Skoltech

Skolkovo Institute of Science and Technology

MASTER'S THESIS

Portfolio Sold-Out Problem in Numbers for DeFi Lending Protocols

Master's Educational Program: Data Science

Student: ______________ Waralak Pariwatphan
signature

Research Advisor: ______________ Yury Yanovich
signature
PhD, Senior Research Scientist

Moscow 2023

Skoltech

Skolkovo Institute of Science and Technology

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Задача распродажи портфеля в цифрах для протоколов кредитования DeFi

Магистерская образовательная программа: Науки о данных

Студент: ____________ Варалак Париватфан
подпись

Научный руководитель: ____________ Юрий Александрович Янович
подпись
к.ф.-м.-н., старший научный сотрудник

Москва 2023

Portfolio Sold-Out Problem in Numbers for DeFi Lending Protocols

Waralak Pariwatphan

Submitted to the Skolkovo Institute of Science and Technology on October 11, 2023

ABSTRACT

Portfolio optimization, as per Markowitz's principles, is a well-established concept in quantitative finance. Its primary goal is to diversify financial assets to minimize portfolio risk and maximize efficiency. However, in traditional investments within secondary markets, regulatory restrictions often prevent asset splitting due to a lack of transparency. Blockchain technology presents a promising solution to this challenge through asset tokenization and package construction. The overarching objective is to maximize the creation of asset packages while adhering to prescribed risk constraints. This thesis centers on addressing the portfolio sold-out problem within decentralized finance (DeFi) lending protocols, a complex issue categorized into six distinct cases defined by unique covariance matrices and package characteristics. Our focus lies in establishing theorems for solving four of these cases, specifically the continuous general, continuous independent, discrete general, and discrete independent scenarios, which have remained unsolved to date. Additionally, we aim to propose practical solutions applied to real loan financial data sourced from MakerDAO, a well-recognized DeFi lending protocol. Our findings demonstrate that both the continuous general and continuous independent cases can be effectively solved using the numerical method known as Second-Order Cone Programming (SOCP). SOCP emerges as a reliable and precise technique, consistently delivering optimal numerical solutions. However, when dealing with discrete general cases, it is crucial to acknowledge their inherent complexity, notably their NP-hard nature. Despite these challenges, we observe a significant pattern related to the number of chosen assets and their impact on the tokenized fraction. Specifically, in scenarios with a relatively small number of selected assets, the tokenized fraction tends to be larger compared to cases with a higher count of chosen assets. Furthermore, we uncover a noteworthy correlation between the Gini coefficient and the tokenized fraction, particularly in situations with a limited number of selected assets. As a result, this research addresses the complex portfolio sold-out problem within DeFi lending protocols. By leveraging blockchain technology and innovative mathematical approaches, we contribute solutions to hitherto unsolved cases, enhancing our understanding of portfolio optimization in the context of real-world financial data. These findings offer valuable insights for DeFi and broader financial applications, though it is essential to recognize the inherent complexities of discrete cases and the limitations of real-world data applicability.

Keywords: portfolio, optimization, tokenization, blockchain, DeFi

Research advisor:
Name: Yury Yanovich
Degree, title: PhD, Senior Research Scientist

Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Motivation

Blockchain is a decentralized database with built-in auditability and a tamper-resistant log [49]. It happened to reduce distrust in various domains, such as government, financial institutions, and commercial sections. On the blockchain, no data or activity can be manipulated or erased [57]. The data in the blockchain is kept as transactions, which are then grouped into blocks and linked together using a chain. In the realm of blockchain technology, the utilization of smart contracts is imperative. Smart contracts represent computer technology designed to digitally facilitate, verify, or enforce the negotiation or performance of a contract on the blockchain [9].

Decentralized Finance (DeFi) has developed as a financial technology based on blockchain that is altering existing financial practices by removing intermediaries and centralized financial institutions from transactions [19]. A prominent example of DeFi innovation is MakerDao, a leading lending platform that empowers users to use their reliable cryptocurrencies as collateral to create the Dai stablecoin [27]. MakerDao plays a pivotal role in the DeFi ecosystem. To maintain the platform's stability, borrowers must maintain a collateral ratio to prevent liquidation. In the case of liquidation, the lending platform holds asset auctions to recover debts, as well as the system's stability and penalty fees. All transactions related to DeFi lending protocols are meticulously recorded on the Ethereum blockchain [3], ensuring the authenticity and integrity of financial activities. This real-time loan data is accessible to the public, promoting transparency in the ecosystem.

In the process of securitization, banks and financial institutions possess the capability to convert a pool of non-marketable assets or the expected future cash flows generated by these assets, such as loans, into securities that can be readily traded in the market. These securities can then be offered in the capital market, thereby generating capital for business operations [62]. The process of portfolio optimization is crucial for preparing these securities for sale. Portfolio optimization, drawing from the principles outlined by Markowitz, is a well-established concept in quantitative finance. Its primary objective is to effectively diversify financial assets to minimize portfolio risk and maximize overall efficiency [37].

However, in traditional investments, regulatory constraints often discourage the practice of asset splitting during the securitization process due to a lack of audibility, constraining small private investors from engaging in some opportunities due to the massive quantity of invested assets. With its essential properties of decentralization, scalability, and security, blockchain technology provides a solution to this challenge. Notably, blockchain enables the distribution of digital ledgers without depending on a central entity to maintain their integrity [76]. Tokenization, which allows for the fractionation of assets into tokens and the transformation of ownership rights into digital tokens, is an essential tool that facilitates this. As a result, blockchain has the ability to make it accessible while decreasing the investment gap for small individual investors.

This research project aims to address the current deficiencies in the securitization process, which is often hindered by a lack of transparency, particularly in asset splitting. Blockchain technology offers a solution to this challenge, creating a compelling incentive for further exploration in this domain. To achieve this goal, it is essential to establish a solid foundation of knowledge for developing optimized portfolio strategies in the realm of decentralized finance (DeFi), leveraging real loan data obtained from DeFi lending protocols. The integration of blockchain technology into the world of digital assets represents a revolutionary step, enabling the seamless fusion of DeFi with traditional finance.

This study contributes to the ever-expanding body of knowledge within decentralized finance by conducting a comprehensive examination of DeFi lending protocols and their untapped potential. The ultimate goal is to foster a more inclusive and efficient financial environment. Additionally, it is worth noting that the principles and techniques of DeFi explored in this research have the potential to benefit classical financial institutions, such as the banking system, by providing them with valuable competitive tools. As we continue to navigate the evolving landscape of finance, the insights gained from this study may well shape the future of both DeFi and traditional financial systems.

1.2 Problem Definition

In the context of the motivation outlined earlier, the securitization process involves a series of fundamental steps. These steps entail the dissection of assets into smaller components, followed by their reassembly into a consolidated package for subsequent sale. We can draw parallels to these steps in the realm of tokenization, as illustrated in Figure 1.1.

Tokenization essentially converts assets into a sequence of digital tokens, each symbolizing ownership in a digital format. These tokens can then be organized into sets or packages. The problem lies in efficiently creating as many packages as possible from a given set of tokens while ensuring that each package carries a risk level below a specified threshold. The overarching goal of asset tokenization and package construction is to maximize the creation of packages within the prescribed risk constraints. This objective is driven by the desire to minimize the number of remaining tokens, as a surplus of tokens can lead to increased maintenance costs. This intricate procedure can be effectively addressed through the resolution of what is known as the portfolio sold-out problem [25].

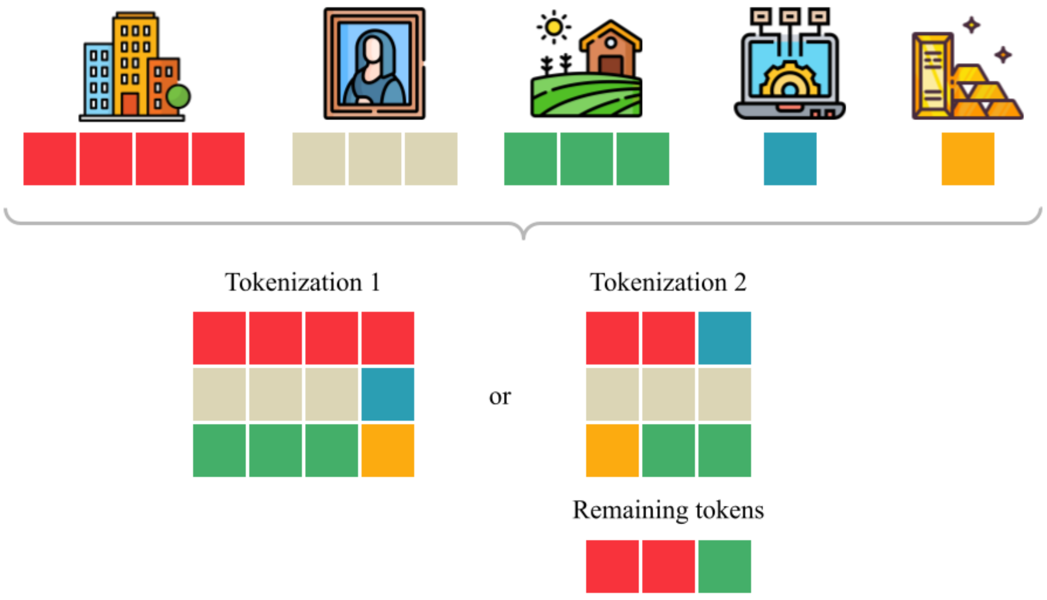

Figure 1.1: Asset tokenization and package construction

1.3 Purpose of the Research

This thesis focuses on solving the portfolio sold-out problem within DeFi lending protocols. The problem can be categorized into six distinct cases, as outlined in Table 3.1. Two of these cases, namely the discrete homogeneous and continuous homogeneous cases, have already been addressed [25, 23]. These solutions are based on the assumption of uniformity in asset variations.

This research objective turns to addressing the remaining four unsolved cases, each characterized by unique covariance matrices and package characteristics:

1. **Discrete Independent Case**: Zero correlations between asset returns and Boolean package matrix $\mathbf{D}_M$ for discrete packaging.

2. **Continuous Independent Case**: Similar to the discrete independent case, but with real-valued $\mathbf{C}_M$ for continuous packaging.

3. **Discrete General Case**: No restrictions on covariance while still utilizing a Boolean $\mathbf{D}_M$ for discrete packaging.

4. **Continuous General Case**: It resembles the discrete general case but utilizes real-valued $\mathbf{C}_M$ for continuous packaging.

The overarching aim is to propose solutions that align more effectively with the intricacies of real-world financial data, tackling these more intricate and realistic challenges.

Following the formulation of the portfolio sold-out problem, the research will pivot to a practical application phase. Real loan data sourced from the MakerDAO system will be harnessed to assess the tangible efficiency gains achieved through this framework. Specifically, we will evaluate the efficiency by examining the tokenized fraction, which serves as a key metric. This fraction signifies the total value of assets successfully integrated into the packages relative to the initial pool of assets available for tokenization.

This empirical phase not only validates the theoretical underpinnings but also provides invaluable insights into the real-world impact of our portfolio construction approach, shedding light on its potential to maximize asset utilization in practical financial scenarios.

1.4 Scientific Novelty

Presently, the predominant share of research efforts is concentrated on portfolio optimization within traditional primary markets, encompassing assets like stocks and bonds. However, it is imperative to recognize that this focus may not be directly applicable to the dynamics of the secondary market, particularly when it comes to the tokenization of loans and their subsequent composition into saleable packages.

Remarkably, there exists a conspicuous void in the realm of research dedicated to portfolio optimization tailored specifically for the secondary market. This void underscores the critical need for innovative approaches and methodologies that can accommodate the distinct intricacies and characteristics of secondary market assets, such as tokenized loans.

This research endeavors to leverage the transformative potential of blockchain technology in the tokenization of bank loans and the subsequent assembly of these tokens into tradable commodities within the secondary market. This pioneering approach holds the promise of not only addressing governmental concerns regarding bank auditability but also ushering in a new era of financial tools and opportunities.

One of the profound implications of this research lies in its capacity to democratize access to the secondary market, empowering small investors with novel competitive tools. By bridging the gap between traditional banking and decentralized finance, this innovative approach can potentially unlock a wealth of opportunities for a broader spectrum of investors, fostering a more inclusive and dynamic secondary market ecosystem.

1.5 Organization of the Thesis

The rest of the thesis is organized as follows: The next chapter presents a thorough literature study covering the most recent breakthroughs in blockchain technology and smart contracts, which are the fundamental innovations of DeFi. It also dives into current solutions to the portfolio sold-out problem in DeFi lending protocols. The problem statement was developed by identifying research gaps related to real loan datasets in the DeFi lending protocols.

The data-gathering procedure and data preparation techniques are explained in Chapter 3. Given that the raw data from smart contracts is encoded using hash functions and lacks human-readable formatting, a thorough data preparation stage is essential. Furthermore, this chapter elaborates on the derivation of critical economic parameters, taking into account relevant assumptions and theoretical foundations.

The next chapter, Chapter 4, is devoted to providing the empirical results of the research that was undertaken. These findings give information on the approaches' effectiveness in resolving the portfolio sold-out problem within DeFi lending protocols.

Finally, the last chapter summarizes the thesis's essential findings and suggests prospective areas for future extensions of this research, providing a comprehensive conclusion to the research.

1.6 Main Contributions

1. Formulated and demonstrated Theorem 2, providing an avenue for achieving optimal numerical solutions.
2. Formulated and demonstrated Theorem 3, which characterizes an NP-hard problem.
3. Implemented tokenization algorithms on a real-world loan dataset.
4. Proved that market concentration serves as a reliable metric for assessing tokenization quality.

Chapter 2
Literature Review

This chapter provides a comprehensive overview of the key areas relevant to this study. It covers blockchain and smart contracts, decentralized finance (DeFi), lending protocols, and the MakerDAO platform. It also discusses the Markowitz model for portfolio optimization, existing solutions for portfolio sold-out problems, optimization methods used in this thesis, and related works in second-order cone programming. Additionally, it explores the subset sum problem and provides explanations for market concentration and Gini coefficient formulations.

2.1 Blockchain and Smart Contracts

A blockchain is a decentralized database that keeps track of all transactions that occur on it. All transactions cannot be deleted or manipulated. The transactions are grouped into blocks, and each block contains the data from the preceding block. Each block is linked by a chain. The benefit of a blockchain is that it enables non-trusting people to securely communicate and trade assets without the need for a trusted third party. As a consequence, both integrity and double-spending issues are alleviated [12].

Cryptographic methods such as digital signatures and hash functions are essential in blockchain systems. They provide security, integrity, authenticity, and non-repudiation for recorded transactions, preserving the reliability of the ledger [35]. Additionally, due to its distributed nature, blockchain relies on a consensus protocol—a set of rules binding all participants. This ensures they collectively establish a consistent and unified perspective on the blockchain's records.

Blockchain fosters trust between untrusted parties, enabling secure transactions and records. By leveraging cryptography and collaboration, it removes the requirement for intermediaries, thus eradicating the need for centralized institutions. Data is securely stored on the blockchain's ledger through cryptographic methods. Blockchain employs several cryptographic building blocks, as follows:

- **Public key cryptography.** Blockchain relies on cryptographic techniques like public key cryptography, wherein securely stored private keys within digital wallets (hardware or software) are utilized to validate transaction origins [46]. Through private keys, users create digital signatures that validate transaction authenticity, while public keys verify their legitimacy. For example, a user hashes transaction data using their private key to generate a signature, which miners on the blockchain network decrypt using the corresponding public key for verification, as illustrated in Figure (2.1). This process ensures transaction authorship. Notably, both Ethereum and Hyperledger Fabric employ digital signatures, employing the Elliptic Curve Digital Signature Algorithm (ECDSA) to produce key pairs [55, 15, 28]. Public keys serve as user identities while preserving privacy within the blockchain ecosystem.

- **Zero-knowledge proofs.** Zero-knowledge proofs have a key role in blockchain, as exemplified by verifying transactions without needing full information [67, 58]. In a money transfer scenario, the blockchain ensures the sender has enough funds without knowing their identity or total balance. This enhances user privacy. While Ethereum lacks zero-knowledge proof

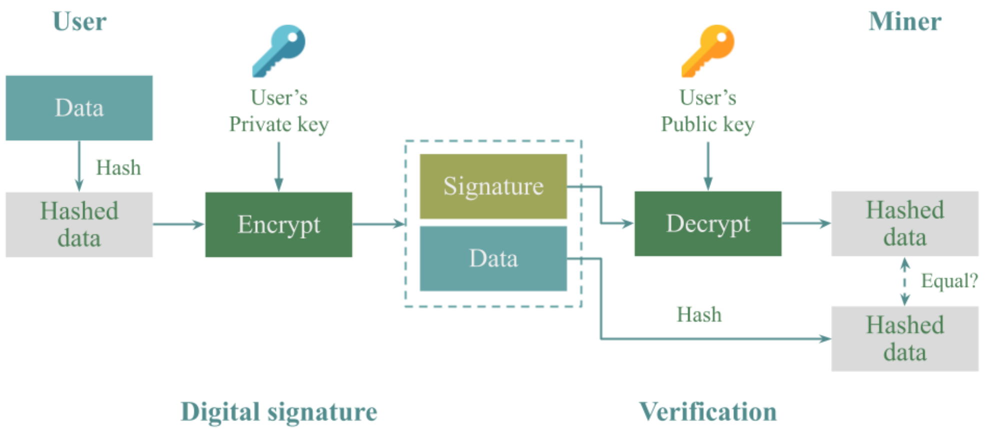

Figure 2.1: Digital signature and verification process in blockchain

support, it plans to incorporate zkSNARKS, a specific type of zero-knowledge proof, into its development roadmap.

- **Hash Functions.** Hash functions play a crucial role in blockchain technology, possessing five significant cryptographic properties. These include producing fixed-size outputs to condense data, being resistant to reverse engineering, making it computationally difficult to find matching outputs for given inputs, avoiding collisions between distinct inputs, and causing significant output changes with even minor input alterations. In a blockchain, cryptographic hash functions interlink blocks by storing the previous block's hash in the current block's header. Transactions within a block undergo hashing and are organized into a Merkle Tree structure. The resulting root of this tree is then stored within the block header, as illustrated in Figure (2.2). This design establishes an immutable, secure, and reliable distributed ledger. Any modifications to blocks or transactions are easily detectable, maintaining the integrity of the entire blockchain.

