# Optimizing Cryptocurrencies for Profit


By
**Shehreyar Rashid**
**NUST201463915MSEECS60014F**

Supervisor
**Dr. Syed Taha Ali**
**Department of Computing**

A thesis submitted in partial fulfillment of the requirements for the degree of Masters of Science in Information Technology (MS IT)

In

School of Electrical Engineering and Computer Science,
National University of Sciences and Technology (NUST),
Islamabad, Pakistan.

(June 2018)

# THESIS ACCEPTANCE CERTIFICATE

Certified that final copy of MS thesis written by Mr. SHEHREYAR RASHID, (Registration No. NUST201463915MSEECS60014F ), of SEECS-NUST has been vetted by undersigned, found complete in all respects as per NUST Statutes/Regulations, is free of plagiarism, errors and mistakes and is accepted as partial fulfillment for award of MS/M Phil degree. It is further certified that necessary amendments as pointed out by GEC members of the scholar have also been incorporated in the said thesis.

Signature: _____

Name of Supervisor: _____

Date: _____

Signature (HOD): _____

Date: _____

Signature (Dean/Principal): _____

Date: _____

# Approval

It is certified that the contents and form of the thesis entitled “Optimizing Cryptocurrencies for Profit" submitted by Shehreyar Rashid have been found satisfactory for the requirement of the degree.

Advisor: Dr. Syed Taha Ali

Signature:_____

Date:_____

Committee Member 1: Dr. Muhammad Shahzad Younis

Signature:_____

Date:_____

Committee Member 2: Dr. Fahd Ahmed Khan

Signature:_____

Date:_____

Committee Member 3: Ms. Haleemah Zia

Signature:_____

Date:_____

# Dedication

This thesis is dedicated to my parents, my wife Mishal Jahan and my son Hanzalah Shehreyar, who never lost faith in me and supported me unconditionally throughout this effort and made it possible. I could not have done it without them.

# Certificate of Originality

I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any degree or diploma at NUST SEECS or at any other educational institute, except where due acknowledgement has been made in the thesis. Any contribution made to the research by others, with whom I have worked at NUST SEECS or elsewhere, is explicitly acknowledged in the thesis.

I also declare that the intellectual content of this thesis is the product of my own work, except for the assistance from others in the project's design and conception or in style, presentation and linguistics which has been acknowledged.

Author Name: SHEHREYAR RASHID

Signature: _____

# Acknowledgments

First of all, I would like to thank ALLAH Almighty for providing me with the insight and knowledge to fulfill this task and pursue my dreams; nothing would‘ve been possible without HIS blessings and benevolence. I would like to express my deep and sincere gratitude to my supervisor, Dr. Syed Taha Ali, Department of Electrical Engineering, SEECS, and NUST. His wide knowledge and his logical way of thinking have been of great value for me. His understanding of the topic, persistent support and constant encouragement has inspired and motivated me as a researcher and a student.

I am also extremely thankful to my committee members Dr. Shahzad Younis, Ms. Haleemah Zia and Dr. Fahd Ahmed for taking out time from work to give their valuable suggestions and kind feedback. I would also like to thank Asad Salman and Mishal Jahan for their constant support and help. In the end I would like to thank everybody who was important to the successful realization of my thesis.

# Abstract

There is a considerable body of research exploring the dynamics of the mining process for Bitcoin, the mining ecosystem, and the inherent risks of concentrating network computing power in the hands of just a few miners. Our research is not about hash riddle portion of block mining in which miners compete with one another to find a block which is hard by design. Instead our work focuses on finding candidate blocks which will be mined and will contribute to the miner's profit. Our work, we believe, is the first to optimize the mining process by giving miners the facility to fine-tune the trade-off between block size and miner profit.

The problem we are trying to solve is similar to the knapsack problem. In knapsack problem we have a given set of values and weights for particular items, among which we have to select items to maximize the value but the weight should not exceed its knapsack limit.

Here items are transactions having transaction fees as its value and transaction size as its weight and considering knapsack limit as 1MB block, which will contain these transactions.

