The Future Banks

May 2023

1 Introduction

The future is here! Yes, you read it right. We are living in an age of technology where advancements in finance and engineering are transforming our lives. From finance to engineering, technology is there to assist us in every aspect. However, amidst all the positive changes, there is one area that lacks a clear and transparent system: the banking system. Banks often charge hefty interest rates on transactions, prioritizing their profits over fair treatment of customers. They provide minimal interest rates to depositors while charging exorbitant amounts on loans. Our money is predominantly controlled by banks and governments, who dictate whether we can deposit or withdraw funds. The decisions made by central banks, such as raising interest rates or geopolitical crises, can create a domino effect, leading to bank runs and causing individuals to lose access to their own funds. We have witnessed such situations in the past, as exemplified by The Silicon Valley Bank incident. To address these issues, Mike Ray McDonald and Fernando Martinelli introduced the idea of Decentralized Finance (DeFi) built on blockchain technology. DeFi presents a revolutionary solution to the problems of traditional banking. It offers several benefits, including significantly reduced transaction times. While traditional banks may take 2-3 working days to process transactions, DeFi can accomplish the same task within as little as 15 minutes. This accelerated speed enables users to conduct financial activities more efficiently. Another crucial aspect of DeFi is the ability to convert crypto assets into stablecoins. Stablecoins are digital currencies pegged to the value of a stable asset, typically the US Dollar. By converting cryptocurrencies into stablecoins, users gain stability and protection against the volatility associated with many cryptocurrencies. This stability facilitates easier financial planning and transactions within the DeFi ecosystem. In addition, DeFi introduces the concept of flash loans, which streamline the borrowing process by minimizing paperwork. Unlike traditional banks that require extensive documentation, DeFi relies on smart contracts enforced by the community. These contracts act as a guarantee for the repayment of borrowed funds. If the borrower fails to fulfill their obligation, the collateral provided as security is automatically confiscated. This streamlined approach benefits both borrowers and lenders, making the lending process more efficient and accessible. One of the significant advantages of DeFi is that it is not controlled by any single

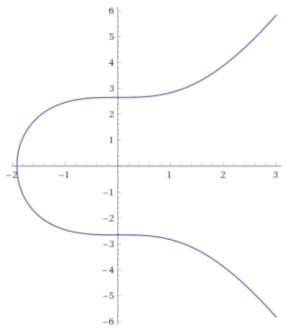

Figure 1: Graph of an elliptic curve

company or entity. Instead, it operates on a decentralized basis, governed by the people who participate in the system. This ensures a fair and transparent financial ecosystem, where decisions are collectively made by the users. As we explore the world of decentralized finance, we will delve deeper into how it functions and examine the relationship between qubits and blockchain technology. Additionally, we will explore the algorithms that ensure the safety and security of this technology, making it resilient against various types of attacks, including brute-force attempts. Join me on this fascinating journey as we uncover the intricacies of decentralized finance and its potential to reshape the future of banking.[1][2][3]

1.1 Introducing elliptic curves

In this section, we will explore elliptic curves in the realm of real numbers (R). We begin by considering a cubic polynomial of the form

$$x^3 + ax + b$$

where a and b are constants, and the polynomial does not have any repeated roots. An elliptic curve is defined as the set of points (x, y) that satisfy the equation

$$y^2 = x^3 + ax + b$$

along with a point called the point at infinity, denoted as O. To find points on an elliptic curve, we follow a straightforward process. We assign real values to the variable x, substitute them into the right side of the equation, and then check if the result is a square in the real number field (R), indicating a real number greater than or equal to zero.

As previously discussed, elliptic curves are crucial in Bitcoin and blockchain technology. Specifically, we will be working with a specific curve: $y^2 = x^3 + 7$.

The graphical representation of this curve is depicted in Figure 1.