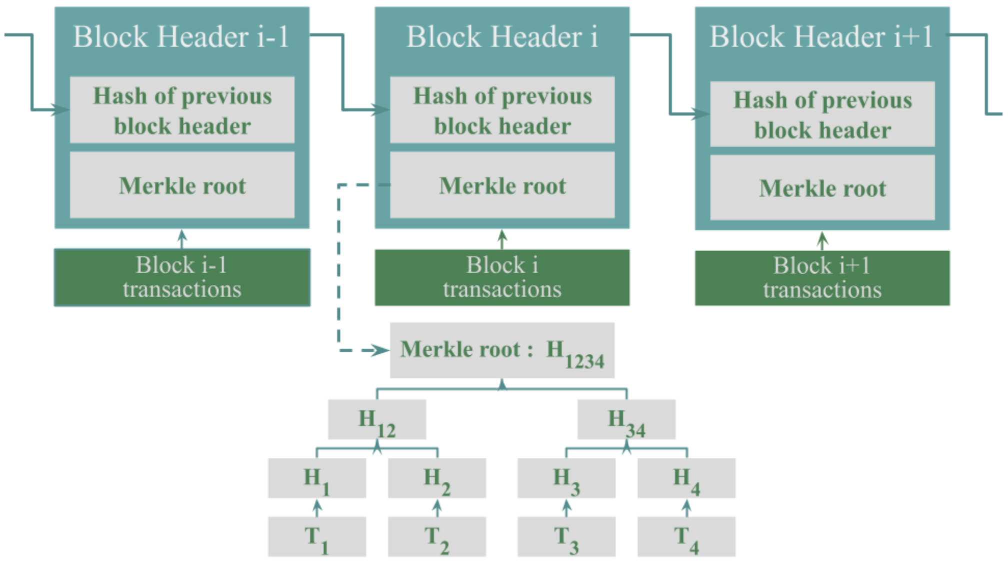

Figure 2.2: Structure of blockchain connections alongside a Merkle tree utilizing hash functions

Since the 1990s, smart contracts have emerged as digital transaction mechanisms built on blockchain technology. They function as containers of code that digitize real-world contract terms, effectively translating legally binding agreements between parties into the digital realm. These contracts can replace trusted intermediaries by using code execution, which is autonomously distributed and validated by nodes within the blockchain network [70].

Smart contracts add dimension to the blockchain. They not only establish an immutable record of events but also enable the creation of precise, objective computer code that defines the management process and necessary actions for specific events. Initially proposed in Ethereum to expand upon Bitcoin's capabilities, smart contracts respond to significant events and aren't constrained by the involvement of multiple parties or legal bindings.

Often referred to as chain codes, smart contracts:

- Embed programmatic rules and decision-making into blockchain operations.
- Automate and standardize transactions, ensuring adherence to predefined rules.
- Operate directly on the blockchain.

Smart contracts are poised to revolutionize business practices and form the cornerstone of enterprise blockchain applications. They empower anyone to develop contracts without intermediaries, delivering autonomy, efficiency, accuracy, and cost savings to various industries [35].

2.2 Decentralized Finance (DeFi) and Lending Protocols

Decentralized finance (DeFi) is a financial system built on the backbone of blockchain technology and smart contracts. Permissionless access, trustlessness, transparency, and network connection are among the benefits of DeFi. DeFi can support a wide range of financial services, including not just lending protocols, but also the operation of Decentralized Exchanges (DEXs) and the construction of sophisticated payment systems. This novel paradigm ushers in a new age of financial transactions, providing open access and reducing reliance on intermediaries while assuring better visibility and connectedness across the network.

DeFi lending protocols let users deposit tokens as collateral, lend another token, and give liquidity to lending pools. Liquidation, on the other hand, assists procedures in reducing debt exposure when collateral prices decline [68]. Here are some prominent DeFi lending protocols [53]:

- **Maker.** Maker stands out as a distinctive DeFi lending platform. It empowers users to borrow funds through its DAI stablecoin, tied to the US dollar value. Utilizing Maker is accessible to all, involving the creation of a vault where collateral like ETH is locked to generate DAI debt.
- **Aave.** Aave, an open-source platform, ranks among the most popular DeFi lending protocols. Functioning as a non-custodial liquidity platform, Aave enables interest earnings on deposits and asset borrowing. Lenders deposit cryptocurrencies, receiving tokens in return. Interest rates adjust algorithmically, proportionate to the tokens held.
- **Kava.** Kava, a Layer-1 blockchain, provides DeFi services across blockchains, offering stablecoins, bonds, and lending. Its goal is diverse open finance, including security and cross-chain features. This aids swift cross-chain DeFi app development for global users. Kava's lending platform and money market app create a decentralized digital asset bank, enabling users to maximize their assets through stablecoins, loans, and interest-bearing accounts.

- **Compound.** Compound, an autonomous money market protocol, is another well-known DeFi lending option. It permits deposits for interest and borrowing against cryptocurrency holdings. Using smart contracts, Compound simplifies capital management and storage. The protocol supports lending and borrowing of assets such as DAI, ETH, WBTC, and more.

Figure 2.3: Example of lending protocols: Maker [5], Aave [1], Kava [4], and Compound [2]

These protocols introduce dynamic ways for users to engage with decentralized lending, enhancing the DeFi landscape.

2.3 MakerDAO

MakerDao is one of the peer-to-peer lending protocols that distributes Dai. Dai is a stablecoin whose value is pegged to the US dollar. As the demand for Dai changes, the demand curve shifts owing to market conditions, Dai holders' confidence, or other causes, and the supply curve is adjusted via a permissionless credit factory on Ethereum. As a result, the algorithm can keep the Dai price as close to 1 US dollar as possible.

Users can borrow Dai by securing their cryptocurrency assets as collateral. To reclaim the collateral, they must return Dai plus a charge. As long as their collateralized vault value exceeds their collateral ratio, their loan status will be secured. If the collateral value (in US dollars) goes too low, the protocol auctions off a portion of the collateral to cover the outstanding debt and penalty cost. The protocol will burn Dai to reduce supply, and the vault owner will get any leftover collateral. This process can be illustrated in Figure (2.4).

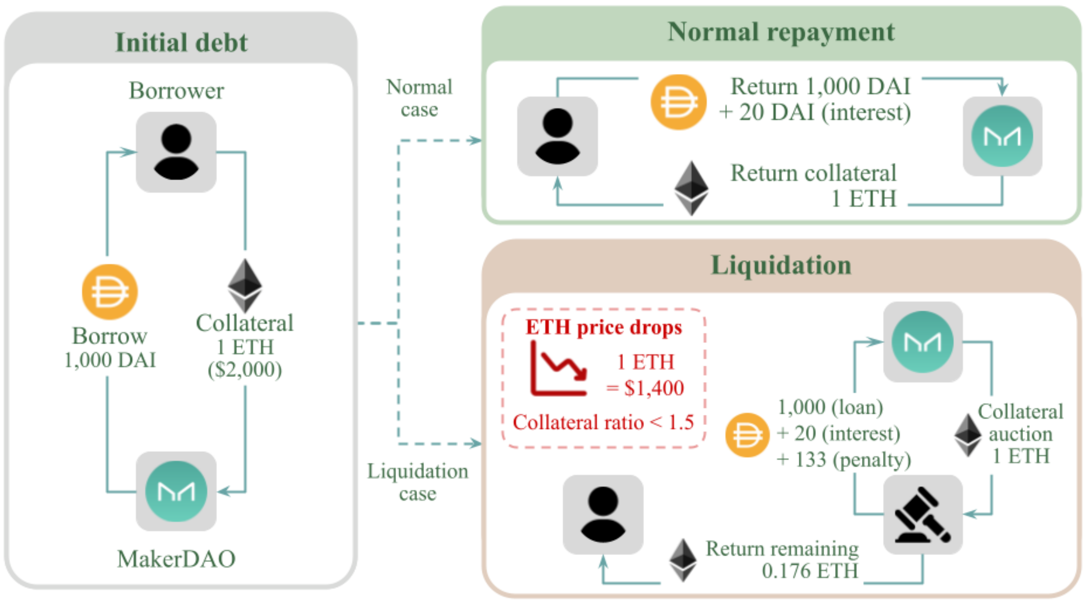

Figure 2.4: Process of borrowing and repaying MakerDAO

Here are some examples of important primary modules in Figure (2.5) that function in tandem to allow MakerDAO's operations:

- **Dai Module.** Anchoring the system, the Dai Module encompasses the Dai Token Contract. This contract manages the Dai stablecoin, upholding its integrity and peg to the US Dollar. It oversees the generation, transfer, and removal of Dai tokens, upholding stability across the ecosystem.
- **Vault Core Module.** The Vault Core Module houses pivotal contracts, namely Vat and Spot. The Vat contract functions as a ledger, diligently documenting the status of all collateral-backed vaults within the system. Meanwhile, the Spot contract calculates collateral asset values, determining their influence on the overall system's well-being.
- **Collateral Module.** Within the Collateral Module, essential contracts named Join and Clip play a central role. The Join contract admits new collateral varieties into the system, permitting users to mint Dai against them. Conversely, the Clip contract streamlines collateral liquidation, curbing potential losses and preserving system stability.

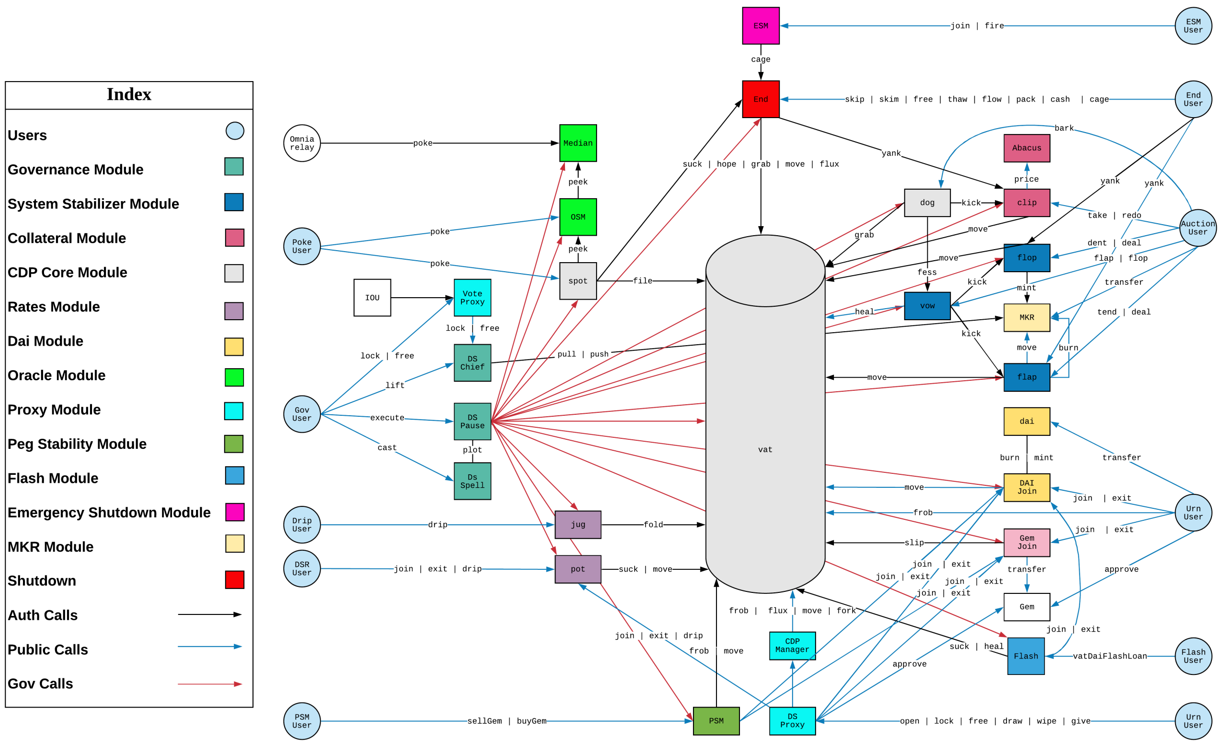

Figure 2.5: MakerDao module entities [18]

These modules work harmoniously, establishing the functional structure of MakerDAO. Their collaboration ensures the dependable issuance and control of Dai, adeptly managing diverse collateral types and associated risks.[42].

2.4 Markowitz Model for Portfolio Optimization

The Markowitz model, developed by Harry Markowitz in 1952 [54], is an essential tool for portfolio optimization in the financial world. It assists in the selection of the most effective portfolio by analyzing various combinations of available assets. According to Markowitz's Modern Portfolio

Theory (MPT), the ideal portfolio strikes a delicate equilibrium between risk and return. This equilibrium ensures that investors either attain the highest possible return for a given level of risk or minimize their risk exposure while achieving a specific rate of return [61]. Subsequently, the mean-variance framework was established. This model is based on its reliance on expected returns (mean) and standard deviation (variance) to assess various portfolio combinations. It is the foundation of modern portfolio theory [14].

In [40], the authors studied portfolio optimization using Analog Complexing (AC) to predict the yield of indexes based on which they built a model called AC-CVaR derived from the mean-variance model applied to China's privately offering funds. They performed the mean-variance model as

$$\begin{aligned} \omega^* &= \arg\max_{\omega} \left(\omega^T \mu - \frac{\lambda}{2} \omega^T \Sigma \omega \right) \\ \text{subject to } \omega^T \mathbf{1} &= 1 \\ \omega &\geq 0. \end{aligned}$$

In this context, the symbol ω^* denotes the optimal asset weight vector, while ω signifies the weight vector assigned to the assets. Additionally, μ represents the expected return vector of these assets, Σ denotes the covariance matrix, λ stands for the risk aversion coefficient, and $\mathbf{1}$ is the column vector consisting of all elements set to 1.

This model can be categorized into two types depending on the investor's requirements:

- **Target return model**. The model sets μ_0 as the asset allocation target income value and can be rewritten as

$$\begin{aligned} \omega^* &= \arg\min_{\omega} \left(\omega^T \Sigma \omega \right) \\ \text{subject to } \omega^T \mu &= \mu_0 \\ \omega^T \mathbf{1} &= 1 \\ \omega &\geq 0. \end{aligned}$$

- **Target risk model**. The model sets σ_0 as the asset allocation target risk value and can be rewritten as

$$\begin{aligned} \omega^* &= \arg\max_{\omega} \left(\omega^T \mu \right) \\ \text{subject to } \sqrt{\omega^T \Sigma \omega} &= \sigma_0 \\ \omega^T \mathbf{1} &= 1 \\ \omega &\geq 0. \end{aligned}$$

Concerning risk assessment, they utilized the Value at Risk (VaR) model, which can be expressed as follows:

$$\text{Prob}(X \leq \text{VaR}(\alpha)) = \alpha.$$

Here, $\text{Prob}(X \leq \text{VaR})$ signifies the likelihood that the portfolio's loss value X does not surpass the VaR value at a specified confidence level α.

Additionally, they employed Conditional VaR (CVaR), which can be thought of as the average of tail losses, encompassing information about losses exceeding the VaR threshold. Where X denotes the loss value of the portfolio, its mathematical representation is as follows:

$$\text{CVaR} = \mathbf{E}[X|X > \text{VaR}(\alpha)].$$

In this study, various methods for forecasting returns were compared, while another research paper, referenced as [36], delved into optimizing investment portfolios following Markowitz's principles and explored the concept of risk tolerance. The optimization strategy employed in this research involved the application of the Lagrange Multiplier method.

2.5 Review of Blockchain-Based Financial Assets Tokenization

With the evolution of blockchain technology and smart contracts, an option emerges in the form of asset tokenization, which essentially converts assets into commodities that can be traded. Existing researches [25] [23] investigate the best methods for two unique scenarios: discrete and continuous homogeneous cases. These methods are aimed at answering a fundamental question about the optimization of standardized package assembly within a given portfolio.

Let's denote the existence of N assets, where $N \geq 1$, and $A_N \geq \cdots \geq A_1 > 0$, representing the expected returns of each asset. The uncertainty associated with each unit of return is represented by random variables $\xi_1, \cdots, \xi_N$, and it holds that $\mathbf{E}\xi_n = 1$, where $n \in \bar{N}$. The covariance is symbolized as $\operatorname{cov}(\xi_i, \xi_j) = K_{ij}$, where $i, j \in \bar{N}$.

A package, denoted by vector $\vec{c} \in \mathbb{R}^N$, is formed from the portfolio $(\vec{A}, \vec{\xi})$, with $0 \leq \vec{c} \leq \vec{A}$ and $\mathbf{E}\vec{c}^T\vec{\xi} = 1$. Since the mathematical expectation of all components of $\vec{\xi}$ is unity and the components of $\vec{c}$ are non-negative, then $\mathbf{E}\vec{c}^T\vec{\xi} = \|\vec{c}\|_1 = \sum_{n=1}^N c_n$.

Representing the covariance matrix as $\mathbf{K} = (K_{ij})_{i,j=1}^N$, the variance of package $\vec{c}$ is given by

$$\text{V}(\vec{c}) = \text{Var}\, \vec{c}^T\vec{\xi} = \vec{c}^T\mathbf{K}\vec{c}, \tag{2.1}$$

while a collection of M packages can be expressed as

$$\mathbf{C}_M = (\vec{c}_1|\ldots|\vec{c}_M) \in \mathbb{R}^{N\times M}. \tag{2.2}$$

$\mathbf{C}_M$ represents the tokenization of portfolio $(\vec{A}, \vec{\xi})$ if $\sum_{m=1}^M \vec{c}_m \leq \vec{A}$. The variance of $\mathbf{C}_M$ is determined by the maximum variance among its packages: $\text{V}(\mathbf{C}_M) = \max_{m\in\bar{M}} \text{V}(\vec{c}_m)$.

Given a specific portfolio $(\vec{A}, \vec{\xi})$ and a predefined variance threshold $\sigma^2 > 0$, the problem of portfolio sold-out can be stated as:

$$M \to \max_{M, \mathbf{C}_M : \text{V}(\mathbf{C}_M) \leq \sigma^2}. \tag{2.3}$$

2.5.1 Discrete Homogeneous Method

In this method, the underlying assumption is built upon the characterization that its constituents $\vec{\xi}$ are uncorrelated and possess equal variances, denoted as $\forall n \in \bar{N} : K_{nn} = \sigma_0^2$. Consequently, the portfolio $(\vec{A}, \vec{\xi})$ exhibits homogeneity, driven by the parameter σ_0^2.

For discrete homogeneous tokenization, a collection of M packages is given by:

$$\mathbf{D}_M = \left(\vec{d}_1|\ldots|\vec{d}_M\right) \in \{0,1\}^{N\times M}. \tag{2.4}$$

This arrangement represents discrete tokenization governed by the parameter k if $\sum_{m=1}^M \vec{d}_m \leq \vec{A}$ and $\forall m \in \bar{M} : \left\|\vec{d}_m\right\|_1 = k$. Consequently, the problem can be simplified to:

$$M \to \max_{M,\mathbf{D}_M} . \tag{2.5}$$

This method's algorithm is expressed in algorithm(1) as follows.

