads without overwhelming system resources. Similarly, CPU usage remained proportional to the number of pairs processed, peaking at 55% under maximum load. This efficient resource utilization ensures that the bot can operate smoothly even during high trading volumes, making it highly suitable for real-world applications.

The latency and response time metrics further validate the bot's adaptability to dynamic market conditions. The experiments demonstrated that latency increased from 15 ms for a single pair to 50 ms for 20 pairs, while the total response time remained under 105 ms at maximum load. These results suggest that the bot can execute trades quickly, an essential feature for scalping strategies that rely on capturing small price fluctuations within volatile markets. These outcomes are attributed to the bot’s streamlined decision-making processes and its use of real-time data streams.

While the scalping bot's rapid trade execution resulted in higher overall profitability compared to a trend-following strategy, this strength also introduced higher risks. The increased trade frequency amplified the overall risk exposure, despite the lower maximum loss per trade for the scalping bot (–8 USDT) compared to the trend bot (–12 USDT). This trade-off between profitability and risk underscores the importance of robust risk management mechanisms in automated trading systems. Future improvements could include the integration of machine learning models to dynamically adjust risk parameters based on market conditions.

The study also revealed potential limitations of the scalping bot. The reliance on linear resource scaling, while effective for moderate workloads, may pose challenges when operating under extreme conditions or with limited hardware capacity. Optimizing algorithms to reduce memory and CPU demands would enable the bot to process more pairs or operate on less powerful hardware. Additionally, the experiments focused primarily on the BTC/USDT pair, which, while highly liquid

and volatile, may not represent the bot's performance across less active trading pairs. Expanding the scope of testing to include a broader range of cryptocurrencies and market scenarios would provide a more comprehensive understanding of the bot's capabilities.

The bot's reliability is another key aspect highlighted by the results. Error-handling mechanisms ensured continuous operation, even during peak loads, preventing disruptions that could have resulted in missed trading opportunities. However, the system's dependence on stable network connections and API availability introduces potential vulnerabilities. Future iterations could incorporate redundancy features, such as fallback mechanisms or integration with multiple exchanges, to enhance the system's robustness in real-world trading environments.

In conclusion, the findings of this study demonstrate the practicality and reliability of the scalping bot for professional trading applications. Its ability to handle multiple cryptocurrency pairs, maintain rapid response times, and scale efficiently makes it a valuable tool for institutional investors and high-frequency traders. Future research should focus on optimizing resource utilization, integrating advanced risk management techniques, and testing the bot in live trading scenarios to further validate its performance and identify additional areas for improvement. These enhancements would ensure that the scalping bot remains competitive in the fast-evolving landscape of cryptocurrency trading.

Conclusions

The study successfully developed an innovative scalping strategy optimized for trade execution, risk management, and scalability in dynamic cryptocurrency market conditions. The implemented automated trading system demonstrated the ability to process real-time market data with minimal latency, ranging from 15 ms to 50 ms, enabling rapid decision-making and high-frequency trading. The system's modular design ensured linear scalability of resource usage, with memory consumption increasing predictably from 120 MB for a single pair to 2100 MB when handling 20 pairs. This confirmed the system's capability to maintain stable performance under varying workloads.

Experimental testing revealed the advantages of the scalping strategy compared to a trend-following approach. Over a two-hour test period, the scalping bot achieved higher overall profitability, conducting 15 trades (13 successful) with a total profit of 120 USDT. In contrast, the trend bot, focused on fewer trades, generated a profit of 85 USDT. The scalping bot's higher trade frequency allowed it to capitalize on short-term price fluctuations, while its low average position holding time of 4 minutes highlighted its adaptability to volatile market conditions. However, the trend bot achieved a higher maximum profit per trade under steady market conditions, demonstrating its more conservative approach.

The scalability of the system was validated through key performance metrics, including memory usage, latency, CPU utilization, and signal processing times. Results showed predictable and proportional increases in resource usage as the number of cryptocurrency pairs increased, ensuring stable performance without significant degradation. The system maintained acceptable response times, growing from 50 ms for a single pair to 105 ms for 20 pairs, while CPU usage scaled efficiently from 5% to 55%. These findings highlight the bot's ability to handle both small-scale and high-volume trading operations effectively.

The practical value of the developed system lies in its efficiency, reliability, and scalability, making it a versatile tool for high-frequency traders operating in volatile markets. Its modular architecture allows for smooth integration with real-world trading platforms, while its ability to adapt to dynamic market conditions ensures robust performance. The experimental results confirm that the system provides a competitive advantage in cryptocurrency trading, offering both high profitability and resource efficiency.