# Table of Contents

# List of Figures

# CHAPTER 1:

# INTRODUCTION AND MOTIVATION

# Introduction and Motivation

## 1.1 Introduction

Now a days the technological advancement is so powerful that it transforms the very basic pillars of our society. It fundamentally influences the way our economy and governance system functions. Crypto currencies have caused a paradigm shift, as they are the hottest assets of the current world. Being a digital currency, crypto currency is used as a medium of exchange. In crypto currency, encryption algorithms are used for currency generation and verification of transfer of funds. Crypto currency has become the global phenomenon known to the whole world. Different kinds of cryptographic techniques are used, to enable the transactions to take place. Some of the basic crypto currencies involve: Bitcoin, Litecoin, Dash, Ripple and Ethereum, among which Bitcoin is the famous of all. Satoshi Nakamoto the inventor of bitcoin accidentally invented the bitcoin. He called it a peer-to-peer Electronic Cash System. Bitcoin is the first decentralized crypto currency that ensure secure peer-to-peer transactions carried out independently of banks.

## 1.2 Problem statement

It is expected that as Bitcoin becomes more mainstream, memory constraints will force miners to select transactions to mine as per their size and transaction fees. At this moment, there is no reliable mechanism available to the Bitcoin community to fine tune the tradeoff between size and fees, leading to the mining of non-optimal blocks.

## 1.3 Motivation

Bitcoin has emerged over the last few years as a very popular digital payment system. However, recent research reveals, there is considerable room for optimization in Bitcoin in terms of scalability, efficiency, communication, etc. In this thesis, we plan to optimize the mining process.

## 1.4 Background

### 1.4.1 Cryptocurrency

Crypto currency is formally defined as a medium of exchange created and stored electronically in the blockchain, using encryption techniques to control the creation of monetary units and to verify the transfer of funds.

**Properties of Cryptocurrency:**

1. **No intrinsic value:**
    - It has no intrinsic value
2. **No physical form**
    - It has no physical form i.e. it has no existence out of the network.
3. **Irreversible**
    - It is irreversible i.e. once committed; the transaction cannot be reversed by anyone.
4. **Pseudonymous**
    - It is pseudonymous i.e. there is no connection of transactions to the real world identities of users. Addresses numbers on which bitcoins are received are random chains of addresses. It does not require any ID.
5. **Global**
    - It is global and fast i.e. transactions are independent of physical location. They are sent instantly to anyone, anywhere in the globe.
6. **Secure**
    - It is secure because of strong cryptography i.e. it is encrypted string of data or a hash. Encryption algorithms are used using public and private keys
7. **Permissionless**
    - It is permissionless i.e. for carrying out transactions; there is no higher authority to be asked for. It can be installed by anybody as it is free to use.

**Working of Cryptocurrency**

Cryptocurrency works in a form of a cycle. In informal terms, it involves following basic steps

1. Someone requests a transaction
2. The requested transaction is broadcast to peer-to-peer network consisting of nodes (computers).
3. This network validates the transaction along with the user's status using algorithms.
4. After verification, this transaction is added to the block of new transactions to be recorded in ledger
5. This new block is added to the existing blockchain permanently