Algorithm 1: Discrete homogeneous algorithm [23]

```
Input: $\vec{A}, N, k$
Output: $\vec{a}, M, D_M$
Function Preprocessing($\vec{A}, k, N$):
    $\vec{A} = k \cdot \vec{A}$
    $\vec{a} = \vec{A}$
    $S = \sum_0^{N-k+1} \vec{a}$
    $B = 0$
    $n = 2, \cdots, k$
    for $i = N - k + 1$ to $N$ do
        $S = \sum_0^{N-k+n-1} \vec{a}$
        $\vec{a}_{N-k+1} = \min(A_{N-k+1}, S/(n-1))$
        $\delta = A_{N-k+n} - \vec{a}_{N-k+n}$
        $B = B + \delta$
        $S = S + \vec{a}_{N-k+n}$
    return $\vec{a}, B$
Function Distribution($\vec{a}, N, k, B$):
    $S = \sum_0^N \vec{a}$
    $M = [S/k]$
    $B_0 = S - M^* \cdot k$
    for $n = N - B_0 + 1$ to $N$ do
        $\vec{a}_n = \vec{a}_n - 1$
    $B = B + B_0$
    for $n = 1$ to $N$ do
        $l_n = \sum_0^{n-1} \vec{a} + 1$
        $r_n = \sum_0^n \vec{a}$
    $\vec{a} = \vec{a}/k$
    for $i = 1$ to $N$ do
        for $j = 1$ to $M$ do
            $D_M[i, j] = 0$
    for $i = 1$ to $k$ do
        for $j = 1$ to $M$ do
            Find n: $l_n \leq M \cdot (i-1) + j \leq r_n$
            $D_M[n, j] = 1/k$
    return $\vec{a}, M, D_M$
```

2.5.2 Continuous Homogeneous Method

In this method, the fundamental assumption also relies on the premise that its components $\vec{\xi}$ are independent and possess equal variances, much like in the discrete homogeneous method.

For the scenario of continuous homogeneous tokenization, a set of M packages can be presented as follows:

$$\mathbf{C}_M = (\vec{c}_1 | \dots | \vec{c}_M) \in \mathbb{R}^{N \times M}. \tag{2.6}$$

Let $\vec{c}$ denote the average composition of the package. Consequently, we can infer that $\vec{c} = (\bar{c_1}, \cdots, \bar{c_N})^T = \frac{1}{M}\sum_{m=1}^{M} \vec{c}_m$. This leads to $\mathbf{C} = \left(\vec{c} | \cdots | \vec{c}\right)$. Consider $\vec{a}$ as the vector representing the sums of rows in the matrix $\bar{\mathbf{C}}$, where $\vec{a} \equiv (a_1, \cdots, a_N)^T = M \cdot \vec{c}$.

Hence, the original problem is transformed into the following formulation:

$$[\|\vec{a}\|_1] \to \max, \tag{2.7}$$

subject to the constraints:

$$\forall n \in \bar{N} : 0 \leq a_n \leq A_N, \tag{2.8}$$

$$\forall n \in \bar{N} : \frac{1}{k}\|\vec{a}\|_1^2 \geq \|\vec{a}\|_2^2. \tag{2.9}$$

This method's algorithm is expressed in algorithm(2) as follows.

Algorithm 2: Continuous homogeneous algorithm [23]

```
Input: $\vec{A}, N, k$
Output: $\vec{a}, M, C_M$
1  Function Search($\vec{A}, N, k$):
2      $E = 0$
3      $V = 0$
4      $\forall n \in N : a_n = 0$
5      for $n = 0$ to $N - 1$ do
6          $E_{new} = E + (N - n)(A_{n+1} - A_n)$
7          $V_{new} = V + (N - n)(A_{n+1}^2 - A_n^2)$
8          if $\frac{1}{k}E_{new}^2 \geq V_{new}$ then
9              $E = E_{new}$
10             $V = V_{new}$
11             $\forall k = n + 1, \cdots, N : a_k = A_{n+1}$
12         else
13             $n^* = n + 1$
14             $x = \max\left(\text{Solve}_x\left(\frac{1}{k}(E + (N - n)(x - A_n))^2 - V - (N - n)(x^2 - A_n^2) = 0\right)\right)$
15             $\forall j \geq n^* : a_j = x$
16             $E = E + (N - n)(x - A_n)$
17             $V = V + (N - n)(x^2 - A_n^2)$
18     return $\vec{a}$
19 Function Construct($\vec{a}$):
20     $M = [\|\vec{a}\|_1]$
21     $\vec{a} = \frac{M}{\|\vec{a}\|_1} \cdot \vec{a}$
22     $\vec{c} = \vec{a}/M$
23     $C_M = \underbrace{(\vec{c}, \cdots, \vec{c})}_{M}$
24     return $\vec{a}, M, C_M$
```

2.6 Optimization Methods

2.6.1 Second-Order Cone Programming

Second-order cone programming (SOCP) is a form of convex optimization problem in which the goal is to minimize a linear function inside the region created by the intersection of an affine set and the combination of second-order cones [7]. SOCPs belong to the category of nonlinear convex problems and encompass linear and convex quadratic programs as specific instances, though they are not as versatile as semi-definite programs (SDPs). SOCP is a subset of SDP focused on optimization at the intersection of an affine set and the domain of positive semi-definite matrices. Furthermore, SOCP lies between linear programming (LP) and quadratic programming (QP) on one end of the spectrum and SDP on the other. Interior point techniques may be used to solve SOCP issues, and while they require more computing effort each iteration than LP and QP approaches, they are still less demanding than handling comparably sized and structured SDPs [13].

The standard form of SOCP and linear programs are explicitly comparable:

$$\begin{aligned} &\min\ f^T x \\ &\text{s.t. } \|A_i x + b_i\|_2 \le c_i^T x + d_i, \quad i = 1, \cdots, n. \end{aligned} \tag{2.10}$$

The optimization variable, denoted as x, belongs to the real vector space of dimension n, which is represented as $x \in \mathbb{R}^n$. The problem's parameters include $f \in \mathbb{R}^n, A_i \in \mathbb{R}^{(n_i-1)\times n}, b_i \in \mathbb{R}^{n_i-1}, c_i \in \mathbb{R}^n$, and $D_i \in \mathbb{R}$. The second-order cone constraint of dimension n_i involves the standard Euclidean norm which can be expressed as $\|u\|_2 = \left(u^T u\right)^{1/2}$.

According to the research [50], this paper highlights several categories of problems that can be stated as SOCP issues. For instance:

1. **Problem involving the quadratically constrained quadratic programming (QCQP)**. The optimization problem:

$$\begin{aligned} \min\ & x^T P_0 x + 2q_0^T x + r_0 \\ \text{subject to } & x^T P_i x + 2q_i^T x + r_i \le 0, \quad i = 1, \ldots, p, \end{aligned}$$

   the matrices $P_0, \ldots, P_p \in \mathbb{R}^{n\times n}$ are symmetric and positive semi-definite. This problem can be reformulated and solved by SOCP as

$$\begin{aligned} \min\ & t \\ \text{subject to } & \|P_0^{1/2} x + P_0^{-1/2} q_0\|_2 \le t, \\ & \|P_i^{1/2} x + P_i^{-1/2} q_i\|_2 \le \left(q_i^T P_i^{-1} q_i - r_i\right)^{1/2}, \quad i = 1, \ldots, p. \end{aligned}$$

   In this reformulated problem, we introduce a new optimization variable $t \in \mathbb{R}$ to minimize, and it is structured as a SOCP problem.

2. **Problem involving the summation of norm**. Consider the unconstrained problem:

$$\min \sum_{i=1}^{p} \|F_i x + g_i\|_2,$$

   given that $F_i \in \mathbb{R}^{n_i \times n}$ and $g_i \in \mathbb{R}^{n_i}$ where $i = 1, \ldots, p$. This problem is derived as SOCP by

$$\begin{aligned} \min\ & \sum_{i=1}^{p} t_i \\ \text{subject to}\ & \|F_j x + g_j\|_2 \le t_j, \quad j = 1, \dots, p, \end{aligned}$$

where $t_i \in \mathbb{R}$ and $x \in \mathbb{R}^n$ are variables for this problem.

3. **Problem involving hyperbolic constraint**. Given that

$$w^2 \le xy, x \ge 0, y \ge 0 \Longleftrightarrow \left\| \begin{bmatrix} 2w \\ x - y \end{bmatrix} \right\|_2 \le x + y,$$

let's consider the following problem:

$$\begin{aligned} \min\ & \sum_{i=1}^{p} 1/(a_i^T x + b_i) \\ \text{subject to}\ & a_i^T x + b_i > 0, \quad i = 1, \dots, p, \\ & c_i^T x + d_i \ge 0, \quad i = 1, \dots, q. \end{aligned}$$

This is a convex problem since $1/(a_i^T x + b_i)$ is convex when $a_i^T x + b_i > 0$. We can rewrite this problem with hyperbolic constraints as

$$\begin{aligned} \min\ & \sum_{i=1}^{p} t_i \\ \text{subject to}\ & t_i \left(a_i^T x + b_i\right) \ge 1, \quad t_i \ge 0, i = 1, \dots, p, \\ & c_i^T x + d_i \ge 0, \quad i = 1, \dots, q. \end{aligned}$$

We can rewrite this problem as

$$\begin{aligned} \min\ & \sum_{i=1}^{p} t_i \\ \text{subject to}\ & \left\| \begin{bmatrix} 2 \\ a_i^T x + b_i - t_i \end{bmatrix} \right\|_2 \le a_i^T x + b_i + t_i, \quad i = 1, \dots, p \\ & c_i^T x + d_i \ge 0, \quad i = 1, \dots, q. \end{aligned}$$

4. **Problem involving matrix fraction**. This problem can be found in the form:

$$\begin{aligned} \min\ & (Fx + g)^T (P_0 + x_1 P_1 + \ldots + x_p P_p)^{-1} (Fx + g) \\ \text{subject to}\ & P_0 + x_1 P_1 + \ldots + x_p P_p \succ 0, \\ & x \ge 0. \end{aligned}$$

The problem variable is $x \in \mathbb{R}^n$ and denote that $P_i \in \mathbb{R}^{n \times n}$ is symmetric, $F \in \mathbb{R}^{n \times p}$, and $g \in \mathbb{R}^n$. We can derive this problem from the SDP problem as

$$\begin{aligned} \min\ & t \\ \text{subject to}\ & \begin{bmatrix} P(x) & Fx + g \\ (Fx + g)^T & t \end{bmatrix} \succeq 0, \end{aligned}$$

where $P_0 + x_1 P_1 + \ldots + x_p P_p$. When P_i are positive semi-definite and assume that the matrix P_0 is non-singular, we can reformulate the problem as

$$\begin{aligned} \min\ & t_0 + t_1 + \ldots + t_p \\ \text{subject to } & P_0^{1/2} y_0 + P_1^{1/2} y_1 + \ldots + P_p^{1/2} y_p = Fx + g \\ & \|y_0\|_2^2 \le t_0 \\ & \|y_i\|_2^2 \le t_i x_i, \quad i = 1, \ldots, p \\ & t_i, x_i \ge 0, \quad i = 1, \ldots, p. \end{aligned}$$

We can derive this problem as SOCP problem as follows:

$$\begin{aligned} \min\ & t_0 + t_1 + \ldots + t_p \\ \text{subject to } & P_0^{1/2} y_0 + P_1^{1/2} y_1 + \ldots + P_p^{1/2} y_p = Fx + g \\ & \left\| \begin{bmatrix} 2y_0 \\ t_0 - 1 \end{bmatrix} \right\| \le t_0 + 1 \\ & \left\| \begin{bmatrix} 2y_i \\ t_i - x_i \end{bmatrix} \right\| \le t_i + x_i, \quad i = 1, \ldots, p. \end{aligned}$$

The duality concept for this particular problem can also be seen as a specific instance of the general duality theory applicable to problems that come with non-negativity constraints, as discussed in [60]. To put it concisely, the problem of convex quadratic programming has an equivalent representation as a SOCP problem.

SOCP qualifies as a convex programming problem primarily because second-order cones are inherently convex sets. Furthermore, it's worth noting that the problem's dimensionality exceeds two and doesn't conform to a polyhedral structure. This unique characteristic implies that the feasible region isn't bounded by polyhedra [13]. For practical problem-solving, it's worth mentioning that the CVXPY package is a valuable tool for effectively addressing and solving SOCP problems.

In the research presented in the paper [24], the authors conducted a comparative analysis of two portfolio optimization methods: QP and SOCP. They utilized daily data from SP500 stocks spanning the period from 2005 to 2010 to assess the performance of these two optimization approaches. The study's findings revealed that SOCP outperformed QP in terms of portfolio optimization.

In another research paper [63], the authors delved into the Sharpe ratio, which is a performance measure based on the mean-variance methodology for evaluating portfolio performance. Within this paper, the authors proposed a Sharpe-ratio portfolio solution employing SOCP. Their approach involved representing the nonlinear portfolio problem using penalty-regularized methods and incorporating a Markov chain structure to depict the underlying asset price dynamics. To identify the optimal portfolio within the context of Markov chains, the authors introduced the use of SOCP as a solution method. The paper showcased the efficiency and effectiveness of this approach through a numerical example.

2.6.2 Metaheuristic Algorithms

Optimization algorithms can be broadly classified into two categories: local and global algorithms [72]. Local algorithms, which are commonly gradient-based methods, focus on refining solutions in the vicinity of a current solution. On the other hand, global algorithms, often utilizing non-gradient-based or evolutionary techniques, aim to explore a wider solution space to locate optimal outcomes. In the present landscape of diverse and intricate research challenges, certain problems may prove unsuitable for gradient-based methods, particularly when they exhibit characteristics such as multi-modality, non-continuity, and non-differentiability [32].

Metaheuristic Algorithms are commonly recognized as global optimization approaches. Their versatility across various domains has attracted significant academic interest. Although conventional and earlier Metaheuristic Algorithms have demonstrated success, they couldn't ensure discovering the global optimum for all optimization problems. Consequently, researchers have devised novel algorithms, culminating in the introduction of an adaptable and effective Metaheuristic Algorithm known as the Slime Mould Algorithm (SMA) [48].

SMA is developed from slime mould's inherent oscillating activity. This unique method includes novel features, such as a different mathematical model. This model uses adaptive weights to simulate the propagation wave of the slime mould, incorporating both positive and negative feedback. SMA develops ideal paths by emulating the bio-oscillatory process, enabling effective exploration and exploitation skills for linking resources [48]. The Figure 2.6 illustrates the SMA overview processes.

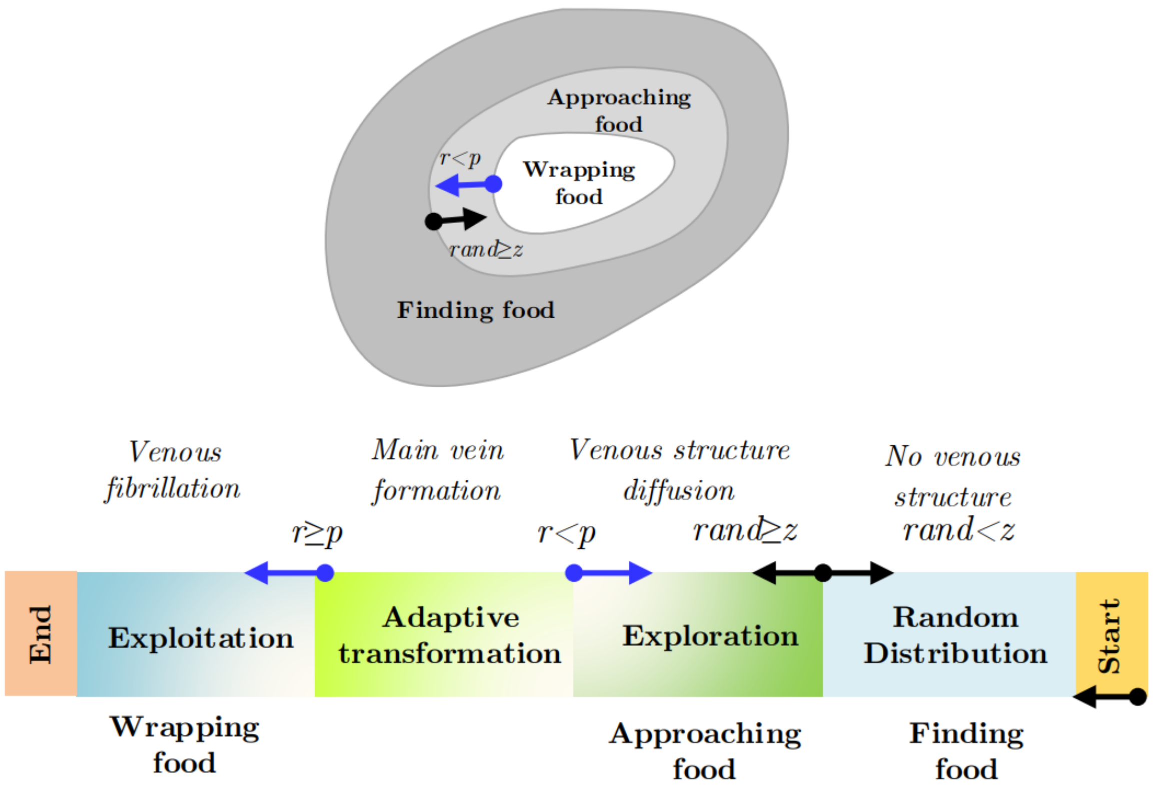

Figure 2.6: SMA overview processes [48]

From the paper [48], the main steps for SMA can be described as follows:

1. **Approach food.** This step is inspired by the behavior of slime mould when it approaches food which can be formulated in the mathematical equation as

$$X_{t+1} = \begin{cases} X_b(t) + v_b(WX_A(t) - X_B(t)), & r < p \\ v_c X_t, \quad r \leq p. \end{cases}$$

In this equation, all variables can be described as follows:

- X: the current location of a slime mold.
- t: the current iteration or time step.

- X_b: the individual location with the highest odor concentration discovered so far.
- v_b: parameter whose value falls within the range of $[-a, a]$ where

$$a = \operatorname{arctan}\left(-\left(\frac{t}{\max_t}\right) + 1\right).$$

- W: parameter signifies the weight or importance of the slime mold's actions where $SmellIndex = sort(S)$ and

$$W(SmellIndex(i)) = \begin{cases} 1 + r\log((b_F - S(i))/(b_F - w_F) + 1), & condition \\ 1 - r\log((b_F - S(i))/(b_F - w_F) + 1), & others. \end{cases}$$

  The condition specifies that $S(i)$ must be the top-ranked individual in the first half of the population. r represents a random value and $r \in [0, 1]$. The b_F is the best fitness achieved in the ongoing iteration, w_F is the worst fitness in the current iteration, and $SmellIndex$ is the sorted list of fitness values, ascending in a minimization problem.
- X_A and X_B: the locations of two individuals randomly selected from the swarm.
- v_c: parameter decreases linearly from 1 to 0 over time.
- p : parameter formulated from

$$p = \tan|S(i) - DF|,$$

  where $S(i)$ is the fitness of X, $i \in 1, \dots, n$, and DF is the best fitness from all iterations.

2. **Wrap food.** This step is inspired by the process when slime mould updates the location of food. It can be formulated as

$$X^* = \begin{cases} rand(UB - LB) + LB, & rand < z \\ X_b(t) + v_b(WX_A(t) - X_B(t)), & r < p \\ v_c X(t), & r \geq p, \end{cases}$$

   where $rand$ and r are the random values and $rand, r \in [0, 1]$. LB and UB signify the lower and upper search range limits.

3. **Oscillation.** In this step, the variable v_b exhibits random oscillations within the range of $[-a, a]$ and progressively converges towards zero as the number of iterations increases. Meanwhile, v_c fluctuates within the interval of $[-1, 1]$ and gradually approaches zero over time.