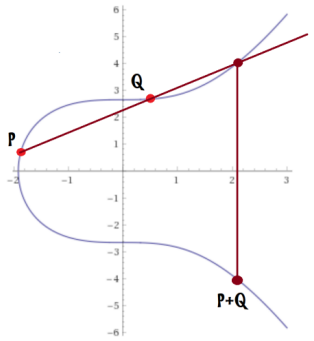

Figure 2: Sum of different points in an elliptic curve

Within an elliptic curve, an addition operation is defined. Point O serves as the identity element for this operation, meaning that for any point P on the curve, P + O = O + P = P.

When considering two distinct points, P and Q, on a curve, their sum can be determined using the following steps:
1. Draw a straight line, denoted as r, passing through points P and Q.
2. Let this line r intersect the curve at another point. The sum P + Q is obtained by reflecting this new point symmetrically with respect to the x-axis.
The accompanying image provides an example that illustrates the process of obtaining the sum of points P and Q on the given curve.

Furthermore, let's delve into the calculation of the sum of a point P with itself, denoted as 2P.
To calculate the double of a point P on an elliptic curve, the following steps are taken:
1. Draw the tangent line, denoted as r, to the curve at point P.
2. The line r intersects the curve at another point. The double of P, represented as 2P, is obtained by reflecting this point symmetrically with respect to the x-axis.

The accompanying image provides an example that illustrates the process of calculating the double point P on the given curve.
Additionally, it is important to note that the operation of doubling a point is a crucial component in elliptic curve arithmetic. It finds various applications in cryptographic protocols and algorithms.

By following this process, we observe that the set of points on an elliptic curve, equipped with the defined sum operation, satisfies essential properties. These properties include the associative property, the existence of the neutral element (O), and the existence of a symmetric element for every point on the curve. As a result, the set forms an abelian group.
Obtaining analytical expressions for the calculation of the sum and the double

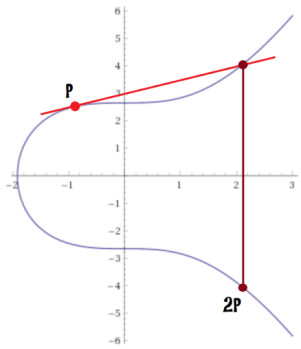

Figure 3: Twice a point in an elliptic curve

of a point on an elliptic curve is not a challenging task. Consider two points on an elliptic curve: P = (x_1, y_1) and Q = (x_2, y_2). The sum of P and Q is calculated as follows: P + Q = (x_3, y_3) where:

$$x_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)^2 - x_1 - x_2$$

$$y_3 = -y_1 + \left(\frac{y_2 - y_1}{x_2 - x_1}\right)(x_1 - x_2)$$

For the double of a point P on an elliptic curve, denoted as $2P$, the resulting point can be expressed as (x_3, y_3), where:

$$x_3 = \left(\frac{3x_1^2 + a}{2y_1}\right) - 2x_1$$

$$y_3 = -y_1 + (\frac{3x_1^2 + a}{2y_1})(x_1 - x_3)$$

We will now extend the concept of an elliptic curve to the domain of finite fields, specifically denoted as Fq. For the sake of simplicity, we will concentrate on elliptic curves defined over characteristic fields other than 2 and 3. As we have discussed earlier, an elliptic curve comprises points (x, y) that satisfy the equation:

$$y^2 = x^3 + ax + b$$

To ensure that the cubic polynomial,

$$x^3 + ax + b$$

does not possess multiple roots. Additionally, the curve includes a distinguished point called the point at infinity, represented as O.

When working with elliptic curves over finite fields, we can define operations such as addition and scalar multiplication using the expressions mentioned earlier. Consequently, the set of points on an elliptic curve, along with the defined addition operation, exhibits an abelian group structure. The addition operation involves geometric constructions and algebraic calculations to compute the sum of two points on the curve. The resulting point is obtained by reflecting the intersection point of a line passing through the given points with respect to the x-axis. Scalar multiplication entails performing repetitive additions of a point with itself. By employing specific formulas, such as the double-and-add algorithm, we can efficiently compute the scalar multiples of a point on the elliptic curve. Overall, the combination of finite fields, the defined equations, and the operations of addition and scalar multiplication establish a well-defined mathematical structure, forming an abelian group on the set of points of an elliptic curve over Fq. The study of elliptic curves over finite fields is an active area of research, with ongoing efforts to deepen our understanding and explore new applications for these mathematical structures.