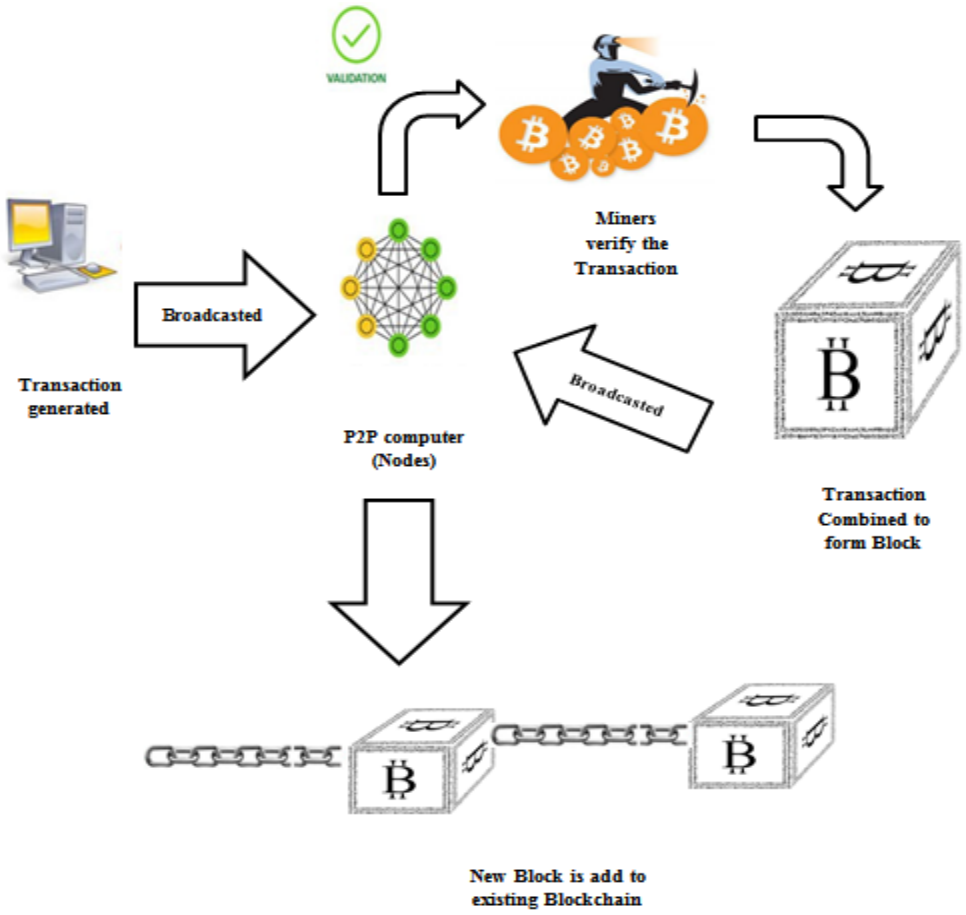


**Figure 1 Cryptocurrency Transaction lifecycle**

## 1.4.2 What is a Bitcoin?

Bitcoin is a groundbreaking internet technology. It is a decentralized currency, since there is no real third party entity or central bank. There is a peer-to-peer network consisting of individual hosts called nodes that carry out the transactions. All these individual hosts or nodes agree on a particular protocol and how it is implemented and used. All tasks are collectively managed by the peers in a network. It makes the currency smarter by atomizing cash or money. It is more than virtual money or a transaction system. It's up to the user to decide the unit of bitcoin. It can be dollars or euros or any other. The user can assign properties to each unit. A bitcoin can represent many kinds of properties. It is something more than money. It can be programmed into any kind, like a company can program its budgets in terms of bitcoin for salaries, machinery, materials and maintenance. It allows us to rebuild and innovate things in our own way there-by increasing the efficiency.

**How does a Bitcoin work?**

When we talk about transferring a bitcoin among different identities or entities, we do not mean those identities in real. It means that these identities have nothing to do with the real-world identities. They are considered as a sequence of numbers or we can say that they are pseudonyms. This is done to provide some privacy to the users with whom we want to transact over a bitcoin network.
In the world of bitcoin we talk in terms of block-chains, transactions, mining etc. Let us discover what these terminologies actually mean.

## 1.4.3 Blockchain

A blockchain is a system of distributed data and logic. It works with cryptographic keys, distributed networks, record keeping and more. People are trading worldwide. Everyone trades with everyone else. This trade is recorded in a book or ledger. We can refer it to book keeping. The information kept in this book is isolated and confidential and is not accessible by the public. Bitcoin enables a network of computers to maintain a collective bookkeeping through internet.

This bookkeeping is accessible to the public and is available in one digital ledger. This digital ledger is fully distributed across the network. This is called a Blockchain. In order to approve and facilitate the transactions, a middle party is entrusted such as governments banks etc. But in this case, there is no need of a third party. A blockchain is a distributed ledger that is completely open to anyone. It contains all information of transactions such as time, date, amount and participant of the transaction. Every computer or node over a network has a copy of this blockchain.

A blockchain uses a peer-to-peer network, allowing anyone to join. Every user gets the full copy of a blockchain. When a user creates a block, it is sent to everyone on the network. Each node verifies this block based on some agreed consensus and add it to their own blockchain.

Each transaction is verified by the Bitcoin miners who maintain the ledger. For every node in the network, it is ensured that every transaction's current state is recorded in the ledger and it is updated. Every transaction that occurs is public, and each node agrees that transaction has occurred over the network at a particular date and time. A blockchain can be trusted as it is a shared single source of truth. A block-chain is a transaction database shared by all the nodes in the network. A complete copy of the currency's block chain contains all the transactions ever executed in the currency. With this information it is known that how much value each address belonged to at any time in the history.