2.6.3 Integer Programming

Integer programming (IP) is a type of discrete optimization problem that is the discrete analog of the linear programming model considered in the preceding section [56]. For integer programming problems, all variables must take on integer values [17]. Consequently, mixed-integer programming (MIP) refers to problems where not all variables need to be integers [73].

In reference to [51], we can examine the following optimization problem:

$$\begin{aligned} \text{max} \quad & \sum_{j=1}^{n} c_j x_j \\ \text{subject to} \quad & \sum_{j=1}^{n} a_{ij} x_j = b_i, \quad i = 1, \ldots, m \\ & x_j \geq 0, \quad j = 1, \ldots, n. \end{aligned}$$

This type of problem is referred to as a linear integer programming problem. To clarify further, if all variables x_j are integers, it is termed pure integer programming. Conversely, if only some variables must be integers and others can be continuous, it is termed mixed-integer programming.

2.7 Subset Sum Problem

The subset sum problem (SSP) can be considered a specific case of the knapsack problem (KP) [45]. The SSP involves a set N consisting of n items $\{1, \ldots, n\}$, each assigned a positive integer weight represented as $w_1, \ldots, w_n$. Additionally, there is a given capacity constraint denoted as c. The objective of SSP is to identify a subset of items from N in a way that maximizes the total weight while ensuring that the sum of weights in the selected subset does not exceed the specified capacity c. In simpler terms, SSP seeks to find the most valuable combination of items within the weight limit and can be formulated as follows:

$$\begin{aligned} \text{max} \quad & \sum_{j=1}^{n} w_j x_j \\ \text{subject to} \quad & \sum_{j=1}^{n} w_j x_j \leq c \\ & x_j \in \{0, 1\}, \quad j = 1, \ldots, n. \end{aligned}$$

The SSP, when formulated as a decision problem, seeks to determine whether there exists a subset of a given set with N elements such that the sum of the weights associated with the elements in this subset equals the capacity value c. To distinguish this decision problem from the optimization variant, we refer to it as SSP-Decision. In the realm of cryptography, SSP-Decision, when it has unique solutions, corresponds to a secret message intended for transmission. Despite being a specific instance of the KP, SSP-Decision remains NP-hard [41].

The study described in the paper [11] delved into the distinctions between manageable and challenging adaptations of this particular problem. In this paper, SSP involves a set of positive integers $a_1, a_2, \ldots, a_n$ and a target integer t. The goal is to figure out if there is a subset of the integers a_i that sums up to the desired number t.

This problem is considered NP-complete according to an original reference [30], although it can be solved efficiently using dynamic programming through a pseudo-polynomial time algorithm. It is worth noting that if the sequence $a_1, a_2, \ldots, a_n$ follows a specific property called super-increasing, then we can solve SSP in polynomial time. To clarify, a sequence $a_1, a_2, \ldots, a_n$ is considered super-increasing if $\sum_{i=1}^{j} a_i \leq a_{j+1}$.

The Subset Sum with Repetitions Problem (SRP) involves a set of positive integers, denoted as $a_1 < \cdots < a_n$, along with an integer target value t. Additionally, there's a corresponding set of non-negative integers, namely $r_1, \ldots, r_n$, which represent the allowable repetition counts for each corresponding element in the first set.

The goal of SRP is to determine whether there exist non-negative integers x_i, where $0 \leq$

$x_i \leq r_i$ for $i = 1, \ldots, n$, such that the sum of the products of a_i and x_i equals the target value t, expressed as $\sum_{i=1}^{n} a_i x_i = t$.

It should be noted that the number of repeats, as indicated by the values r_i, for each element in SRP can grow exponentially with respect to the number of elements n. Consequently, SRP is classified as an NP-complete problem since it encompasses the classic SSP, where each r_i is constrained to be 1. However, it is noteworthy that we can also solve SRP in polynomial time for any fixed value of n, as established in the original reference [47].

In this paper, we remind that a sequence represented as $a_1, \ldots, a_n$ is classified as an arithmetic progression when there exists a number j such that the elements of the sequence can be expressed as $a_i = a_1 + (i-1)j$ for i ranging from 2 to n. To succinctly describe such a sequence, we can use the triple notation (a_1, n, j). The variations of SSP can be described as follows:

- SSP is solvable in polynomial time if the sequence $a_1, \ldots, a_n$ is restricted to be an arithmetic progression, even if specified concisely by the triple (a_1, n, j).
- SRP is NP-complete even if the sequence $a_1, \ldots, a_n$ is super-increasing and $r_i = 1$ or 2 for all i.
- The given optimization problem, often referred to as the Sequential Knapsack Problem (SKP), can be described as follows:

$$\begin{aligned} \text{max} \quad & \sum_{j=1}^{n} a_j x_j \\ \text{subject to} \quad & \sum_{j=1}^{n} a_j x_j \leq c \\ & 0 \leq x_j \leq r_j, \quad j = 1, \ldots, n. \end{aligned}$$

  It is worth noting that there exists a polynomial-time algorithm to solve the SKP efficiently when the sequence of coefficients $a_1, \ldots, a_n$ is constrained to form a chain-like structure.

In the paper [66], the authors explored the algorithmic complexity of the SSP when considering binary multiplicity encoding. In this encoding scheme, each item, denoted as i in the input, is represented by a positive integer multiplicity u_i. This multiplicity value indicates that up to u_i copies of the item can be used in a solution. These variants involve a smaller number of distinct items, n, with a total multiplicity denoted as N, calculated as the sum of all u_i values. Binary multiplicity encoding poses challenges because algorithms need to run efficiently in polynomial time in terms of the input size, which can be exponentially smaller compared to a naive encoding.

To formally define the SSP with multiplicities, it can be framed as a feasibility integer linear program with the following constraints:

- $0 \leq x_i \leq u_i$ for each item i, where x_i is an integer variable representing the number of copies of item i in the solution.
- $\sum_{i \in [n]} s_i x_i = t$, where s_i represents the value associated with item i, and t is the target sum to be achieved.

Throughout the paper, the authors use the symbol u to represent the maximum item multiplicity, defined as the maximum among all u_i values. Additionally, they assume that u is lower than or equal to the target sum t.

As a result, their focus shifted towards developing pseudo-polynomial time algorithms concerning the maximum item size, denoted as $s = \max_{i \in [n]} s_i$. This choice is essentially equivalent to focusing on s as opposed to another parameter, such as t. It's worth noting that s can be considerably smaller than t but not the other way around.

For solving the Subset Sum problem with multiplicities, they devised randomized algorithms with an approximate runtime of $\tilde{\mathcal{O}}(n + s^{5/3})$. These algorithms have a one-sided error and provide a correct answer with a high probability. Furthermore, their algorithms are capable of retrieving a solution without significantly increasing the asymptotic running times. This feature is particularly noteworthy for the Subset Sum problem, where they employ the Bringmann-Wellnitz algorithm as a black box, which only yields yes/no answers. Their ability to handle this limitation stems from their capacity to allocate more time to retrieve a solution compared to what the Bringmann-Wellnitz algorithm can afford.

However, it's essential to acknowledge a limitation of their algorithms. They can only provide an answer for a single target value, denoted as t, at a time. In contrast, several known Knapsack and Subset Sum algorithms can offer solutions for all target values ranging from 0 to t concurrently. Nonetheless, this limitation is inherent because their primary aim is to achieve running times that are independent of the target value t. To this end, they cannot afford an output size linearly proportional to t since t cannot be bounded solely in terms of n and s.

In summary, the pursuit of pseudo-polynomial time algorithms for the SSP has been driven by its special relationship with the KP, to significantly enhance its computational efficiency. Notable developments in this area indicated in this paper [66] include:

1. Pisinger's algorithm [65], which operates in $\mathcal{O}(Ns)$ time for Subset Sum.

2. Koiliaris and Xu's groundbreaking improvement over Bellman's $\mathcal{O}(Nt)$ time algorithm [44], presenting $\tilde{\mathcal{O}}(\sqrt{N}t + N)$, $\tilde{\mathcal{O}}(N + t^{5/4})$, and $\tilde{\mathcal{O}}(\Sigma)$ time deterministic algorithms for Subset Sum, where Σ represents the total sum of items. These algorithms all leverage a common technique of encoding Subset Sum as a convolution problem solvable via Fast Fourier Transform.

3. Bringmann's randomized $\tilde{\mathcal{O}}(N+t)$ time algorithm for Subset Sum, based on the color-coding technique [20].

4. Jin and Wu's alternative randomized $\tilde{\mathcal{O}}(N+t)$ time algorithm, employing Newton's iterative method. Their proof is known for its concise nature [39].

Taking a different approach, Galil and Margalit [29] utilized additive combinatorics methods to demonstrate that Subset Sum can be solved in nearly linear time when t significantly exceeds $\sum s/N^2$, assuming all items are distinct. More recently, Bringmann and Wellnitz [21] extended this result to multisets.

By combining their algorithm with the $\tilde{\mathcal{O}}(N + t)$ time algorithm [20], a $\tilde{\mathcal{O}}(N + u^{1/2}s^{3/2})$ time algorithm for Subset Sum with multiplicities is derived. For $u = 1$, this yields the current fastest $\tilde{\mathcal{O}}(N + s^{3/2})$ time algorithm, especially when s is small. Their $\tilde{\mathcal{O}}(n + s^{5/3})$ time algorithm represents an improvement over their result, particularly for cases where u is much larger than $s^{1/3}$. It's worth noting that even without binary encoding for multiplicities, this improvement remains significant, as the scenario with $u = 1$ necessitates distinct item sizes. For instance, achieving an $\mathcal{O}(N + s^{1.99})$ time algorithm is not a direct consequence of [20] when multiple items can share the same size.

2.8 Market Concentration

In the field of economics, a market concentration (MC) quantifies the proportion of a specific company's market share in relation to the overall size of the market. This ratio serves as a gauge of a company's magnitude relative to the entire market [34]. The market concentration (MC_n) is typically calculated using the following formula:

$$MC_n = S_1 + S_2 + \ldots + S_n,$$

where, S is the market share of the nth largest company [59].

In the business context, this ratio can also be calculated using alternative factors like revenue (turnover), workforce size, profitability, gross value added (GVA), and overall output [52].

The market concentration ranges from 0% to 100%, and it provides insights into the level of competition within the industry. When the market concentration falls within the 0% to 50% range, it suggests a high degree of competitiveness, often referred to as low market concentration. A commonly used guideline is that an oligopoly emerges when the top five companies in the market collectively control more than 60% of total market sales. Conversely, if a single company's market concentration reaches 100%, it signifies a monopoly within that industry [43].

2.9 Gini Coefficient

The Gini coefficient, developed through a series of research studies conducted by Corrado Gini, serves as a crucial measure for evaluating the concentration of wealth and income within a society [33]. This statistical metric is employed to gauge economic inequality within a given population. It provides a numerical representation of the distribution of income or wealth within an economy, offering insights into the degree to which this distribution deviates from a perfectly equal state [22].

The Lorenz curve is a visual depiction showing how total income is distributed among individuals or households, starting from the least affluent and progressing toward the most affluent. In contrast, the Gini coefficient is a numerical measure that gauges the level of inequality within this income distribution. It does so by calculating the area between the Lorenz curve and a line representing perfect equality, expressed as a percentage of the maximum possible area under that line. Essentially, a Gini coefficient of $0(0\%)$ signifies a state of complete income equality, while an index of $1(100\%)$ represents extreme income inequality [75].

From A note on a property of the Gini coefficient [31], we denote $x_1, \ldots, x_n$ as n non-negative values and Gini coefficient can be defined as

$$G(x_1, \ldots, x_n) := \frac{1}{2n \sum_{i=1}^{n} x_i} \cdot \sum_{1 \le i,j \le n} |x_i - x_j|.$$

To improve efficiency, we can reformulate the equation for calculating the Gini coefficient. This is necessary because the current formula has high computational complexity, with the number of algebraic operations increasing quadratically with n. To achieve faster Gini coefficient calculations, we need to reformulate the equation $\sum_{1 \le i,j \le n} |x_i - x_j|$ as follows:

$$\begin{aligned}
\sum_{1 \le i,j \le n} |x_i - x_j| &= \sum_{\substack{1 \le i,j \le n \\ j<i}} |x_i - x_j| + \sum_{\substack{1 \le i,j \le n \\ j>i}} |x_i - x_j| \\
&= 2 \sum_{1 \le j < i \le n} (x_i - x_j) \\
&= 2 \sum_{i=2}^{n} \sum_{j=1}^{i-1} (x_i - x_j) \\
&= 2 \sum_{i=2}^{n} \left(\sum_{j=1}^{i-1} x_i - \sum_{j=1}^{i-1} x_j \right).
\end{aligned}$$

.

Therefore,

$$\sum_{i=2}^{n}\sum_{j=1}^{i-1} x_i = \sum_{i=1}^{n}(i-1)x_i, \qquad \text{and}$$
$$\sum_{i=2}^{n}\sum_{j=1}^{i-1} x_j = \sum_{i=1}^{n}(n-i)x_i.$$

So, we can derive $\sum_{1\le i,j\le n} |x_i - x_j|$ as

$$\begin{aligned}\sum_{1\le i,j\le n} |x_i - x_j| &= 2\sum_{i=1}^{n}(i-1)x_i - 2\sum_{i=1}^{n}(n-i)x_i \\ &= 2\sum_{i=1}^{n}(i-1-n+i)x_i \\ &= 4\sum_{i=1}^{n} ix_i - 2(n+1)\sum_{i=1}^{n} x_i.\end{aligned}$$

As a result, given that $\sum_{i=1}^{n} x_i > 0$, the Gini coefficient can be reformulated as

$$\begin{aligned}G(x_1,\dots,x_n) &= \frac{1}{2n\sum_{i=1}^{n} x_i}\cdot\left(4\sum_{i=1}^{n} ix_i - 2(n+1)\sum_{i=1}^{n} x_i\right) \\ &= \frac{2\sum_{i=1}^{n} ix_i}{n\sum_{i=1}^{n} x_i} - 1 - \frac{1}{n}.\end{aligned}$$

Furthermore, we can determine the Gini coefficient, as illustrated in Figure 2.7. The Gini coefficient is computed by dividing the area between the perfect equality line and the Lorenz curve (A) by the total area under the perfect equality line ($A+B$), as described in the reference [69], so

$$\text{Gini coefficient} = \frac{A}{A+B}.$$

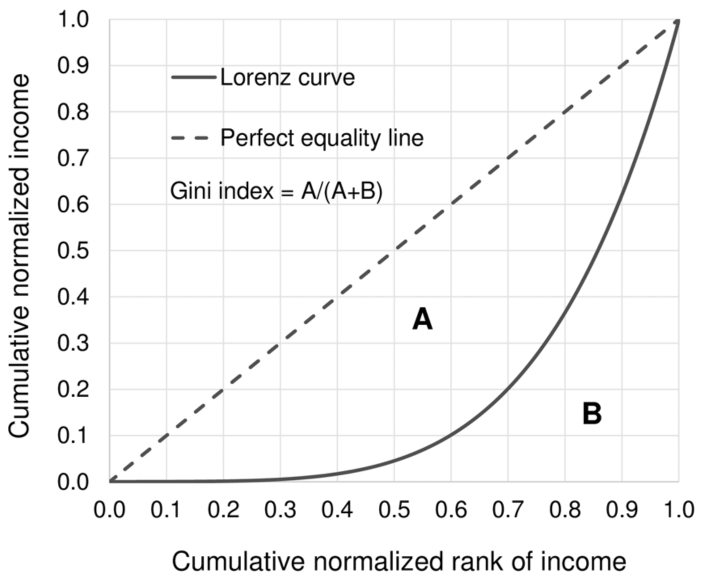

Figure 2.7: Gini coefficient calculated by Lorenz curve [69]

By definition, the Gini coefficient is equal to twice the area between the Lorenz curve and the perfect equality line [10]. This relationship can be derived using the property of triangle, which states that the area of the triangle under the perfect equality line equals $1/2$:

$$A + B = \frac{1}{2}.$$

We can express the Gini coefficient using these areas:

$$\text{Gini coefficient} = \frac{A}{A+B} = \frac{A}{\frac{1}{2}} = 2A.$$

Since we know that $A = \frac{1}{2} - B$, we can express the Gini coefficient as:

$$\text{Gini coefficient} = 2A = 2\left(\frac{1}{2} - B\right) = 1 - 2B.$$

2.10 Literature Review Conclusion

In this chapter, we delve into the foundational technologies and principles underpinning this thesis. This encompasses blockchain and smart contracts, the realm of DeFi and lending protocols, with a particular emphasis on MakerDAO as our primary data source.

Additionally, we explore the Markowitz principle, as discussed in [54, 61, 14], which serves as the bedrock for portfolio optimization. We also draw from the research conducted by [36], where the authors proposed portfolio optimization through the application of the Markowitz principle and its solution using the Lagrange multiplier method. Furthermore, [40] derived the mean-variance model, demonstrating its efficiency.

In the following section, we delve into existing solutions for homogeneous cases and the algorithm for the portfolio sold-out problem, drawing insights from [25, 23]. To tackle the remaining challenges within this thesis, we adopt a numerical approach. This involves exploring optimization methods such as second-order cone programming (SOCP), metaheuristic algorithm (MA), and integer programming (IP).

For SOCP, we establish its principles and problem definitions with references to [7, 13, 50, 60]. Notably, research in [24] demonstrates the effectiveness of portfolio optimization with SOCP and QP, highlighting the superior performance of SOCP.

Additionally, the concept of MA is discussed, with insights drawn from [72, 32, 48]. We also explore IP, detailing problem definitions and drawing from [56, 17, 73, 51].

Subset sum problem (SSP) is a central focus of this study, with its problem definition outlined in [45, 41, 11]. In the research presented by [30], efforts were made to address this problem by formulating it and devising a pseudo-polynomial time algorithm to establish its NP-completeness. Subsequently, the derived problem, SRP, was introduced in [47], demonstrating its solvability within polynomial time. Furthermore, [66] proposed an algorithm designed to handle SSP instances with small items, offering a comparison of its complexity against [65, 44, 20, 39, 29, 21]. Despite these efforts, SSP remains categorized as an NP-hard problem, underscoring its inherent complexity, and researchers continue their endeavors to find effective solutions.

Lastly, we introduced two important economic parameters. The concept of the market concentration was defined based on sources such as [34, 52, 43], while the formula for its calculation was derived from [59]. Similarly, the Gini coefficient was explained using definitions from sources like [33, 22, 75], and its formula was elucidated with reference to [31, 69, 10].

Chapter 3
Methods

This chapter serves as an introduction to both the dataset and the methodology used in this thesis. First, we explain the techniques for data collection and preparation, including the inclusion of essential economic parameters. Following that, we enter into a study of the portfolio sold-out problem inside a blockchain-based real loan data system. In relation to this, we provide optimization methods aimed at handling this complex issue, as well as a thorough explanation of the technique used for results evaluation.

3.1 Data Collection and Data Preparation

As our major case study of interest, this thesis focuses on the MakerDAO lending protocol.