1.2 Cryptographic Elements in Bitcoin and Blockchain

The cryptographic algorithm utilized in Bitcoin and blockchain relies on the security provided by the discrete logarithm problem in elliptic curves over finite fields. This problem is analogous to the discrete logarithm problem encountered in finite fields, thereby establishing a connection between the two.

Specifically, when considering an elliptic curve denoted as c defined over a finite field F_p, the discrete logarithm problem involves determining the existence of an integer $n \in \mathbb{Z}^+$ such that $nP = Q$ for given points P and Q belonging to c. Resolving this problem requires finding the value of n that satisfies the equation, given the points P and Q.

The utilization of the discrete logarithm on elliptic curves presents us with a novel mathematical problem that exhibits a contrasting simplicity in one direction and a profound complexity in the other. Consequently, given an integer $n \in \mathbb{Z}^+$ and a point $P \in c$, the computation of $Q = nP$ can be efficiently performed. However, the inverse scenario, wherein we possess points P and Q and seek to derive the discrete logarithm (n), becomes an arduous task, particularly when n is large.

The intricacy of the summation operation on elliptic curves implies that solving the discrete logarithm problem for such curves is significantly more challenging compared to finite fields. As a result, utilizing algorithms based on elliptic curves allows us to use shorter cryptographic keys while still achieving a commendable level of security. This attribute plays a crucial role in various applications, including Bitcoin and blockchain, where safeguarding sensitive data is of utmost importance.

A critical question arises: how can we efficiently compute nP when n is large? The straightforward approach of repeatedly adding P to itself n times is highly inefficient. Fortunately, the most suitable method for calculating nP is the technique known as the method of successive squares.

Initially, this method is employed to efficiently compute powers, utilizing the concept of successive squaring. However, it can also be adapted for the calculation of nP. Let's delve into the steps involved:

First, we generate a sequence of points: $P, 2P, 4P, \ldots, 2^r P$, where the value of r satisfies the condition $2^r \leq n \leq 2^{r+1}$. It's worth noting that each element in this sequence is double the value of the previous one.

Next, we express the value of n in binary form, obtaining its binary representation.

We traverse the binary expression of n from left to right. For each bit encountered:

- If the bit is 0, we double the current point. For instance, if the bit is 0 at position i, we replace the current point with $2^i P$.
- If the bit is 1, we double the current point and then add P to it. Using the same example, if the bit is 1 at position i, we replace the current point with $(2^i P) + P$.

The final result can be expressed as:

$$nP = \sum_{i=0}^{r} (a_i 2^i) P$$

Substituting the values obtained previously and making these sums, we obtain:

$$nP = a_0 P + a_1(2P) + \ldots + a_r 2^r P$$

For instance, if we want to calculate $13P$, as $13 = 2^3 + 2^2 + 2^0$, we have to calculate $P, 2P, 4P, 8P$, so:

$$13P = 8P + 4P + P$$

As we have said before, for Bitcoin and the blockchain, the elliptic curve is given by:

$$y^2 = x^3 + 7$$

over the field F_p where:
p =115792089210356248762697446949407573530086143415290314195533631308867097853951.

2 Decentralized

The figure above illustrates how a centralized banking system operates, showcasing its functioning and processes.
In the past, we relied on centralized finance systems, where a central authority, such as the government and banks, controlled the flow of money. They possessed the ability to print more currency at their discretion and had the