The blockchain is connected through the pointer to the previous block. In a blockchain, each block contains a data, a hash of the block, a hash value of the previous block, a timestamp and the root of transactions.

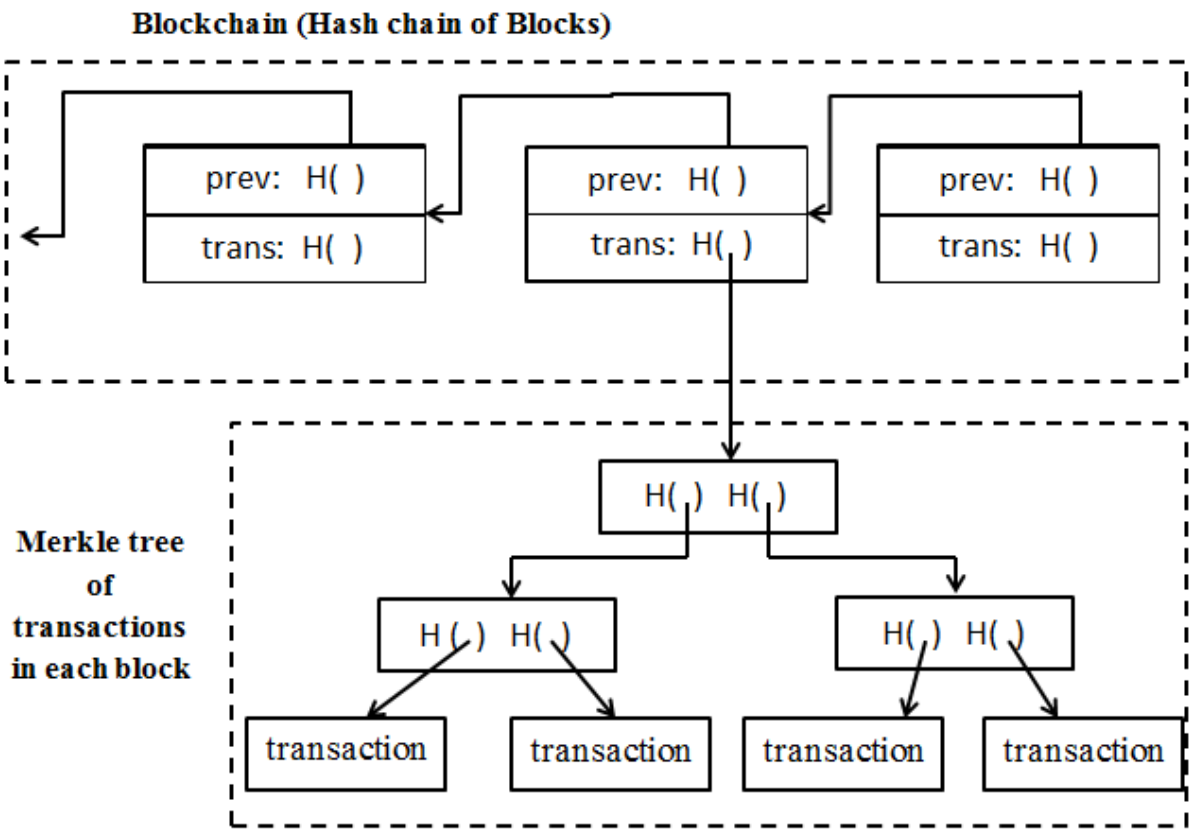


**Figure 2 Two Hash Structure: 1) Hash of previous Block. 2) Merkle tree of Transactions within block.**

The data of the transaction includes the sender, receiver and the amount of coins. Hash is used for the identification of the block and its contents and is always unique. If the content of a block changes, the hash value of that block also changes, and that block does not remains the same block. The hash of the previous block creates the chain of blocks that makes a blockchain secure. The timestamp includes the date and time of the bitcoin transactions. Since the volume of transactions increases, it increases the size of the block significantly, which further increases the process of validation. So a block creates a hash tree (based on Merkle trees) whose root contains the hash value of all transactions of the block. There is a continuous chain of blocks based on these hash values from the initial or genesis block to the current block. This order of the blocks has to be in chronological order otherwise the previous block's hash would not be known. Once a block is added to the chain, it cannot be modified. If so, other blocks would also have to be regenerated.