3.1.1 Data Collection

The BigQuery platform was used to obtain the necessary data because of its ability to give access to public datasets via the Google Cloud Public Dataset Program. This platform enabled the retrieval of relevant data by utilizing both classic SQL queries and Google SQL queries. Within the scope of this thesis, the datasets were gathered from three main contracts, including the MCD Vat in Core Modele, MCD Jug in Rates Modules, and PIP ETH in Oracle Security Module (OSM).

MCD Vat Module

The Vat module is the core accounting system in Maker protocol. It is essential in vault management since it monitors the corresponding Dai stablecoin and collateral balances. It also defines guidelines for the acceptable methods for modifying vaults and their related balances. All MakerDAO transactions are stored in this module.

MCD Jug Module

The Jug module is in charge of accumulating stability fees for each form of collateral. These stability fees are critical in calculating the total debt across all vaults of a particular collateral type. Furthermore, the Vat maintains track of the total accumulated debt, and the Dai surplus is also accounted for in this process.

PIP ETH

The Oracle Security Module (OSM) is critical for interacting with updated pricing values after a specified delay. This function is critical for the MakerDAO lending protocol to ensure appropriate pricing. Each sort of collateral, such as Ether (ETH), has its own OSM contract, such as PIP ETH, which has the current ETH/DAI price feed. This structure guarantees that the protocol's pricing is precisely aligned with market fluctuations, demonstrating the protocol's flexibility and personalized collateral management.

The data collection time frame spanned from November 11^{th}, 2019 to March 31^{st}, 2023, allowing for a thorough review. By diving into this data, the research hopes to get detailed insights into the dynamics of the MakerDAO protocol and its many components.

3.1.2 Data Preparation

The data obtained from the blockchain-based system was cryptographically encoded using a hash function, resulting in structured data that people cannot directly comprehend. We used Infura, a third-party service that provides Application Programming Interfaces (APIs) for Ethereum network access, to make data retrieval from the Ethereum network easier. We extracted the original data by using the Web3 library and connecting to the Ethereum testnet using the URL supplied by Infura. In the scope of this study, we concentrated mainly on the ETH-collateralized vault. Because of its widespread application, this sort of collateral is prominent and serves as a substantial asset class. Given its significant price volatility, it has the ability to cause liquidation events under specific conditions.

After decoding the data, we filtered the data that is in 5 functions including **Frob**, **Fork**, **Grab**, **Flux**, and **Fold**. The provided list consists of essential actions with related functional descriptions within the framework of the MakerDAO protocol:

1. **Frob**: This function captures the process of modifying a vault, which includes a wide range of operations such as the smooth transfer of collateral to and from a vault, changing modifications to vault debt, and changes in collateral or debt amounts.

2. **Fork**: This function refers to a specific procedure of separating a vault. This action has two components: the binary approval of a split and the more involved process of vault splitting and merging. This includes factors such as the amount of stablecoin debt to be exchanged and the amount of collateral to swap.

3. **Grab**: This function is important in dealing with the liquidation of an undercollateralized vault. This procedure includes not only the liquidation itself but also the subsequent adjustment of internal balances to guarantee equilibrium.

4. **Flux**: This function allows users to swap collateral tokens among themselves, allowing for the smooth transfer of arbitrary collateral. Notably, because the sender initiated the transaction, prior permission is required before any balance adjustment is performed.

5. **Fold**: This function allows for dynamic change of the debt multiplier. This move, in turn, causes the creation or removal of equivalent debt, giving the protocol's financial landscape greater significance.

3.2 Calculating Essential Economic Parameters

3.2.1 Interest Rate

Given that every transaction data records events at time t, processing this data to ensure that it aligns with real-world values becomes necessary. This processing is required to account for the temporal dimension and handle the issue of time value of money, which affects financial data intrinsically. As a result, the data must be preprocessed in order to correctly reflect economic reality and account for the temporal complexities associated with financial transactions. We needed to calculate the total outstanding debts and associated collateral values to analyze the data effectively. This required the use of the Dai Savings Rate (DSR) and stability fee interest rates.

The Rates Module [6] contains the cumulative stability fees and DSR. It's crucial to note that despite how many vaults and DSR deposits are involved, the technology employed for these accumulation operations must maintain a consistent temporal operation. The stability fee is calculated for each type of collateral, while the interest rates are saved as cumulative rates. The total debt or deposit value is subsequently calculated by multiplying this fee by a normalized debt or deposit amount.

The accumulative rate r at time t can be expressed as

$$r\left(t\right) \equiv r_0 \prod_{i=t_0+1}^{t} f_i, \tag{3.1}$$

where the initial accumulative rate r_0 will be accumulated by stability fees f.

The initial deposit or normalized debt amount a will be summed up as d. The total amount of the loan or deposit at time t can be expressed as

$$d\left(t\right) \equiv a \cdot r\left(t\right) = d_0 \prod_{t=1}^{T} f_i. \tag{3.2}$$

Users may generate debt multiple times for genuine loan data via depositing collateral and borrowing DAI. If we take the initial debt to be $d\left(t_0\right)$, the generated debt to be $d_g\left(t\right)$, and the effective interest rate to be r, we can compute the total debt, which includes stability fees, at time t as follows,

$$d\left(t_i\right) = d\left(t_{i-1}\right) \prod_{t=1}^{i} (1+r)^{t_i - t_{i-1}} + d_g\left(t_i\right), \tag{3.3}$$

where $i = 1, \ldots, T$. We propose to identify the constant interest rate r_c that can reflect the adjusted rate of the interest rate r with the factor term F and improve numerical stability because stability fees are subject to change by the contract as:

$$d\left(t_i\right) = d\left(t_{i-1}\right) \prod_{t=1}^{i} \left(r_c\right)^{\left(t_i - t_{i-1}\right)/F} + d_g\left(t_i\right). \tag{3.4}$$

In the event where $r > 10^{-4}$, we set $F = 30$ in stead of $F = 20$. The total cumulative debt at time T in the event of liquidation is $d\left(T\right) = 0$.

Since the initial approximation is accurate and the Newton-Raphson optimization approach has rapid techniques to calculate roots, r_c, we use these to optimize the rate. Therefore, the interest rate for a specific loan period is

$$r = 2^{\log_2^{r_c}/F} - 1. \tag{3.5}$$

3.2.2 Probability of Default (PD)

In the context of a portfolio sold-out thesis, the Probability of Default (PD) serves a critical role as an essential risk assessment tool used by financial institutions to quantify the likelihood of borrowers defaulting on their financial obligations within a specified time horizon. This fundamental indicator is crucial in assessing credit quality, allowing for an in-depth investigation of many variables such as previous financial performance, current economic conditions, and sector-specific dynamics. In addition to its significance in credit evaluation, the calculation of PD has a substantial impact on decisions regarding the portfolio sold-out problem.

In the realm of PD calculation, the application of Geometric Brownian motion assumes significance. This stochastic process is used to analyze different financial market dynamics, including stock prices, foreign exchange rates, and financial assets. It provides a mathematical framework for simulating random motions, similar to a random walk. The Wiener stochastic process, often known as standard Brownian motion [64], is frequently used as a core idea in financial modeling. The Wiener process, denoted as $W(t)$ for values in the real number space $\mathbb{R}$ when t occurs inside the interval $[0, \infty)$, demonstrates key properties that underlie its importance in quantitative financial studies:

1. $W(0) = 0$.
2. for all $0 \leq t_1 < t_2, W(t_2) - W(t_1) \sim N(0, t_2 - t_1)$.
3. $W(t)$ has independent increments. For all $0 \leq t_1 < t_2 < \cdots < t_n$, the random variables $W(t_2) - W(t_1), W(t_3) - W(t_2), \cdots, W(t_n) - W(t_{n-1})$ are independent.
4. $W(t)$ paths are continuous.

When we consider the first passage times of levels, which can be utilized in a variety of applications, according to Brownian Motion theory, the time when the process reaches an arbitrary threshold for the first time is a random variable [8]. It is possible to assume that the process begins at a certain position $x_0 > 0$ and is bounded to the positive semi-axis by an absorbing barrier x = 0. Assume that $a > 0$ and let $T_a = \inf\{t \geq 0 : B_t = a\}$. Then we will have

$$\mathrm{P}\left(T_a < t\right) = \mathrm{P}\left(\sup_{s \leq t} B_s > a\right). \tag{3.6}$$

Applying the reflection principle of D. Andre, it follows that

$$\mathrm{P}\left(T_a < t\right) = 2\mathrm{P}\left(B_t \geq a\right). \tag{3.7}$$

Since

$$\mathrm{P}\left(B_t \geq a\right) = \frac{1}{\sqrt{2\pi t}} \int_a^\infty e^{-\frac{x^2}{2t}} dx, \tag{3.8}$$

then,

$$P\left(T_a < t\right) = \int_0^t \frac{a}{\sqrt{2\pi s^3}} e^{-\frac{a^2}{2s}} ds. \tag{3.9}$$

In the context of analyzing the Probability of Default (PD) for a single collateral asset, we introduce key variables: let $a(t)$ represent the number of collateral assets and $d(t)$ denote the amount of debt at a given time t. Furthermore, we denote the exchange rate between the collateral asset and the debt asset as $e(t)$, with an initial value of $e(0) = e_0$. Subsequently, the collateralized ratio $r(t)$ for instances when $d(t) > 0$ equates to

$$r(t) = \frac{e(t) \cdot a(t)}{d(t)}. \tag{3.10}$$

In instances where $d(t) = 0$, we establish $r(t)$ as $+\infty$, signifying a state of unencumbered collateral. Our assumption pertains to the natural logarithm of the exchange rate $\frac{e(t)}{e_0}$, characterized by a Brownian motion possessing a mean of zero and an unspecified standard deviation $\sigma > 0$. Consequently, the transformation $\frac{1}{\sigma} \ln \frac{e(t)}{e_0}$ yields a Brownian motion B_t, preserving a mean of zero and unit variance. This mathematical representation underpins the analytical exploration, capturing the exchange rate's unpredictable movements.

The liquidation of debt becomes feasible when its collateral ratio falls below a certain threshold, such as 1.5 or 1.45. Assuming no alterations in both debt and collateral subsequent to $t = 0$, and with parameters set as stability fee $f = 0$, collateral amount $a(t) = a(0) \equiv a_0$, and debt amount $d(t) = d(0) \equiv d_0$, a critical relationship emerges. Specifically, the occurrence of liquidation is contingent upon the existence of a time $t > 0$ for which the exchange rate $e(t)$ satisfies the equation:

$$e(t) = \frac{d_0}{a_0} \cdot r_{min} \equiv e_{min}. \tag{3.11}$$

This equation encapsulates the threshold at which the exchange rate aligns with the minimum required collateral ratio, marking the point of potential liquidation.

The event of liquidation or default corresponds to the first passage of the level x_{min} by the Brownian motion B_t, governed by the equation:

$$x_{min} = \frac{1}{\sigma} \ln \left(\frac{d_0 \cdot r_{min}}{a_0 \cdot e_0} \right). \tag{3.12}$$

In the case where $C < 0$, let $T_C = \inf\{t > 0 : B_t = C\}$ denote the first passage time. Then, for a given $T > 0$, the probability of the first passage occurring before T is given by:

$$P\left(T_{x_{min}} < T\right) = \int_0^T \frac{-x_{min}}{\sqrt{2\pi s^3}} e^{-\frac{x_{min}^2}{2s}} ds, \tag{3.13}$$

leading to the distribution density $p_C(t) = \frac{C}{\sqrt{2\pi t^3}} e^{-\frac{C^2}{2t}}$.

These mathematical expressions capture the probabilistic nature of liquidation or default events in the context of the given Brownian motion model.

Within our dataset, the stability fee is not consistently set at a constant zero value but rather adheres to $f > 0$. If we consider the case $f > 0$, then the relationship $d(t) = d_0 \cdot e^{ft}$ holds. A default event occurs when there exists a time $t > 0$ satisfying the condition:

$$e(t) = \frac{d_0 \cdot e^{ft}}{a_0} \cdot r_{min} \equiv e_{min}(t). \tag{3.14}$$

This default occurrence is akin to the passage of the level by Brownian motion as

$$x_{min}(t) = \frac{1}{\sigma} \ln \left(\frac{d_0 \cdot r_{min}}{a_0 \cdot e_0} \right) + ft = x_{min} + ft. \tag{3.15}$$

Defining $T_{C,f} = \inf\{t > 0 : B_t = C + ft\}$, under the conditions $C < 0$ and $f > 0$, the probability $P\left(T_{x_{min},f} < T\right)$ signifies the likelihood of the event $T_{x_{min},f}$ occurring before time T. This probability is represented by the integral:

$$P\left(T_{x_{min},f} < T\right) = \int_0^T \frac{|x_{min} + fs|}{\sqrt{2\pi s^3}} e^{-\frac{(x_{min}+fs)^2}{2s}} ds. \tag{3.16}$$

Within the system, a multitude of users are present. Let us consider a scenario in which the second user experiences the first passage of the level $y_{min}(t) = y_{min} + ft$, subject to the condition $y_{min} \leq x_{min}$. The probability associated with the simultaneous occurrence of the first passage at bothx_{min} and y_{min} can be symbolized as

$$P\left(T_{x_{min},f} < T; T_{y_{min},f} < T\right) = P\left(T_{y_{min},f} < T\right).$$

Let $I(A)$ and $I(B)$ represent the indicator of the event $T_{x_{min},f} < T$ and $T_{y_{min},f} < T$ respectively. This implies that $I(A) = 1$ and $I(B) = 1$ if $T_{x_{min},f} < T$ and $T_{y_{min},f} < T$, and conversely $I(A) = 0$ and $I(B) = 0$ otherwise. The following is a mathematical expression for expectation and covariance:

$$\begin{aligned}
\mathbf{E}I(A) &= P\left(T_{x_{min},f} < T\right) = \psi\left(x_{min}\right)\\
\mathbf{E}I(B) &= P\left(T_{y_{min},f} < T\right) = \psi\left(y_{min}\right)\\
\text{cov}(I(A), I(B)) &= \mathbf{E}\left(I(A)I(B)\right) - \mathbf{E}I(A)\mathbf{E}I(B)\\
&= 1 \cdot P\left(T_{x_{min},f} < T; T_{y_{min},f} < T\right) - \mathbf{E}I(A) \cdot \mathbf{E}I(B) \qquad (3.17)\\
&= \psi\left(y_{min}\right) - \psi\left(x_{min}\right) \cdot \psi\left(y_{min}\right)\\
&= \psi\left(y_{min}\right) \cdot \left(1 - \psi\left(x_{min}\right)\right)\\
&= \psi\left(\min\left\{x_{min}, y_{min}\right\}\right) \cdot \left(1 - \psi\left(\max\left\{x_{min}, y_{min}\right\}\right)\right).
\end{aligned}$$

We can find PD by using a quad, SciPy's integration functions, which use an algorithm from the Fortran library QUADPACK that can integrate functions for specific time intervals [74].

Based on equation (3.16), when $x_{min} + fs$ approaches zero, the function's values exhibit a pulse-like behavior that tends to approximate 1.0. However, due to the distinct pulse shape compared to other values within the integration interval, caution is necessary as this method might not yield accurate integration results due to a numerical issue in the library. It is advisable to cross-verify the PD values using alternative mathematical programming tools, such as Mathematica [38], to ensure the accuracy and reliability of the outcomes.

3.3 Portfolio Sold-Out Problem

3.3.1 Problem Definition

Following the gathering of essential data and parameters, our attention moves to the application of traditional Markowitz theory to a major challenge - the portfolio sold-out problem. This problem arises in the context of asset management when various assets have different risk profiles and expected returns. Our objective is to systematically package these assets into portfolios, maximizing their diversity to provide potential stakeholders with valuable investment options.

The challenging nature of this work is further complicated by the presence of intrinsic uncertainty, which must remain within defined parameters. This adds complexity to the portfolio optimization process, as we seek to combine assets into packages that maximize convergence while staying within specified uncertainty constraints. It is worth noting that not all assets will fit effectively into these optimized packages. Some assets fall beyond the area of inclusion due to their risk profiles and require separate maintenance, incurring related expenses.

Let us commence by introducing essential terms that underpin our analysis. We denote N as the number of assets under consideration, with $N \geq 1$, and symbolize the expected returns of each asset as $\vec{A}$, defined as $\vec{A} = (A_1, \ldots, A_N)^T$. The random variables $\xi_1, \ldots, \xi_N$ describe the uncertainty per unit of return which $\mathbf{E}\xi_n = 1$, where $n \in \bar{N}$.

Additionally, we introduce the vector $\vec{c}$ to denote the allocation of investment proportions to each asset package, subject to the constraint $\|\vec{c}\|_1 = 1$. Here, the first norm ($\| * \|_1$) signifies the L1 norm, also known as the Manhattan norm, calculated by summing the absolute values of the vector's components.

The covariance matrix $\mathbf{K} \in \mathbb{R}^{N \times N}$, characterizing the dependence among individual assets, assumes the form of a positive semi-definite matrix. The determination of covariance equations $\mathbf{K}$, coupled with the investment distribution for $i = 1, \ldots, N$, is facilitated by the utilization of Equation (3.17). This results in the following explicit formulation as

$$\mathbf{K} = \begin{bmatrix} K_{11} & \ldots & K_{1N} \\ \vdots & \ddots & \vdots \\ K_{N1} & \ldots & K_{NN} \end{bmatrix},$$

where $K_{ij} = \text{cov}(\xi_i, \xi_j)$ corresponds to the covariance associated with assets i, j where $i, j \in \bar{N}$.

Definition 1. The packages comprising the portfolio $(\vec{A}, \mathbf{K})$ are represented by $\vec{c}$, where $\vec{c} \in \mathbb{R}^N$, subject to the condition $\overrightarrow{0}_N \leq \vec{c} \leq \vec{A}$ and $\mathbf{E}\vec{c}^T\vec{\xi} = 1$.

Definition 2. Let $\mathbf{K} = (K_{ij})_{i,j \in \bar{N}}$ symbolize the asset covariance matrix. The variance of the package $\vec{c}$ is given by

$$\mathrm{V}(\vec{c}) = \vec{c}^T \mathbf{K} \vec{c}. \tag{3.18}$$

Definition 3. A collection of M packages

$$\mathbf{C}_M = (\vec{c}_1 | \dots | \vec{c}_M) \in \mathbb{R}^{N \times M} \tag{3.19}$$

is considered a tokenization of the portfolio $(\vec{A}, \mathbf{K})$ if $\sum_{m=1}^{M} \vec{c}_m \leq \vec{A}$.

Definition 4. The variance (V) of tokenization $\mathbf{C}_M$ is the maximum variance of its packages:

$$\mathrm{V}(\mathbf{C}_M) = \max_{m \in \bar{M}} \mathrm{V}(\vec{c}_m). \tag{3.20}$$

Theorem 1. Given a portfolio $(\vec{A}, \mathbf{K})$ and a variance $\sigma^2 > 0$, the portfolio sold-out problem can be formulated as

$$\begin{aligned} & M \to \max_{M, \mathbf{C}_M}, \\ \text{subject to} \quad & \mathrm{V}(\mathbf{C}_M) \leq \sigma^2 \\ & 0 \leq \sum_{m=1}^{M} \vec{c}_m \leq \vec{A}. \end{aligned} \tag{3.21}$$

These definitions were covered within the study focusing on the tokenization of assets using blockchain technology [23].