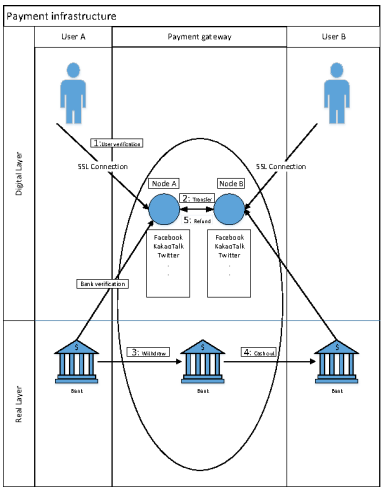

Figure 4: Payment Process in Centralized System

power to prevent individuals from borrowing money or even deny them access to a bank account. In such a system, proving ownership of one's money was challenging since it was essentially based on trust. If you entrusted your money to them, it was difficult to argue against their authority over it, as they could manipulate it or exercise control without much recourse on your part. When it comes to running a business, the limitations and deposit caps imposed by traditional finance can be quite restrictive. Moreover, the exorbitant rates charged in traditional finance can be burdensome, making it essential to seek alternative solutions. This is where Decentralized Finance (DeFi) steps in, offering a game-changing approach. DeFi operates through transparent lines of code that function as autonomous banks, open to everyone. The beauty lies in the fact that these codes can be audited, allowing individuals to verify their legitimacy and avoid any potential scams. By embracing decentralization, DeFi systems become resistant to censorship, providing businesses with greater freedom and flexibility. Not to mention, the cost-effectiveness of DeFi compared to traditional finance is truly remarkable, enabling businesses to save significantly on fees and expenses.

Now, let's shift our focus to the advantages of Decentralized Finance (DeFi) after discussing the drawbacks of traditional finance.

Decentralized finance (DeFi) relies on three core elements: cryptography, blockchain technology, and smart contracts. The predominant cryptocurrency used in DeFi is Ethereum. This ecosystem leverages decentralized applications (DApps) that offer a diverse range of services, including lending, insurance, and more. These DApps often utilize stablecoins, which are pegged to a central currency such as the US Dollar. It is worth noting that stablecoins are typically overcollateralized, which means

1 USD=0.6 Eth Coins

And if you want your money back just return the Coin.[4]

2.1 Smart Contracts

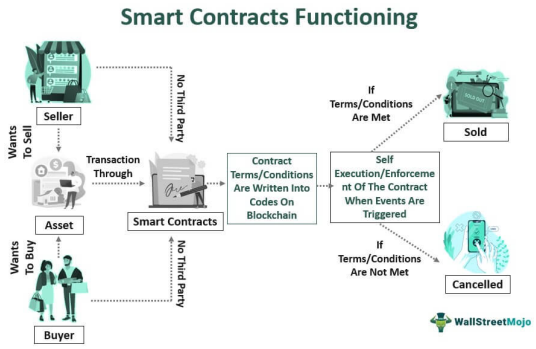

Figure 5: Working of a Smart Contract

When borrowing a loan from a bank, failing to repay it can have legal consequences, as the government or the bank can take action to ensure repayment. However, in the case of cryptocurrencies, there is a concern that someone could easily borrow funds and evade repayment since there is no centralized authority overseeing transactions. To address this issue, a solution is needed to provide custody of our digital assets, such as coins, while still allowing lending and borrowing within the crypto ecosystem. This is where smart contracts come into play. Smart contracts are essentially pieces of code that facilitate and enforce the terms of agreements between parties involved in a transaction. By leveraging the power of blockchain technology, smart contracts provide a trustless and decentralized mechanism to secure funds and ensure their proper utilization. These contracts can be programmed to automatically execute actions and enforce repayment terms, removing the need for a centralized authority. With smart contracts, lending and borrowing in the crypto space can be conducted with greater security and transparency, as the code itself acts as a safeguard against potential fraud or default.
Smart contracts function similarly to traditional if-else loops in programming languages. They are executed when specific conditions outlined in the contract are met. These conditions determine when the contract will be executed and identify the buyer and seller involved. Additionally, smart contracts can be stored in blocks and executed at predetermined times. The versatility of this algorithm allows for a wide range of applications. For instance, we can store codes or addresses of stablecoins, among other possibilities.
Assuming you have 10 Ethereum, each valued at $100, and you have a strong belief in the Ethereum project, you decide not to sell them. Instead, you opt to use them as collateral to borrow $800 worth of Tether, a stablecoin pegged to the United States dollar. With this borrowed amount, you engage in trad-