A chain is valid only if it starts with the initial or genesis block and all the transactions within it are valid. For each block in the chain, there is only one path to the genesis block. However, coming from the genesis block, there may be forks. Forks of a block are created when two blocks are created just a few seconds apart. Whichever block is received first, the generating nodes build onto that block, becoming the part of main chain, as that chain is longer. There is no use of the blocks in shorter chains. They are referred as invalid chains. When a user moves to another longer chain, the short-chain transactions are added back to the transaction pool in the queue and included in another block.

**Pros:**

A blockchain makes everything visible to the user, keeping it secure at the same time. It is cost efficient, as it reduces the probability of mistakes, expense and delays carried out in traditional record assimilation methods. Moreover being automated system, it is fast and flawless and easy to maintain. It is also time efficient as blockchain processes the payments regardless of time-zone issues over the world. A blockchain is used to manage and secure digital relationships in terms of records.

**Cons:**

Besides its complexity, cost, security issues and its ever-growing size, the complicated concept of the blockchain technology is not understood by most of the users.

In the current era, the Blockchain technology advances and matures rapidly and its demand is on the rise.

### 1.4.4 Transactions:

In Bitcoin protocol, we usually talk in terms of coins. These virtual coins do not actually exist; rather they are the transactions that have some ownership rights. (Tschorsch and Scheuermann 2016) A transaction is a digitally signed statement or declaration among two parties, based on the transfer of the bitcoins. (Narayanan, et al. 2016)

In the real world life, when we talk about a wallet, it's a physical one that can be touched. Similarly the coins that we use are also physical, that can be spending in any ways. Bitcoin does not work actually like a wallet. We never hold a bitcoin physically; instead we have rights for it.

Just like we transfer the coins from person to person, the rights are transferred from one holder to another in case of a bitcoin. These rights are basically known as transactions.

#### 1.4.4.1 Structure of Transaction:

A transaction comprises of a version no., inputs outputs and lock time. If we want to send bitcoins to someone, we need to know, where these bitcoins come from? Unlike the real coins, that we pay no attention to that who gave us which coin, for each particular bitcoin we need to know who gave us that bitcoin. In a transaction we can put together many locations and identify where the bitcoin came from. We keep a track of who gave us which bitcoin before passing it on. For this we have a list of inputs. In each input we list the transaction that generated the bitcoin. We combine the inputs of transactions that locate the places of the bitcoins and send it to the output transaction. Just like we can have multiple inputs to our transactions, we can assign them by combining them together to the multiple outputs. A transaction in actual contains an array of inputs and outputs (Bonneau et al., 2015 )R3. Furthermore, the rules, to whom the bitcoin is to be transferred (output) and from whom (input), are also specified in a transaction. These rules are specified on the basis of some public, private key conditions or encryption strategies. Each transaction has some fee. It is basically an incentive from nodes for validation of transactions. For input transactions, the sum of all inputs must be greater than or equals to the output transaction. Incase if input transactions are greater than the output ones, their difference is the transaction fee. This transaction fee is collected by the miner who mines the block. For inputs, we need to specify the proof that we meet the conditions. Since the outputs from one transaction becomes the input to the other transactions, so the conditions on the output must be validated to the conditions in the input i.e. input's conditions are already met. All these inputs, outputs and conditions, become a digital message together. Each message or a transaction has its name in terms of a hash. We take a hash of a message that becomes the transaction's name, by which a transaction is specified. We can specify a transaction in three parts i.e. Metadata, input and output as shown in Figure. Metadata refers to the information about transaction, i.e. size of transaction, series of inputs and outputs.

In each block, there is one special transaction called coinbase transaction, which is the first transaction of a block. This transaction have no inputs, otherwise it is the same. It has outputs and conditions similar to the other transactions.