3.3.2 Portfolio Sold-Out Problem Special Cases

The portfolio sold-out problem can be dissected into 6 distinct cases, each characterized by a unique covariance matrix and package types.

1. **Discrete Homogeneous Case**: In this scenario, the covariance matrix (**K**) is a diagonal matrix with uniform elements (σ_0^2) representing the squared variances of the individual assets. The matrix $\mathbf{D}_M$ used to define the packages is a Boolean matrix, indicating discrete packages of assets. This case assumes discrete units of assets for trading and focuses on the impact of variances in creating packages for selling.

2. **Continuous Homogeneous Case**: Similar to the discrete homogeneous case, the covariance matrix (**K**) is a diagonal matrix with uniform elements (σ_0^2), but here the matrix $\mathbf{C}_M$ is real rather than Boolean. This implies continuous units of assets for packaging. The focus remains on variances, but now with continuous packages for selling.

3. **Discrete Independent Case**: In this case, the covariance matrix (**K**) is constructed so that all off-diagonal elements are zero, indicating no correlation between asset returns. The matrix $\mathbf{D}_M$ for packages is again Boolean. This scenario isolates the impact of uncorrelated assets on creating packages for selling discretely.

4. **Continuous Independent Case**: Similar to the discrete independent case, but here the matrix $\mathbf{C}_M$ for packages is real, indicating continuous packages. The absence of correlations between assets is considered in the context of continuous packaging for selling.

5. **Discrete General Case**: In this scenario, there are no restrictions on the covariance

matrix ($\mathbf{K}$), allowing for any form of covariance between asset returns. The matrix $\mathbf{D}_M$ for packages remains Boolean. This case delves into the interplay between covariance patterns and discrete packaging for selling.

6. **Continuous General Case**: Lastly, the continuous general case is similar to the discrete general case, but with the matrix $\mathbf{C}_M$ for packages being real-valued, signifying continuous packages. The exploration of covariance's influence on packaging for selling extends to the realm of continuous units.

These cases can be represented in Table 3.1 and offer a comprehensive understanding of the portfolio sold-out problem across various covariance structures and packaging types, shedding light on the intricacies of optimizing selling strategies for different financial scenarios.

Table 3.1: Special cases of portfolio sold-out problem.

Cases	Discrete	Continuous
Homogeneous	$\mathbf{K} = \sigma_0^2 \mathbf{I}_N$ $\mathbf{D}_M$ is Boolean matrix.	$\mathbf{K} = \sigma_0^2 \mathbf{I}_N$ $\mathbf{C}_M$ is real matrix.
Independent	$K_{ij} = 0$ for $i \neq j$ $\mathbf{D}_M$ is Boolean matrix.	$K_{ij} = 0$ for $i \neq j$ $\mathbf{C}_M$ is real matrix.
General	any $\mathbf{K}$ is allowed $\mathbf{D}_M$ is Boolean matrix.	any $\mathbf{K}$ is allowed $\mathbf{C}_M$ is real matrix.

For the cases of discrete homogeneous and continuous homogeneous scenarios, solutions were obtained using Discrete and Continuous Algorithms, as referenced in Algorithms 1 and 2 respectively. Our current objective is to extend these findings to encompass the continuous general case and discrete independent case.

3.4 Evaluation Criteria

To assess the effectiveness of the methods of solving portfolio sold-out special cases within the specified risk threshold, we employ a metric known as the tokenized fraction. This metric is defined as:

$$\text{tokenized fraction} = \frac{\text{total amount of tokenized assets}}{\text{total amount of initial assets}}.$$

In essence, the tokenized fraction quantifies the proportion of total assets that can be transformed into tokenized packages relative to the initial assets. A higher tokenized fraction is indicative of a more favorable outcome.

This metric is particularly valuable as it allows us to gauge the efficiency of the methods in terms of asset utilization. The larger the tokenized fraction, the better the methods are at converting the available assets into tokenized packages while adhering to the defined risk threshold. This measurement becomes crucial in evaluating the overall effectiveness and performance of the applied methodologies within the risk constraints established.

Chapter 4
Results and Discussion

This chapter serves as the results and discussion section of the thesis, encompassing 3 main segments: theorem results, the application of theorems to the dataset, and the subsequent discussion of these outcomes. The first part presents theorems for proven cases, which serve as the foundation for addressing portfolio sold-out special cases. The second part demonstrates how these methodologies were applied to real-world data. Finally, the chapter delves into a comprehensive discussion of the results.

4.1 Theorem Results

Since the portfolio sold-out problem can be formulated as

$$\begin{aligned} & M \to \max_{M, \mathbf{C}_M}, \\ \text{subject to} \quad & \max_{m \in \bar{M}} \mathrm{V}\left(\vec{c}_m\right) \leq \sigma^2, \\ & 0 \leq \sum_{m=1}^{M} \vec{c}_m \leq \vec{A}. \end{aligned} \tag{4.1}$$

4.1.1 Continuous General Case

Theorem 2. The optimal portfolio sold-out problem for the continuous general case (CGOPSO) is polynomially reducible to second-order cone programming (SOCP), allowing for optimal numerical solutions.

Proof. In the context of the continuous general case, let's consider an optimal solution matrix denoted as $\mathbf{C}_M \in \mathbb{R}^{N \times M}$. Given the capabilities of blockchain technology for tokenization, we can introduce a transformed matrix $\bar{\mathbf{C}}_M$. In this transformed matrix, each column vector $\vec{c}_1, \vec{c}_2, \ldots, \vec{c}_M$ is identical, represented as $\bar{\vec{c}}$, where $\bar{\vec{c}} = \frac{1}{M} \sum_{m=1}^{M} \vec{c}_m$. Consequently, the matrix $\bar{\mathbf{C}}_M$, which can be obtained by replicating $\bar{\vec{c}}$ across all columns as $\bar{\mathbf{C}}_M = (\bar{\vec{c}}| \ldots |\bar{\vec{c}})$, becomes the optimal solution.

Now, consider a vector $\vec{a} \in \mathbb{R}^{N \times 1}$ defined as $\vec{a} = M\bar{\vec{c}}$. If this vector $\vec{a}$ satisfies the constraints, it allows us to create M packages, each sharing similar components, thereby satisfying the original problem requirements.

Let's consider $\vec{a} = M\bar{\vec{c}}$. We can express the L1 norm as follows: $\|\vec{a}\|_1 = M\|\bar{\vec{c}}\|_1$. Additionally, we are aware that $\mathbf{E}\bar{\vec{c}}^T\vec{\xi} = \|\bar{\vec{c}}\|_1 = 1$. Therefore, we can derive the relationship:

$$M = \|\vec{a}\|_1. \tag{4.2}$$

Next, we can calculate the variance of the packages:

$$\begin{aligned}
\mathrm{V}\left(\vec{\bar{c}}\right) &= \vec{\bar{c}}^T \mathbf{K} \vec{\bar{c}} \\
&= \left(\frac{1}{M}\vec{a}\right)^T \mathbf{K} \left(\frac{1}{M}\vec{a}\right) \\
&= \frac{1}{M^2}\vec{a}^T \mathbf{K}\vec{a} \\
&= \frac{1}{\|a\|_1^2}\vec{a}^T \mathbf{K}\vec{a} \quad \leq \sigma^2.
\end{aligned} \tag{4.3}$$

This allows us to reformulate the variance of the packages as:

$$\mathrm{V}\left(\vec{\bar{c}}\right) = \vec{a}^T \mathbf{K}\vec{a} \leq \sigma^2 \|a\|_1^2. \tag{4.4}$$

The assets allocated into the packages total less than the initial asset pool. Thus, we have:

$$\sum_{m=1}^{M} \vec{c}_m = M\vec{\bar{c}} = \vec{a} \leq \vec{A}. \tag{4.5}$$

By combining Equations (4.2), (4.4), and (4.5), we can reformulate the original portfolio sold-out problem and perform it as :

$$\begin{aligned}
& \|\vec{a}\|_1 \to \max, \\
\text{subject to} \quad & \vec{a}^T \mathbf{K}\vec{a} \leq \sigma^2 \|\vec{a}\|_1^2, \\
& \vec{0} \leq \vec{a} \leq \vec{A}.
\end{aligned} \tag{4.6}$$

This reformulation simplifies the complexity of the problem and offers a more transparent approach to solving the portfolio sold-out problem. To tackle this challenge, we employ two optimization techniques: a Metaheuristic Algorithm (MA) as a baseline and Second-Order Cone Programming (SOCP).

For the MA, we can leverage the MEALPY library [71] in Python. In parallel, when addressing the portfolio sold-out problem using SOCP, we can derive constraints on the variance before applying them to the SOCP framework.

Let **K** represent a symmetric positive semi-definite covariance matrix, which can be decomposed into the form $\mathbf{K} = LL^T$ using Singular Value Decomposition (SVD), as follows:

$$\mathbf{K} = U\Sigma U^T = U\Sigma^{\frac{1}{2}}\Sigma^{\frac{1}{2}}U^T. \tag{4.7}$$

By defining $L = U\Sigma^{\frac{1}{2}}$, we establish $\mathbf{K} = LL^T$. Consequently, the variance constraint can be reformulated as:

$$\vec{a}^T K\vec{a} = \vec{a}^T LL^T\vec{a} = \left(L^T\vec{a}\right)^T \left(L^T\vec{a}\right) = \|L^T\vec{a}\|_2^2. \tag{4.8}$$

With this reformulation in place, we can directly address the problem using SOCP, implementing it with the CVXPY library [26] in Python.

4.1.2 Discrete Independent Case

Theorem 3. The optimal portfolio sold-out problem for the discrete independent case (DIOPSO) lies in the NP-hard class.

Proof. To establish this, we will reduce the half-partition problem to it. The partition problem is NP-complete and is formulated as follows: given a set of positive integers $a_1, \ldots, a_N$, find out whether there is a subset of indexes $I \subset \{1, \dots, N\}$ such that

$$\sum_{n \in I} a_n = \frac{S}{2}; \quad S \equiv \sum_{n=1}^{N} a_n.$$

Let us polynomially reduce the partitioning problem to a DIOPSO problem. Given data from the partitioning problem, construct a tuple $(\vec{A}, k, \mathbf{K}, \sigma^2)$ serving as the input for DIOPSO problem as follows:

- $\vec{A} = (1, \cdots, 1)^T$ where $\dim \left(\vec{A}\right) = 2N$
- $k = N$
- $\mathbf{K}$ is diagonal matrix with $(a_1, \dots, a_N, \underbrace{0, \cdots, 0}_{N})$ on the diagonal
- $\sigma^2 = \frac{S}{2}$.

The original discrete independent problem can be reduced to the equivalent

$$(M, \mathbf{D}_M) = \arg \max_{M, \mathbf{D}_M} M,$$

where $\mathbf{D}_M = \left(\vec{d}_1 | \dots | \vec{d}_M\right) \in \{0, 1\}^{2N \times M}$ and $M \in \{0, 1, 2\}$ ($M \leq 2$ as $\|\vec{d}_m\|_1 = N$, $\|\vec{A}\|_1 = 2N$ and $M \cdot \|\vec{d}_m\|_1 \leq \|\vec{A}\|_1$) is the number of package set for portfolio $(\vec{A}, \mathbf{K})$ if $\sum_{m=1}^{M} \vec{d}_m \leq \vec{A}$ and $\forall m \in \{1, \dots, M\}$:

$$\sum_{i=1}^{N} d_{m,i}^2 \cdot a_i = \sum_{i=1}^{N} d_{m,i} \cdot a_i \leq \frac{S}{2}, \tag{4.9}$$

where $d_{m,i}$ is the ith component of the vector $\vec{d}_m$.

If the answer for discrete independent optimal portfolio sold-out problem is $M = 2$, then the components of $\vec{A}$ are divided into 2 groups, $\vec{d}_1$ and $\vec{d}_2$:

(a) $\vec{d}_1$ and $\vec{d}_2$ meet the constraint (4.9)

(b) $\vec{A} = \vec{d}_1 + \vec{d}_2$

(c) $\vec{d}_1, \vec{d}_2 \in \{0, 1\}^{2N}$.

Note that

$$\sum_{i=1}^{N} d_{1,i} \cdot a_i + \sum_{i=1}^{N} d_{2,i} \cdot a_i = \sum_{i=1}^{N} a_i = S.$$

So in (4.9) we get equality, but no inequality:

$$\begin{cases} \sum_{i=1}^{N} d_{1,i} \cdot a_i = S/2 \\ \sum_{i=1}^{N} d_{2,i} \cdot a_i = S/2. \end{cases}$$

Both vectors $\vec{d}_1$ and $\vec{d}_2$ give solutions for the partition problem, where the first N components are indicators of subset elements I.

This reduction is carried out in polynomial time. Hence, these subsets could be returned as a solution to the partition problem in polynomial time.

Furthermore, if the partition problem has a solution, the corresponding DIOPSO has a solution $(M, \mathbf{D}_M)$ with $M = 2$. For example, given a solution subset of indexes $I \subset \{1, \ldots, N\}$, we can construct

$$\vec{d}_1 = ([1 \in I], \ldots, [N \in I], \underbrace{0, \cdots, 0}_{|I|}, \underbrace{1, \cdots, 1}_{N-|I|})^T$$
$$\vec{d}_2 = ([1 \notin I], \ldots, [N \notin I], \underbrace{1, \cdots, 1}_{|I|}, \underbrace{0, \cdots, 0}_{N-|I|})^T,$$

where $[C]$ is an indicator of the event C ($[C] = 1$, if C is true, $[C] = 0$ otherwise), $|C|$ is the cardinality of the set C. Indeed,

(a) as I is the solution of the partition problem:

$$\begin{cases} \sum_{i=1}^{N} d_{1,i} \cdot a_i = S/2 \\ \sum_{i=1}^{N} d_{2,i} \cdot a_i = S/2 \end{cases}$$

(b) $\vec{d}_1 + \vec{d}_2 = (1, \cdots, 1)^T = \vec{A}$

(c) $\vec{d}_1, \vec{d}_2 \in \{0, 1\}^{2N}$.

NP-hardness is proved.

Algorithm for Discrete Independent Case

Despite the knowledge that the discrete independent case falls within the NP-hard class, it remains feasible to address the problem effectively, especially when dealing with a limited number of initial assets, as illustrated in Figure 4.1.

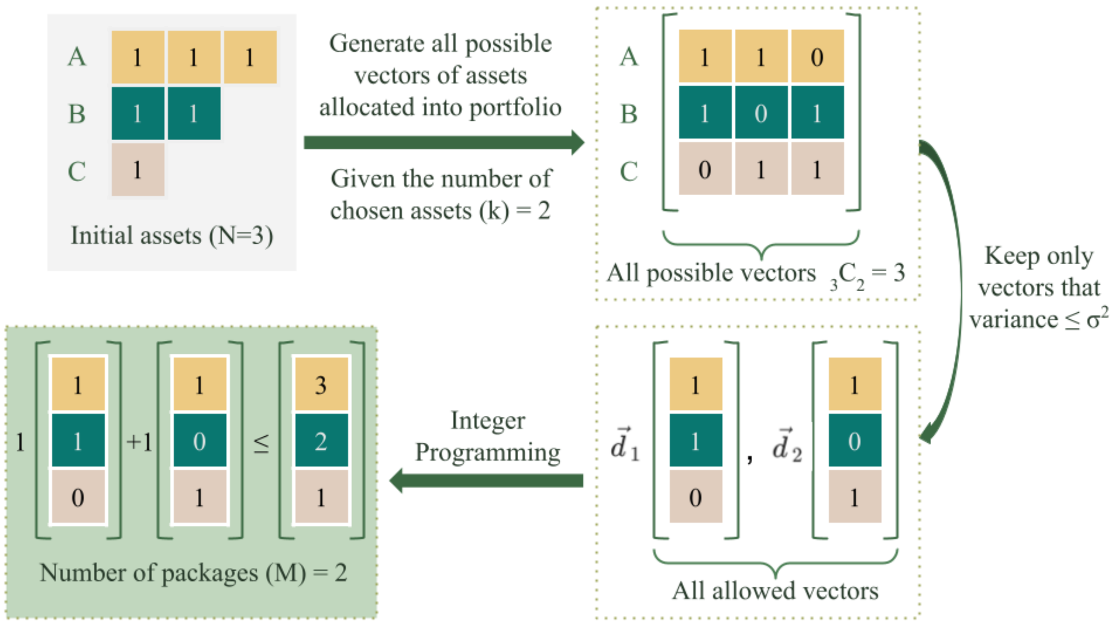
A
B
C
Initial assets (N=3)
Generate all possible vectors of assets allocated into portfolio
Given the number of chosen assets (k) = 2
All possible vectors ${}_3C_2 = 3$
Keep only vectors that variance ≤ σ²
All allowed vectors
Integer Programming
Number of packages (M) = 2

Figure 4.1: Discrete independent method.

Furthermore, the same methodology can be extended to tackle the discrete general case. The algorithm for addressing both discrete independent and general cases is provided in Algorithm 3.

Algorithm 3: Algorithm for discrete independent and general cases

```
Input: $\vec{A}, K, k, \sigma^2$
Output: $M, list\_\alpha, allowed\_d, D_M$
// Generate vectors that satisfy the variance constraint
Function GenerateVectors($\vec{A}, k, \sigma^2$):
    $N \leftarrow$ length of $\vec{A}$
    Create an array $items$ from 0 to $N-1$
    for $idx$ in combinations($items, k$) do
        Create $\vec{v}$ of zeros of length $N$
        Set elements in $\vec{v}$ at indices in $idx$ to 1
        if $(\vec{v}^T K \vec{v})/(k^2) \leq \sigma^2$ then
            yield $\vec{v}$
// Optimize the package
Function OptimizePackages($\vec{A}, K, k, \sigma^2$):
    $N \leftarrow$ length of $\vec{A}$
    if $k > N$ then
        return $0, [0], [0]$
    else
        $allowed\_d \leftarrow$ using GenerateVectors
        $n\_allowed\_d \leftarrow$ length of $allowed\_d$
        if $n\_allowed\_d = 0$ then
            return $0, [0], [0]$
        else
            Define optimization variables and problem
            Solve the optimization problem with Integer Programming
            $M \leftarrow$ objective value
            $list\_\alpha \leftarrow$ variable values
            return $M, list\_\alpha, allowed\_d$
// Construct a matrix $D_M$
Function OptimalPackages($M, list\_\alpha, allowed\_d$):
    if $M = 0$ then
        $D_M \leftarrow [0]$
    else
        Initialize empty list $all\_matrices$
        for $i, array$ in enumerate($allowed\_d$) do
            $num\_columns \leftarrow list\_\alpha[i]$
            Create a matrix $matrix$ by repeating $array$ as columns
            Append $matrix$ to $all\_matrices$
        if $M = 1$ then
            $D_M \leftarrow all\_matrices[0]$
        else
            $D_M \leftarrow all\_matrices[0]$
            for $idx, mat$ in enumerate($all\_matrices[1:]$) do
                Concatenate $mat$ with $D_M$ along columns;
        return $D_M$
```

4.1.3 Solutions for Portfolio Sold-Out Problem Special Cases

Table 3.1 provides a clear distinction between the covariance matrix structures of the independent and general cases. Notably, the independent case exhibits a simpler covariance structure compared to the general case. In the independent case, covariance values take the form of $K_{ij} = 0$ for $i \neq j$, making it less complex.