ing activities, experiencing both gains and losses. As time goes by, it eventually becomes necessary to repay the loan. At this point, you possess $850 worth of Tether. However, your 10 Ethereum tokens have appreciated in value to $150 each. In order to retrieve your Ethereum collateral, which is now worth a total of $1,500, you must repay the original $800 loan. Considering your trading activities, let's assume you made a profit of $50. Thus, you currently have control over assets worth $1,500, in addition to the $50 trading profit. However, in the event that your trades resulted in losses, and you only have $750 in Tether, you have two options:
Option A: Add the extra $50 to repay the full loan of $800 and reclaim all your collateral.
Option B: Keep your $750 but forfeit your 10 Ethereum, which could potentially be worth a significant amount now. Additionally, in the world of cryptocurrency, there is a type of loan known as a flash loan. These loans are short-term and typically last for a brief period, such as 10 seconds. The concept behind a flash loan is that if you can take advantage of a price discrepancy between different platforms, such as buying Ethereum for $10 on Coinbase and selling it for $11 on Gemini, you could potentially make a profit of $1 for each successful trade executed within that short timeframe.

3 Algorithms

The below diagram shows how a transaction takes place in Bitcoin.

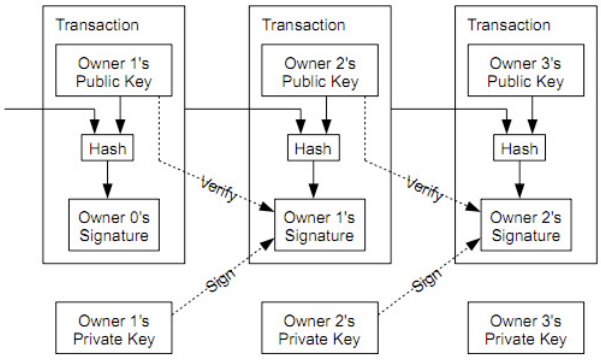

Figure 6: Transaction Process

The SHA-256 algorithm is commonly used for securing data. It serves as a basic hashing algorithm, which ensures the integrity of information. In traditional banking transactions, there are two parties involved: the buyer and the seller. However, in blockchain technology, there is an additional party: the community. This community consists of individuals like you and me who verify the legitimacy of transactions by reviewing the underlying code. When

a transaction occurs in the blockchain, it undergoes a validation process by the community. Only after the transaction is approved by the community, the funds are transferred to the buyer's wallet. The concept of wallets and their functioning will be discussed in the upcoming section. Moreover, we have developed an original algorithm specifically designed for storing data in the blockchain. This algorithm incorporates principles from Graph Theory and probabilities to enhance its functionality and security.

3.1 SHA-256 Algorithm

The Secure Hash Algorithm 256-bit (SHA-256) is a widely adopted cryptographic hash function that belongs to the SHA-2 (Secure Hash Algorithm 2) family. Developed by the National Security Agency (NSA) in the United States, SHA-256 is known for its robust security, efficient performance, and versatility in various cryptographic applications.

3.1.1 Hash Function

A hash function is a mathematical algorithm that transforms an input message of arbitrary size into a fixed-size output called a hash value or digest. SHA-256, as a cryptographic hash function, is designed to be computationally infeasible to reverse-engineer or reproduce the original input from the output hash value.

3.1.2 Structure of SHA-256

SHA-256 operates on a message block of 512 bits and produces a 256-bit hash value. The algorithm consists of several logical steps, including message padding, message block partitioning, message schedule computation, and round-based hash value calculation using a compression function.

3.1.3 Features and Functionality of SHA-256

- One-Way Function: SHA-256 is a one-way function, meaning that it is computationally infeasible to determine the original input (message) from its hash value. This property provides pre-image resistance, ensuring that it is extremely difficult to reverse-engineer the input from the output hash.
- Deterministic Output: Given the same input, SHA-256 will always produce the same output hash value. Even a small change in the input will result in a significantly different hash, exhibiting the avalanche effect. This property enables consistency and predictability in hash calculations.
- Collision Resistance: SHA-256 is designed to be collision-resistant, meaning that it is computationally improbable to find two different inputs

that produce the same hash value. This property ensures the integrity and security of the hash function, making it highly reliable for applications that require data integrity verification.