References

[1] B. Miles and D. Cliff, "A cloud-native globally distributed financial exchange simulator for studying real-world trading-latency issues at planetary scale," *arXiv preprint*, arXiv:1909.12926, 2019. [Online]. Available: https://doi.org/10.48550/arXiv.1909.12926

[2] Y. Fan, Z. Hu, L. Fu, Y. Cheng, L. Wang, and Y. Wang, "Optimizing real-time data processing in high-frequency trading algorithms using machine learning," *arXiv preprint*, arXiv:2412.01062, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2412.01062

[3] X. Liu *et al.*, "FinRL: A deep reinforcement learning library for automated trading in quantitative finance," *arXiv preprint*, arXiv:2011.09607, 2020. [Online]. Available: https://doi.org/10.48550/arXiv.2011.09607

[4] A. Hafid, M. Rahouti, L. Kong, M. Ebrahim, and M. A. Serhani, "Predicting Bitcoin market trends with enhanced technical indicator integration and classification models," *arXiv preprint*, arXiv:2410.06935, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2410.06935

[5] A. T. Meem, "A deep learning approach to predict the fall of cryptocurrency long before its actual fall," *arXiv preprint*, arXiv:2411.13615, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2411.13615

[6] I. Jericevic, D. Sing, and T. Gebbie, "CoinTossX: An open-source low-latency high-throughput matching engine," *arXiv preprint*, arXiv:2102.10925, 2021. [Online]. Available: https://doi.org/10.48550/arXiv.2102.10925

[7] D. Byrd *et al.*, "The importance of low latency to order book imbalance trading strategies," *arXiv preprint*, arXiv:2006.08682, 2020. [Online]. Available: https://doi.org/10.48550/arXiv.2006.08682

[8] J. C. King, R. Dale, and J. M. Amigó, "Blockchain metrics and indicators in cryptocurrency trading," *arXiv preprint*, arXiv:2403.00770, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2403.00770

[9] S. Sood *et al.*, "Evaluation of reinforcement learning techniques for trading on a simulated stock market," *arXiv preprint*, arXiv:2309.03202, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2309.03202

[10] S. N. Tumpa and K. D. G. Maduranga, "Utilizing RNN for real-time cryptocurrency price prediction and trading strategy optimization," *arXiv preprint*, arXiv:2411.05829, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2411.05829

[11] J. Bollinger, "Bollinger on Bollinger bands," McGraw Hill Professional, New York, 227 p., 2002.

УДК 004.94

АВТОМАТИЧНА СИСТЕМА ТОРГІВЛІ КРИПТОВАЛЮТОЮ З ВИКОРИСТАННЯМ СКАЛЬПІНГОВОЇ СТРАТЕГІЇ

Бераудо Еліза
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
https://orcid.org/0000-0001-7550-3620

Олійник Юрій
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
https://orcid.org/ 0000-0002-7408-4927

У дослідженні розглянуто питання розробки та впровадження автоматизованої системи для скальпінгових стратегій на криптовалютному ринку. Скальпінг є високочастотною стратегією, що орієнтована на отримання прибутку від незначних цінових коливань. Основна мета роботи – створення автоматизованого торгового бота, що вирішує ключові проблеми, пов'язані з латентністю, управлінням ризиками, масштабованістю та надійністю у реальних ринкових умовах. Для досягнення цієї мети було сформульовано завдання: розробити новий метод скальпінгу, реалізувати програмне рішення для його інтеграції у систему автоматизованої торгівлі та оцінити ефективність роботи системи за допомогою експериментального тестування.

Методологія дослідження базується на використанні технічних індикаторів, таких як експоненційне ковзне середнє (*Exponential Moving Average, EMA*) та об'ємнозважена середня ціна (*Volume Weighted Average Price, VWAP*). Псевдокод методики було розроблено для кращого розуміння процесу прийняття рішень, включаючи параметри згладжування, часові періоди та порогові значення для входу та виходу з позицій. Реалізована архітектура системи включає такі модулі: модуль інтеграції з біржею *Binance*, модуль збору та управління даними, модуль аналізу та стратегії, модуль виконання торгових операцій і зберігання історичних даних. Для розробки системи використовувалися технології *PostgreSQL, Redis, WebSocket* та бібліотеки *Python* (*Pandas, NumPy, TA-Lib*).

Експерименти проводилися на парі *BTC/USDT*, яка характеризується високою ліквідністю та волатильністю. Система тестувалася на обладнанні з процесором *Intel Core i*7-10700 K, 32 ГБ оперативної пам'яті та підключенням до мережі зі швидкістю 1 Гбіт/с. Порівняльний аналіз скальпінгової стратегії та трендової стратегії показав перевагу першої в умовах високої волатильності ринку: бот виконав 15 угод (13 успішних) із загальним прибутком 120 *USDT* протягом двох годин.

Було оцінено ключові показники продуктивності, такі як затримка (15–50 мс), час обробки сигналів, завантаженість *CPU* (5–55%) та використання пам'яті (120–2100 МБ). Лінійне масштабування ресурсів підтвердило модульність і гнучкість архітектури системи, що дозволяє ефективно керувати збільшенням обсягів торгівлі.

Отримані результати свідчать про ефективність запропонованого методу та надійність розробленої системи у реальних умовах. Подальші дослідження можуть бути спрямовані на оптимізацію алгоритмів для зменшення споживання ресурсів та інтеграцію передових методів управління ризиками.

Ключові слова: автоматична торгівля, скальпінг, криптовалюта, *Binance API*, алгоритмічна торгівля.