Having successfully addressed the continuous general case, it's worth emphasizing that the methodology used can also be applied to the **continuous independent case**. This case is inherently more straightforward due to its less intricate covariance matrix structure.

Furthermore, considering that solving the discrete independent case is proven to be an NP-hard problem, it logically follows that the **discrete general case**, characterized by a more complex covariance matrix structure, also falls within the NP-hard complexity class. This increased complexity poses a significant challenge for finding straightforward solutions. Importantly, solutions derived from NP-hard scenarios inherit the complexities of their continuous counterparts. The comprehensive set of solutions is summarized in Table 4.1 below.

Table 4.1: Solutions for portfolio sold-out problem special cases.

Cases	Discrete	Continuous
Homogeneous	optimal explicit solution	optimal explicit solution
Independent	NP-hard	optimal numerical solution
General	NP-hard	optimal numerical solution

4.2 Application of Theorems to the Dataset

After completing the data preprocessing for the MakerDAO dataset, we have a dataset that contains daily transaction records. Among the various data points, there are key components of significance: borrowers, their corresponding debt amounts, and the associated risks, which are quantified as the probability of default, as visualized in Figure 4.2.

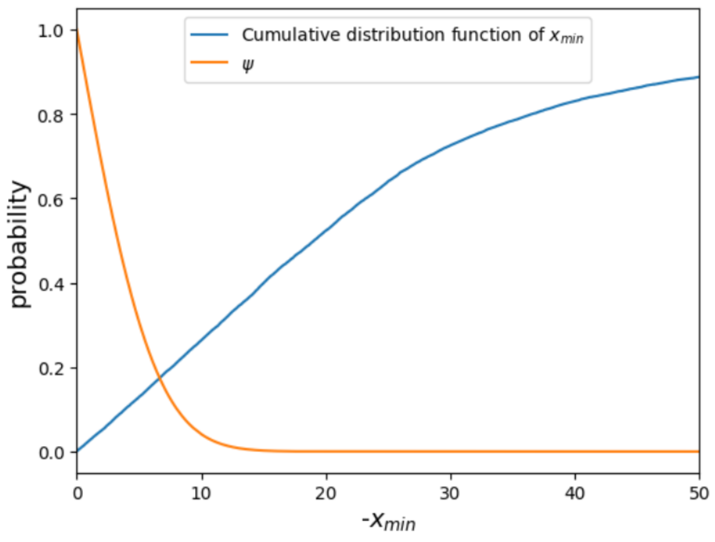

Figure 4.2: Probability and the level of default.

One crucial parameter in our analysis is the level of default, denoted as x_{min}, which can be calculated using the formula presented in Equation (3.15). It's worth noting that the exchange rate between Ethereum (ETH) and DAI (the stablecoin used within the MakerDAO system) plays a pivotal role in determining this parameter. Specifically, we observe that the level of default, x_{min}, tends to increase when the exchange rate (ETH/DAI) decreases. This means that as the exchange rate drops, the risk of default becomes more pronounced and escalates.

The graphical representation in Figure 4.2 visually illustrates this relationship. As the value of x_{min}, representing the default threshold, increases, we observe a corresponding rise in the probability of default. In essence, this graph underscores the critical interplay between the exchange rate and default risk within the MakerDAO ecosystem, highlighting the importance of monitoring and understanding these dynamics for risk management and decision-making.

The covariance matrices used for modeling default correlations are derived from these probability of default values. Upon observation of the dataset, it becomes apparent that the covariance matrix $\mathbf{K}$ conforms to the general case, characterized by its positive semi-definite properties. Consequently, the portfolio optimization methods to be applied to this dataset are limited to the continuous and discrete general methods.

4.2.1 Applying the Continuous General Method

We apply the continuous general method to the MakerDAO dataset by employing two distinct optimization techniques:

- **MA** as the baseline with MEALPY [71] library in Python utilizing Slime Mould Algorithm (SMA).
- **SOCP** with CVXPY [26] library in python utilizing Mosek solver [16].

We have generated overall tokenized fractions as detailed in Table 4.3. Notably, the tokenized fraction achieved using SOCP stands at an impressive 65.69%, outperforming the MA, which achieves a tokenized fraction of 65.20%.

Table 4.2: Tokenized fractions for continuous general method.

Optimization methods	Tokenized fraction (%)
Baseline: Metaheuristic Algorithm (MEALPY)	65.20
Second-Order Cone Programming (CVXPY)	**65.69**

When we delve into the daily transaction data and sort the tokenized fractions, the resulting trends are visually depicted in Figure 4.3. This graphical representation illustrates that SOCP consistently outperforms the MA as the baseline method.

For a more comprehensive understanding, Figure 4.4 offers a direct comparison of tokenized fractions between SOCP and MA. The comparison unmistakably demonstrates that SOCP consistently achieves higher tokenized fractions.

Graph 4.5 provides a visual representation of the tokenized fractions achieved by the two optimization methods across various levels of risk denoted by σ. Notably, the SOCP method consistently outperforms the MA across all levels of risk (σ).

In summary, the findings strongly indicate that SOCP stands as the preferred choice when compared to the MA. This preference is substantiated by its ability to consistently yield higher tokenized fractions. This outcome underscores the effectiveness of SOCP in optimizing the arrangement of assets within the MakerDAO dataset while maintaining compliance with the predefined risk threshold.

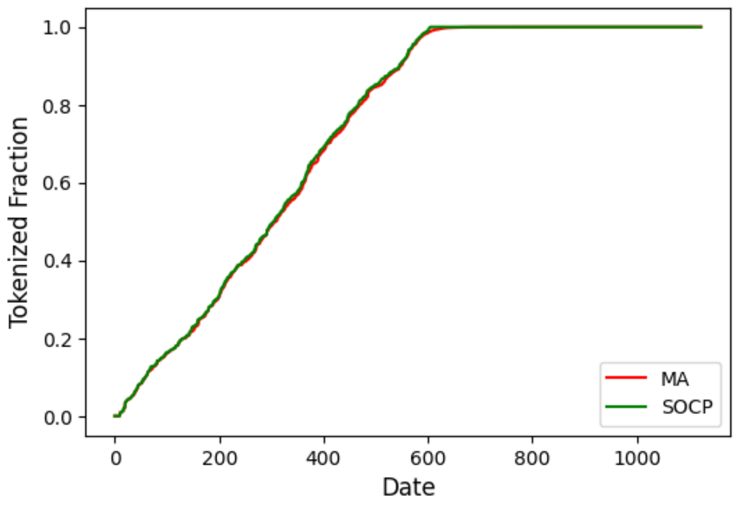

Figure 4.3: Arranging the daily tokenized fractions in ascending order for the continuous general method.

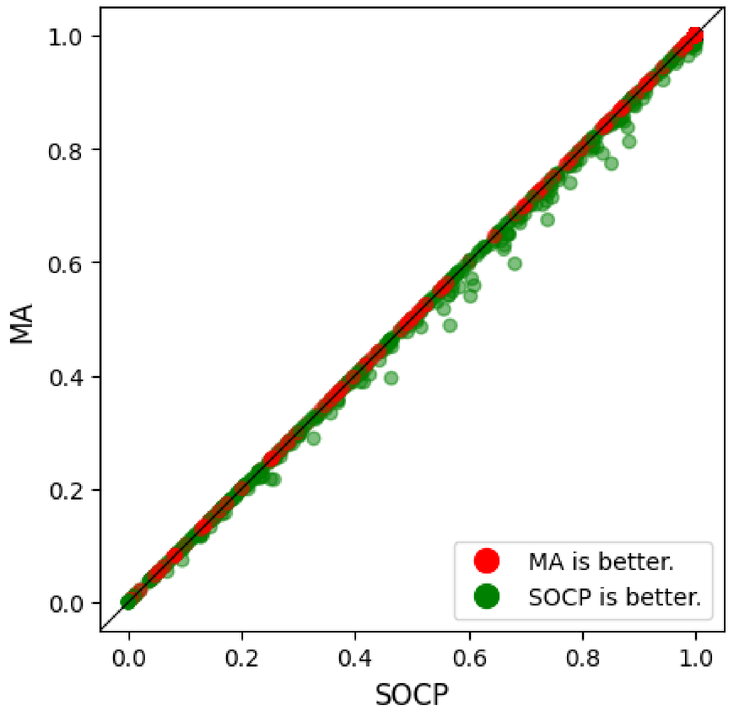

Figure 4.4: Comparing fraction of assets that can be organized within a portfolio using MakerDAO data with different optimization methods.

4.2.2 Applying the Discrete General Method

Because the discrete general portfolio sold-out problem falls into the NP-hard class, we encounter complexity issues when dealing with a high number of daily assets. Therefore, we employ the discrete general method for all transactions where the number of assets (N) is lower than or equal to 20. This choice is motivated by the fact that such transactions encompass approximately 74.58% of the entire dataset, as illustrated in Figure 4.6.

After implementing the discrete general method on the data, Figure 4.7 illustrates the relationship between the number of chosen assets (k) and the fraction of allowed $\vec{d}$ vectors. This fraction represents the proportion of $\vec{d}$ vectors for which their variances are less than a specified

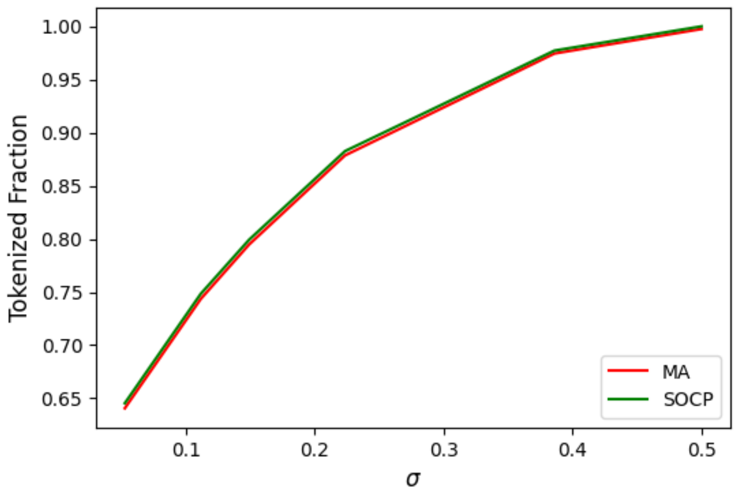

Figure 4.5: Tokenized fractions for continuous general method across varying levels of σ.

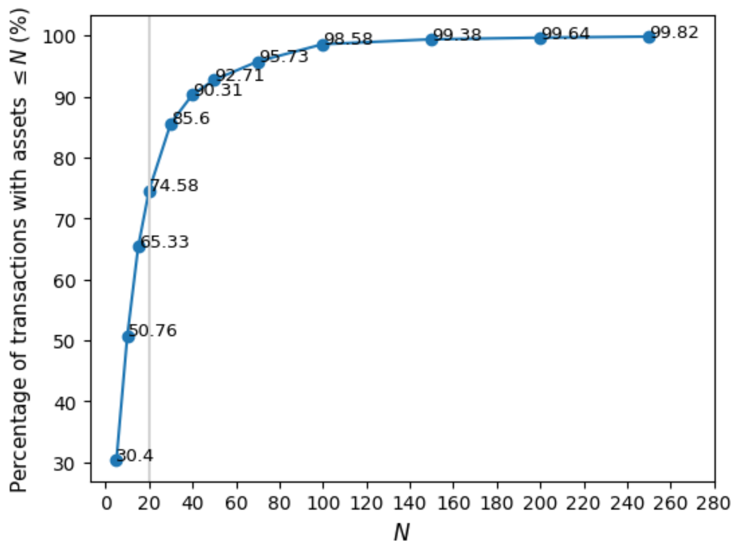

Figure 4.6: Percentage of transactions with assets $\leq N$ (%).

value (σ^2). Notably, as we observe from the figure when k is relatively small, the fraction tends to be larger compared to when k is larger. This finding underscores the impact of asset selection on the proportion of allowable $\vec{d}$ vectors with variances below a given threshold.

Upon applying the complete discrete general algorithm to the dataset, we have computed tokenized fractions, as outlined comprehensively in Table 4.3. The arrangement of daily tokenized fractions in ascending order, corresponding to the discrete general method with various selections of assets (k), is presented in Figure 4.8. Remarkably, it is evident from the table and the figure that the tokenized fractions for smaller values of k outperform those for larger values of k. This observation underscores the relative performance of different asset selections in the analysis.

As illustrated in Figure 4.9, when we examine the relationship between the market concentration and the fraction, we observe consistently low correlations across various values of k. This suggests that the concentration of assets within the portfolio doesn't strongly influence the tokenized fraction, regardless of the number of chosen assets (k).

In contrast, when we analyze the correlations between the Gini coefficient and the fraction, particularly for portfolios with a smaller number of chosen assets (k), we uncover notably high correlations. This underscores the significance of the Gini coefficient as a portfolio characteristic, especially in cases where asset selection is limited. The strong correlations highlight the Gini

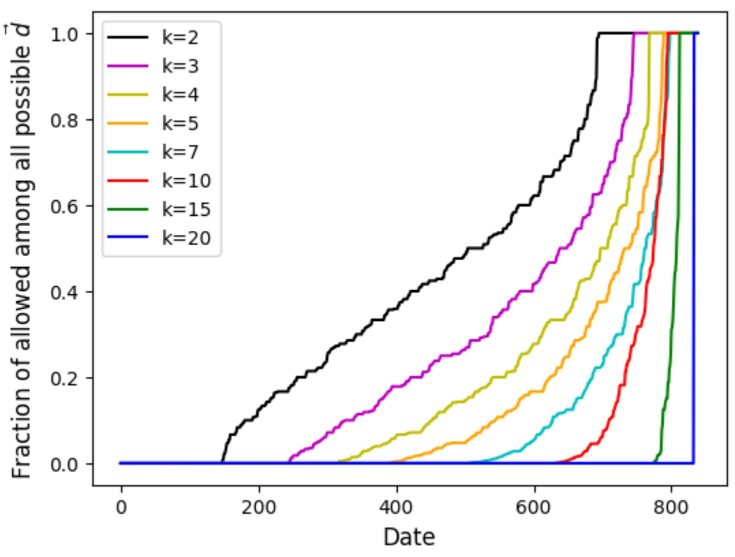

Figure 4.7: Fraction of allowed among all possible $\vec{d}$.

Table 4.3: The portion of assets that can be structured within a portfolio utilizing the discrete general method.

The number of chosen assets (k)	Tokenized fraction (%)
$k = 2$	28.03
$k = 3$	20.40
$k = 4$	14.93
$k = 5$	11.73
$k = 7$	7.17
$k = 10$	3.45
$k = 15$	0.78
$k = 20$	0.04

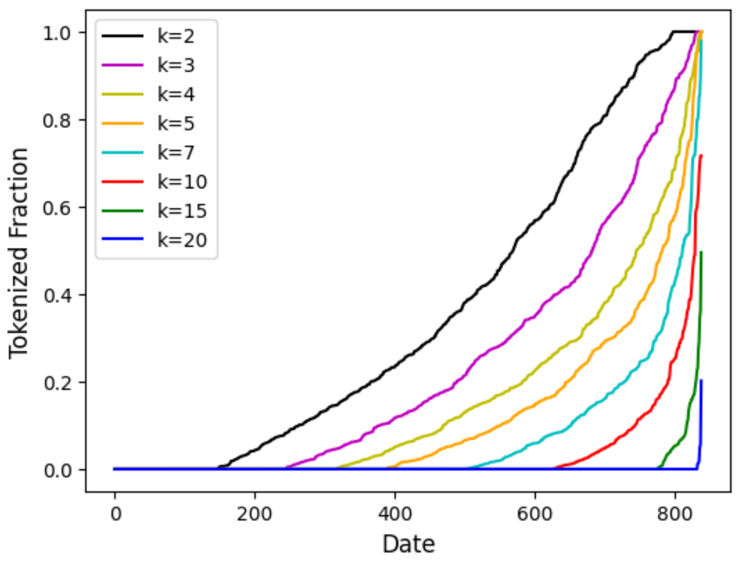

Figure 4.8: Arranging the daily tokenized fractions in ascending order for the discrete general method with different chosen assets (k).

coefficient's role in shaping the tokenized fraction in portfolios with fewer selected assets.

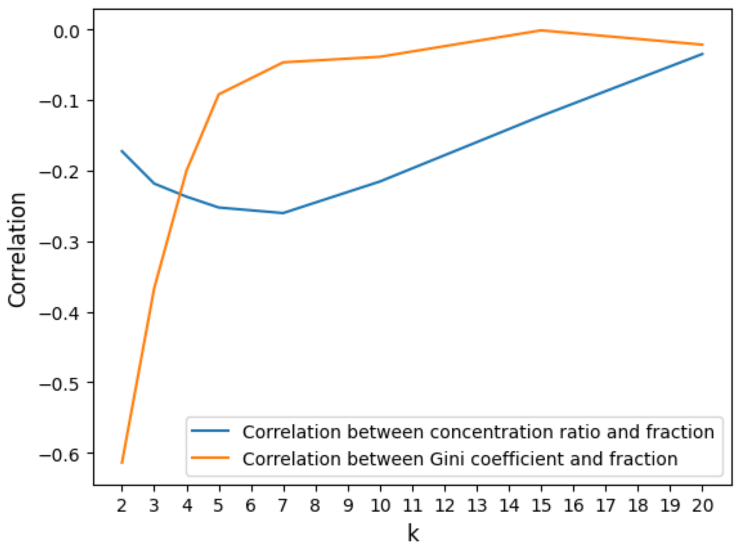

Figure 4.9: Correlation between market concentration and fraction, as well as the correlation between the Gini coefficient with different chosen assets (k).

4.3 Discussion of Results

In the context of the research findings, we have unearthed several critical insights about optimization methodologies and their applications in portfolio management.

First and foremost, the SOCP method stands out as a stalwart technique for achieving optimal numerical solutions. This method particularly shines when applied to continuous and independent scenarios, showcasing its ability to adeptly handle problems involving real-valued variables. The outcomes it generates are characterized by their precision and reliability.

Expanding the perspective to the world of tokenized fractions within the continuous general case, we find that SOCP plays a pivotal role. It offers valuable upper bounds, a significant asset in the analysis toolkit. These bounds provide a foundation for direct comparisons with those produced by MA, allowing us to assess the effectiveness of SOCP in approximating tokenized fractions in this specific context.

However, it's essential to acknowledge that the situation isn't always smooth in portfolio optimization. As we venture into discrete scenarios, exemplified by discrete independent and general cases, we encounter the formidable challenge of NP-hardness. This inherent complexity stems from the discrete nature of the decision variables, rendering optimization endeavors considerably more complicated and time-consuming.