- Efficiency: SHA-256 offers efficient computation, allowing it to process large amounts of data relatively quickly. It utilizes bitwise logical operations, modular addition, and logical functions to perform calculations on message blocks. This efficiency makes SHA-256 suitable for various practical applications that involve hashing extensive data.

- Fixed Output Size: SHA-256 generates a fixed-size output of 256 bits (32 bytes), regardless of the input size. This fixed output size makes it convenient for storing and comparing hash values efficiently.

- Cryptographic Strength: SHA-256 is considered to be cryptographically secure, meaning that it has withstood extensive cryptanalysis and is resistant to various known attacks. It offers a high level of protection against brute-force attacks, pre-image attacks, and collision attacks, making it suitable for a wide range of cryptographic applications.

- Standardized Algorithm: SHA-256 is a widely adopted and standardized hash function. It is defined in the Federal Information Processing Standards (FIPS) publications by the National Institute of Standards and Technology (NIST), ensuring its compatibility and interoperability across different systems and implementations.

- Verifiability and Integrity: SHA-256 provides a means to verify the integrity of data. By computing the hash value of a file or message and comparing it with the original hash, one can determine if the data has been tampered with or modified. This property is crucial for ensuring the trustworthiness of transmitted or stored information

3.1.4 Limitations of SHA-256

- Quantum Computing Vulnerability: SHA-256 algorithm is really weak in front of Quantum Algorithms such as Shor's Algorithm as Quantum Algorithm have the ability to break the mathematical foundation of Sha-256

- Collision Probability: Even though SHA-256 is collision-resistant but there are still chances that two different inputs can have the same hash values.Also, as in when new algorithms get developed there much more chances of the collisions to occur in SHA-256.

- Speed and Resources: We know that SHA-256 is really efficient in hashing number of datas but as the number of data increases then there are much more chance of limitation in speed of SHA-256.

- Deterministic Output: This determinism can be a limitation in scenarios where randomization or non-repeatability is desired, such as in certain privacy-enhancing protocols or some forms of randomized encryption schemes.
- Limited Input Size: This limitation means that if the input message exceeds the block size, it needs to be divided into multiple blocks for hashing, which may introduce additional computational overhead and complexity.
- Dependence on Hash Value integrity: The algorithm mostly relies on the secrecy of the hash value.If someone gets through the secrecy it can really expose the overall system security.

3.1.5 Conclusion

SHA-256 is a widely used cryptographic hash function renowned for its security, efficiency, and extensive applications. It provides robust protection against data tampering, ensures data integrity, and enables secure communication in various domains. Despite its strengths, researchers continue to explore advanced hash functions and cryptographic algorithms to address future challenges, including quantum computing threats.

3.2 The SHAKS Algorithm

This algorithm employs Graph Theory, Probability, and Hashing to store data in blocks. The central concept revolves around a core block known as the head node. A cluster is defined as a collection of blocks that utilize hash values for data storage. Each cluster comprises layers, with a typical layer containing 64 blocks. Additionally, multiple identical layers make up a cluster. This innovative structure ensures efficient data storage and retrieval within the blocks.

Figure 7: Diagram of a Cluster

Now all the nodes in the cluster are interconnected to each other using connections and are also connected to the head node. The system is a type of Undirected Graph. Now let's suppose we want to store some passwords in the block or we want to conduct a financial transaction by storing some data and then validating it with data from another cluster and thus completing the transaction. We are also going to use Complex numbers. We have a Complex numbers like

$$C = a + ib$$

We are utilizing complex numbers in a cryptographic system. The variable 'a' represents the public key, while the variable 'b' represents the private key. These values will be stored in a complex number object, denoted as C. The value of 'a' will be stored in the head node and made accessible to all the data. Additionally, we have a specific data item, denoted as "A," which we intend to process through our hash function.

$$y^2 - 4x^2 + x^4 = 0$$

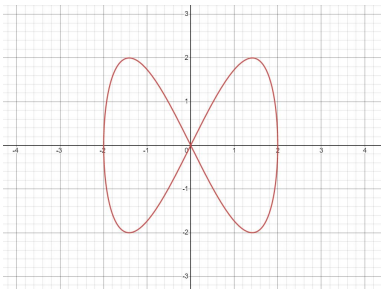

Figure 8: Graph of the above function

The hash function generates a unique value for each input. For example, if the input is 'A', it is converted into its binary equivalent (0b100001). Then, we compute the XOR of all the digits in the binary representation, breaking the input into 2 bits at a time. For instance, the output may be '101'. This output is passed into a function that produces a value with a precision of up to two decimal places, such as '10198.99'. Next, the private key, which is only accessible to the blocks within that specific cluster, is multiplied by the hash value. The private key could be a number, such as '2546'. On the other hand, the public key is used to establish connections with other clusters. Now, we need to store this first hash function value in a block. To do so, we navigate to a specific block and define a variable 'k'.

$$K = 1$$

The value of 'k' for the head node is now known. For the next layer of nodes, we have a probability of 0.5, and this probability will be passed to the hash

function. The resulting value will then be added to the existing value of the data.
Suppose we have reached a block to store the data in. If the block is full, it will disconnect itself from other vertices and only remain connected to the Head Node.
To facilitate financial transactions, the data is stored in blocks. Let's consider a scenario where one wallet address is located in cluster 1 and another wallet address is in cluster 2. Suppose we need to access the wallet address in cluster 1. First, we start by entering the head node with the wallet address. Then, we apply the hash function and multiply the result with the private key. This process helps us navigate to the specific block where the wallet address is stored. Since the addresses are stored in blocks that are already filled, we only need to check those filled blocks. This approach reduces the time complexity and speeds up the process. Once we reach the block, we validate whether both hashes (the one obtained from the hash function and the one associated with the wallet address in the cluster) are the same. If the wallet addresses match in both clusters, the transaction is approved. On the other hand, if the addresses don't match, the transaction is denied. By employing these steps, we can efficiently access and verify wallet addresses in different clusters to facilitate secure financial transactions.

4 Conclusion

In our exploration, we delved into the fascinating realm of elliptic curves, unraveling the mathematical foundations that underpin the security and functionality of cryptocurrencies. By understanding the intricacies of these curves, we gained insights into the robustness of cryptographic protocols and the protection of sensitive data in blockchain networks. Furthermore, our foray into algorithm development through the lens of graph theory and probability yielded promising results. By leveraging the power of graphs and probability theory, we aimed to devise an algorithm that could enhance the efficiency and effectiveness of cryptocurrency transactions. This innovative approach opens up avenues for further research and optimization in the field, paving the way for more streamlined and secure digital transactions. Finally, we explored the immense potential of Smart Contracts in the realm of Decentralized Finance (DeFi). With their self-executing nature and immutability on the blockchain, Smart Contracts offer unprecedented opportunities for automating financial processes, ensuring trustless interactions, and enabling innovative financial applications. By harnessing the power of Smart Contracts, the decentralized finance domain has the potential to revolutionize traditional financial systems and empower individuals with greater control over their assets and transactions. In conclusion, our comprehensive exploration of these topics has provided a solid foundation for understanding the underlying principles and practical applications of cryptocurrencies, algorithm development, and Smart Contracts. By staying abreast of these cutting-edge technologies, we can con-

tribute to the ongoing advancements in the world of finance, paving the way for a more inclusive, secure, and efficient financial ecosystem.

References

[1] Wolfgang Karl Härdle, Campbell R Harvey, and Raphael CG Reule. Understanding cryptocurrencies, 2020.

[2] Campbell R Harvey. Cryptofinance. *Available at SSRN 2438299*, 2016.

[3] Ghassan O Karame and Elli Androulaki. *Bitcoin and blockchain security*. Artech House, 2016.

[4] Dirk A Zetzsche, Douglas W Arner, and Ross P Buckley. Decentralized finance (defi). *Journal of Financial Regulation*, 6:172–203, 2020.