In the practical implementation of the discrete general method, a pivotal determinant is the number of chosen assets, which assumes a significant role in this context. Furthermore, it is noteworthy that when a smaller number of assets is selected, there exists a notable and strong correlation between the Gini coefficient and the tokenized fraction.

In sum, this research not only highlights the effectiveness of SOCP in optimizing portfolios but also underscores the challenges posed by discrete scenarios. Moreover, we emphasize the practical utility of tokenization algorithms in bridging the gap between theory and real-world application, ultimately contributing to more informed decision-making in the DeFi space.

Chapter 5
Conclusions

This research delves deep into the intricate domain of portfolio optimization within the realm of DeFi lending protocols, placing a specific emphasis on the intricate challenge known as the portfolio sold-out problem. This formidable challenge has been meticulously dissected into six distinct cases, thoughtfully detailed in Table 3.1. While the previous research has successfully tackled the discrete homogeneous and continuous homogeneous cases, this primary research objective now pivots towards resolving the remaining and more complex four cases: continuous general, continuous independent, discrete general, and discrete independent scenarios. These scenarios present unique covariance matrices and package characteristics, necessitating innovative solutions that can intricately align with the subtleties of real-world financial data. These endeavors are critical as they prepare us to address the more intricate and practical challenges that lie ahead.

Upon successfully addressing the continuous general and continuous independent cases, the key findings illuminate the exceptional effectiveness of the second-order cone programming (SOCP) method in optimizing portfolios. SOCP emerges as a robust and precise technique, consistently delivering optimal numerical solutions that underscore its reliability and efficacy.

However, when it comes to discrete scenarios, it is essential to acknowledge the inherent complexity inherent in these cases. This complexity is vividly illustrated in both the discrete independent and general cases, where we confront the formidable challenge of NP-hardness. This complexity arises from the discrete nature of decision variables, significantly increasing the time and effort required for optimization endeavors in these contexts.

Another pivotal objective of this research is the empirical application, utilizing real loan data sourced from the MakerDAO dataset. As we transition from the theoretical formulation of the portfolio sold-out problem to the practical development of solutions, we immerse this research in the realm of real-world data to assess the tangible efficiency enhancements achieved through the framework. The central focus revolves around the evaluation of efficiency, a critical metric assessed through the examination of the tokenized fraction. This metric signifies the total value of assets successfully integrated into the packages relative to the initial pool of assets available for tokenization.

Given that the MakerDAO dataset's covariance matrix falls within the general case, characterized by positive semi-definiteness, we have limited the analysis to two specific methods: continuous and discrete general cases. In the practical implementation of methodologies for the continuous general case, the SOCP method plays a pivotal role. It provides valuable upper bounds that enable insightful comparisons with metaheuristic algorithms (MA), enhancing the understanding of the dataset's behavior.

In the discrete general case, we make a significant observation concerning the impact of the number of chosen assets on the tokenized fraction. Specifically, we find that when the number of chosen assets is relatively small, the tokenized fraction tends to be larger compared to scenarios with larger values of chosen assets. This discovery also highlights the substantial influence of asset selection on the proportion of allowable vectors as distinct packages with variances below a specified threshold. It also sheds light on the relative performance of various asset selections within the analysis.

Moreover, the research reveals intriguing patterns when examining the correlation between the Gini coefficient and the tokenized fraction. Notably, when the number of chosen assets is small, a strong correlation emerges. This suggests that there is an effect of inequality in the distribution of assets within the packages when a limited number of assets are selected. In contrast, the tokenized fraction is not greatly influenced by the portfolio's market concentration.

This research, along with prior investigations focused on discrete and continuous homogeneous cases as detailed in [25] and [23], now delivers a comprehensive solution to the portfolio sold-out problem. This solution encompasses both theoretical theorems and practical empirical applications. Our collective efforts have bridged the gaps in understanding this intricate problem, propelling the field of portfolio management within DeFi lending protocols forward and providing valuable insights for real-world financial applications.

Furthermore, our research's crucial discovery is that SOCP outperforms MA finds resonance in the findings of [24]. In this prior study, conducted on daily data from SP500 stocks spanning from 2005 to 2010, SOCP demonstrated superior performance when compared to quadratic programming (QP). This alignment of results across different research endeavors underscores the robustness and reliability of SOCP as an optimization technique in various financial contexts.

In the broader context of global research endeavors, our work not only provides valuable insights into the intricacies of portfolio optimization but also sheds light on the practical challenges embedded within the four distinct scenarios under examination. Furthermore, we emphasize the pivotal role assumed by tokenization algorithms in bridging the divide between theoretical concepts and their real-world applications.

The research represents a significant leap forward in the ongoing evolution of portfolio management strategies within DeFi lending protocols. It holds the promise of enhancing asset utilization and bolstering risk mitigation in the complex landscape of real-world finance. Moreover, it has the potential to emerge as a cutting-edge tool within the financial sector, offering fresh perspectives and competitive advantages in a rapidly changing industry.

This research primarily addresses the portfolio sold-out special cases using numerical methods, providing valuable insights and solutions within this specific scope. However, it is essential to acknowledge that the applicability of these findings to more complex and diverse real-world situations may be limited. Moreover, the dataset utilized in this research is derived from MakerDAO and is predominantly based on a single collateral type. This limitation means that the findings and conclusions drawn from this dataset may not fully encompass the complexities and dynamics of multi-collateral environments commonly encountered in decentralized finance (DeFi) systems. In conclusion, while this research contributes valuable insights and solutions to the portfolio sold-out special cases and offers a foundation for further exploration in DeFi, it is important to acknowledge these limitations when interpreting and applying the research's findings to broader financial contexts and evolving DeFi ecosystems.

Innovations

Within the rapidly developing DeFi industry, this research has enormous practical utility. The study helps to ensure the reliability and stability of DeFi lending systems by tackling the difficulty of preventing rapid asset depletion in unstable market situations. The results not only enhance investor and user confidence but also provide useful insights for risk management techniques.

The applications of this research involve startup potential and industrial adoption. A business might use its insights to provide particular risk management solutions to DeFi platforms, therefore setting a new standard for portfolio management. Furthermore, traditional financial institutions interested in decentralized models might use the study techniques to improve their asset management procedures. Overall, this study not only addresses an urgent issue but also sets the road for innovation and collaboration between the DeFi and traditional financial sectors.

Author Contribution

In the preparation of this thesis, the author undertook the responsibility for the majority of tasks, including project conception, literature review, data collection, data preprocessing, analysis, and the comprehensive composition of this document. Nevertheless, I would like to acknowledge a valuable collaboration with Sudarut Kasemsuk during the investigation of data within the MakerDAO structure. I conducted the data collection independently, meticulously gathering and preprocessing the dataset for further analysis. During the data preprocessing phase, I incorporated a mathematical theorem, initially proposed by Sudarut, into the research methodology to calculate the probability of default. It is important to emphasize that while I applied this theorem, I conducted the calculations and derivations autonomously, aligning them with the specific objectives of this study. Subsequently, all subsequent data preprocessing, analysis, and the entirety of the research were conducted by the author. It is worth noting that, as a measure of quality control, I engaged in extensive cross-checking of our respective datasets with Sudarut, ensuring the accuracy and consistency of the data. While our datasets were nearly identical, minor differences in data types emerged due to my subsequent work in preparing for the steps of portfolio optimization, reflecting the distinct requirements of that phase of the research.

Acknowledgements

I wish to extend my deepest appreciation to Yury Yanovich for the invaluable support and thoughtful guidance offered throughout this research journey. His unwavering dedication, insightful suggestions, and mentorship have played an important role in shaping the direction and outcomes of this thesis. His wealth of knowledge and commitment to excellence have been a constant source of inspiration, and I am profoundly grateful for the opportunity to benefit from his expertise. Yury's contributions have been pivotal in elevating the quality and rigor of this work, and his mentorship will undoubtedly continue to influence my academic and professional pursuits in the future.

Bibliography

[1] Aave. https://aave.com/. Accessed: 2023-10-09.

[2] Compound. https://compound.finance/. Accessed: 2023-10-09.

[3] Decentralized finance (defi). https://ethereum.org/en/defi/#ethereum-and-defi. Accessed: 2023-10-08.

[4] Kava. https://app.kava.io/home. Accessed: 2023-10-09.

[5] Makerdao. https://makerdao.com/en/. Accessed: 2023-10-09.

[6] Rates module: Maker protocol technical docs. https://docs.makerdao.com/smart-contract-modules/rates-module. Accessed: 2023-10-10.

[7] Second-order cone programming - university of california, berkeley. https://inst.eecs.berkeley.edu/~ee127/sp21/livebook/l_socp_main.html. Accessed: 2023-10-06.

[8] a. N. Shiryaev. *Essentials of stochastic finance*. 1999.

[9] Aggarwal, S., and Kumar, N. Chapter fifteen - blockchain 2.0: Smart contracts□□working model. In *The Blockchain Technology for Secure and Smart Applications across Industry Verticals*, S. Aggarwal, N. Kumar, and P. Raj, Eds., vol. 121 of *Advances in Computers*. Elsevier, 2021, pp. 301–322.

[10] Alem, D., Caunhye, A. M., and Moreno, A. Revisiting gini for equitable humanitarian logistics. *Socio-Economic Planning Sciences 82* (2022).

[11] Alfonsín, J. L. R. On variations of the subset sum problem. *Discrete Applied Mathematics 81* (1998).

[12] Alharby, M., Aldweesh, A., and Moorsel, A. V. Blockchain-based smart contracts: A systematic mapping study of academic research (2018).

[13] Alizadeh, F., and Goldfarb, D. Second-order cone programming. *Mathematical Programming, Series B 95* (2003).

[14] AnalystPrep. Mean-variance portfolio theory. https://analystprep.com/study-notes/actuarial-exams/soa/ifm-investment-and-financial-markets/mean-variance-portfolio-theory/, 2020. Accessed: 2023-10-07.

[15] Androulaki, E., Barger, A., Bortnikov, V., Muralidharan, S., Cachin, C., Christidis, K., Caro, A. D., Enyeart, D., Murthy, C., Ferris, C., Laventman, G., Manevich, Y., Nguyen, B., Sethi, M., Singh, G., Smith, K., Sorniotti, A., Stathakopoulou, C., Vukolić, M., Cocco, S. W., and Yellick, J. Hyperledger fabric: A distributed operating system for permissioned blockchains. vol. 2018-January.

[16] ApS, M. *MOSEK Optimizer API for Python. Version 10.1.12*, 2022.

[17] Arora, J. S. Discrete variable optimum design concepts and methods. *Introduction to Optimum Design* (1 2004), 513–530.

[18] Author, N. N. Vat - detailed documentation, the maker protocol's core accounting system, 2022.

[19] Born, A., Gschossmann, I., Hodbod, A., Lambert, C., and Pellicani, A. Decentralised finance – a new unregulated non-bank system? https://www.ecb.europa.eu/pub/financial-stability/macroprudential-bulletin/focus/2022/html/ecb.mpbu202207_focus1.en.html. Accessed: 2023-10-08.

[20] Bringmann, K. A near-linear pseudopolynomial time algorithm for subset sum. vol. 0.

[21] Bringmann, K., and Wellnitz, P. On near-linear-time algorithms for dense subset sum.

[22] CFI Team. Gini coefficient a statistical measure of economic inequality in a population. https://corporatefinanceinstitute.com/resources/economics/gini-coefficient/. Accessed: 2023-10-04.

[23] Chaleenutthawut, Y., Davydov, V., Kuzmin, A., and Yanovich, Y. Practical blockchain-based financial assets tokenization.

[24] Davidsson, M. Portfolio theory and cone optimization. *Journal of Applied Finance & Banking 1* (2011).

[25] Davydov, V., and Yanovich, Y. Optimal portfolio sold-out via blockchain tokenization.

[26] Diamond, S., and Boyd, S. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research 17*, 83 (2016), 1–5.

[27] Ellinger, E. W., Mini, T., Gregory, R. W., and Dietz, A. Decentralized autonomous organization (dao): The case of makerdao. *Journal of Information Technology Teaching Cases* (2023).

[28] Fu, J., Zhou, W., and Zhang, S. Fabric blockchain design based on improved sm2 algorithm. *International Journal on Semantic Web and Information Systems 19* (2023).

[29] Galil, Z., and Margalit, O. Almost linear-time algorithm for the dense subset-sum problem. *SIAM Journal on Computing 20* (1991).

[30] Garey, M. R., and Johnson, D. S. Computers and intractability: A guide to the theory of np-completeness (series of books in the mathematical sciences). *Computers and Intractability* (1979).

[31] Genčev, M. A note on a property of the gini coefficient. *Communications in Mathematics 27* (2020).

[32] Gharehchopogh, F. S., Ucan, A., Ibrikci, T., Arasteh, B., and Isik, G. Slime mould algorithm: A comprehensive survey of its variants and applications. *Archives of Computational Methods in Engineering 30* (2023).

[33] Giorgi, G. M., and Gigliarano, C. The gini concentration index: A review of the inference literature. *Journal of Economic Surveys 31* (2017).

[34] Gordon, J. What is the concentration ratio? https://thebusinessprofessor.com/en_US/economic-analysis-monetary-policy/concentration-ratio-definition. Accessed: 2023-10-05.

[35] Guo, H., and Yu, X. A survey on blockchain technology and its security. *Blockchain: Research and Applications 3* (2022).

[36] Hali, N. A., and Yuliati, A. Markowitz model investment portfolio optimization: a review theory. *International Journal of Research in Community Services 1* (2020).

[37] Herve M. Tenkam, J. C. M., and Mwambi, S. M. Optimization and diversification of cryptocurrency portfolios: A composite copula-based approach. *Appl. Sci. 2022, 12, 6408* (2022).

[38] Inc., W. R. Mathematica online, Version 13.3. Champaign, IL, 2023.

[39] Jin, C., and Wu, H. A simple near-linear pseudopolynomial time randomized algorithm for subset sum. vol. 69.

[40] Ju, L., Yu, N., and Ma, M. The optimization of the portfolio selection based on ac-cvar model—evidence from china's privately offering funds. *Open Journal of Business and Management 10* (2022).

[41] Kellerer, H., Pferschy, U., and Pisinger, D. *Knapsack Problems*. 01 2004.

[42] Kenton, Wouter, Soren, Tom, and B, C. Maker protocol 101, 12 2020.

[43] KENTON, W. Concentration ratio definition, how to calculate with formula. https://www.investopedia.com/terms/c/concentrationratio.asp, 2020. Accessed: 2023-10-04.

[44] Koiliaris, K., and Xu, C. Faster pseudopolynomial time algorithms for subset sum. *ACM Transactions on Algorithms 15* (2019).

[45] Kolpakov, R., and Posypkin, M. Optimality and complexity analysis of a branch-and-bound method in solving some instances of the subset sum problem. *Open Computer Science 11* (2021).

[46] Kumar, R., Khan, F., Kadry, S., and Rho, S. A survey on blockchain for industrial internet of things. *Alexandria Engineering Journal 61*, 8 (2022), 6001–6022.

[47] Lenstra, H. W. Integer programming with a fixed number of variables. *Mathematics of Operations Research 8* (1983).

[48] Li, S., Chen, H., Wang, M., Heidari, A. A., and Mirjalili, S. Slime mould algorithm: A new method for stochastic optimization. *Future Generation Computer Systems 111* (2020).

[49] Lipton, A., and Treccani, A. *Blockchain and distributed ledgers: mathematics, technology, and economics*. WSPC, 2021.

[50] Lobo, M. S., Vandenberghe, L., Boyd, S., and Lebret, H. Applications of second-order cone programming. *Linear Algebra and Its Applications 284* (1998).

[51] Magnanti, T., and Orlin, J. Integer programming, 2022.

[52] Mahajan, S. Concentration ratios for businesses by industry in 2004, 2006.

[53] Maishera, H. What are lending protocols? the rise of defi lending, 11 2021.

[54] Markowitz, H. Portfolio selection. *The Journal of Finance 7*, 1 (1952), 77–91.

[55] Mayer, H. Ecdsa security in bitcoin and ethereum : a research survey. *Blog.Coinfabrik* (2016), 1–10.

[56] Meerschaert, M. M. Computational methods for optimization. *Mathematical Modeling* (1 2013), 57–112.

[57] Monash Blockchain Technology Centre. How do we know blockchain can't be hacked or manipulated (or can it?). https://www.monash.edu/blockchain/news/how-do-we-know-blockchain-cant-be-hacked-or-manipulated-or-can-it. Accessed: 2023-10-08.

[58] Morais, E., Koens, T., van Wijk, C., and Koren, A. A survey on zero knowledge range proofs and applications. *SN Applied Sciences 1* (2019).

[59] Nasrudin, A. Concentration ratio: Meaning, formula, how to calculate, pros, cons. https://penpoin.com/concentration-ratio/. Accessed: 2023-10-05.

[60] Nesterov, Y., and Nemirovskii, A. *Interior-Point Polynomial Algorithms in Convex Programming*. 1994.

[61] Ningsih, E. S., Sukono, and Rusyam, E. Optimization of foreign exchange using kelly criteria model. *International Journal of Recent Technology and Engineering (IJRTE) 8* (2019), 7963–7967.

[62] Onyiriuba, L. Chapter 17 - sensitizing bankers in developing economies to securitization risks and management. In *Bank Risk Management in Developing Economies*, L. Onyiriuba, Ed. Academic Press, 2016, pp. 321–337.

[63] Ortiz-Cerezo, L. L., Carsteanu, A. A., and Clempner, J. B. Sharpe-ratio portfolio in controllable markov chains: Analytic and algorithmic approach for second order cone programming. *Mathematics 10* (2022).

[64] Pishro-Nik, H. Introduction to probability, statistics, and random processes. *Kappa Research LLC* (2014).

[65] Pisinger, D. Linear time algorithms for knapsack problems with bounded weights. *Journal of Algorithms 33* (1999).

[66] Polak, A., Rohwedde, L., and Wegrzycki, K. Knapsack and subset sum with small items. vol. 198.

[67] Salleras, X., and Daza, V. Zpie: Zero-knowledge proofs in embedded systems. *Mathematics 9* (2021).

[68] Shuai, Y., and Wei, C. An evaluation system for defi lending protocols.

[69] Sitthiyot, T., and Holasut, K. A simple method for measuring inequality. *Palgrave Communications 6* (2020).

[70] Taherdoost, H. Smart contracts in blockchain technology: A critical review. *Information 14* (2 2023), 117.

[71] Thieu, N. V., and Mirjalili, S. MEALPY: a Framework of The State-of-The-Art Meta-Heuristic Algorithms in Python, June 2022.

[72] Venter, G. *Review of Optimization Techniques*. 2010.

[73] Vielma, J. P. Mixed integer linear programming formulation techniques, 2015.

[74] Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python, 2020.

[75] World Bank. Gini index. https://databank.worldbank.org/metadataglossary/gender-statistics/series/SI.POV.GINI. Accessed: 2023-10-04.

[76] Yanovich, Y. Lecture note in introduction to blockchain, February 2023.