**Figure 3 Structure of transaction (Narayanan, et al. 2016)**

### 1.4.4.2 Example of Transaction:

In order to understand the mechanics of a transaction, let us consider an example. Suppose we have two parties, **A** and **B**. **A** wants to transfer some bitcoins to **B**. Now each party have its own identity, a pseudonym or verification key generated by each party. The verification key for all parties is public. Corresponding to each public verification key, there is a private key called the signing key for each separate identity. Let's consider that A has already received some bitcoins from other parties. Now instead of including transaction details of each previous transaction, which **A** has previously received, it will apply a cryptographic hash function to these previous transactions and get a digest for each party. **A** has now the ownership of the transactions in the form of digests. **A** will then include these digests in transaction records for **B**. These digests are

necessary to verify the chain of transactions that guarantees that **A** has these transactions from these parties. Now all the transactions are inputs of **A**. When **A** wants to transfer the bitcoins to **B**, **A** must include the list of recipients i.e. the verification public key of **B** and specify the amount of number of bitcoins to be transferred. All this data regarding to a transaction is digitally signed by the sender i.e. **A**, through its private signing key. This signature is appended to the contents of the transaction that binds the identity of **A** with the transaction record, in order to ensure that **A** has created the block having its private signature that corresponds to the public key of **A**. Then this whole data is broadcasted to all nodes over the Bitcoin peer-to-peer network showing that **A** wants to send the specific amount of bitcoins to **B**. For this, a transaction also has its fee, which is collected by the miner who mines the block first.

The node in the network receives this transaction information along with the other multiple transactions taking place over the network. The nodes then incorporate this transaction in the ledger containing all other transactions that take place over the network. The Bitcoin nodes or miners starts collecting all unincorporated transactions into a transaction block. Each transaction has an entry in a ledger. A transaction block has pages in ledger. These pages contain transactions.[1] The miners combine the transactions into pairs and apply hash function to it and get a digest. It then combines those digests into a single digest value that encodes all of the unincorporated transactions received by individual nodes. A bitcoin network contains a series of transaction blocks. This digest is further combined with the hash of transaction block, accepted by the bitcoin network, previously. In this chain of transaction blocks, every block incorporates the transactions of the previous block and this chain continues till the first coinbase block. No block is isolated now; each contains the data of the preceding block except the coinbase block. After incorporating all these block, a cryptographic hashing function is applied to this combination in order to generate a sequence of numbers. This sequence of numbers is further used along with the proof of work by the miners. Once the proof of work is found by any miner, it is then announced and broadcasted to all nodes in the peer-to-peer network. All other nodes stop working on the current proof and start working on the new updated and unincorporated transaction block.

[1] https://www.khanacademy.org/economics-finance-domain/core-finance/money-and-banking/bitcoin/v/bitcoin-transaction-block-chains

### 1.4.5 Proof of work:

The proof of work actually a computational effort done by the miners in order to find a solution. It's sort of solving a puzzle that requires serious computational effort to solve a specific problem.

### 1.4.6 Mining:

#### 1.4.6.1 Background/introduction:

A bitcoin miner is actually a node in a bitcoin network, which makes an effort to validate a transaction. What miner actually does is, it compiles all the transactions, taking place over a network, in a transaction block and solves the current block with a valid solution that requires a heavy computation. Once a block is validated by a miner, it is then easy for rest of the nodes in a network to confirm the solution. A miner includes a transaction called reward in the block for solving each block. This reward is meant for a miner only, for doing all the computational effort to mine the block. The miner also includes an encoded sequence of numbers to this block of transaction, called a Proof of work. It requires a lot of effort to do in order to get a transaction fee and the reward set for validating the transaction. Since we do not have a single block of transactions, all previous blocks of transactions are incorporated with the current block, so we have a transaction block chain. Once all unrecorded transactions and the proof of work are combined, the miner broadcasts the details on the peer-to-peer bitcoin network to all nodes. When this new transaction block is verified, it is then used by all nodes and they append new blocks to the chain. The bitcoin miners work on the longest chain or the chain on which the most work is done.

Transactions are assigned a priority. Transactions that are larger and have older coins and having small scripts have high priority.

**priority=sum(input_value * input_age)/ size_in_bytes** [2]

Here input_age is how long ago input was on the block chain.

Because of high block reward to miners, currently a transaction fee does not matter. But the size of mining reward is continuously decreasing; it's getting half after every 4 years. Every four

[2] https://en.bitcoin.it/wiki/Transaction_fees

years the block reward is cut in half restricting the aggregate supply of bitcoins to 21 million. It started with 50 BTC, and then it decreased to 25 BTC, now-a-days it is 12.5 BTC. and it will continue to decrease. So mining rewards will become quite less in the upcoming years and the transaction fees will be dominated, from where miners will get the revenue.

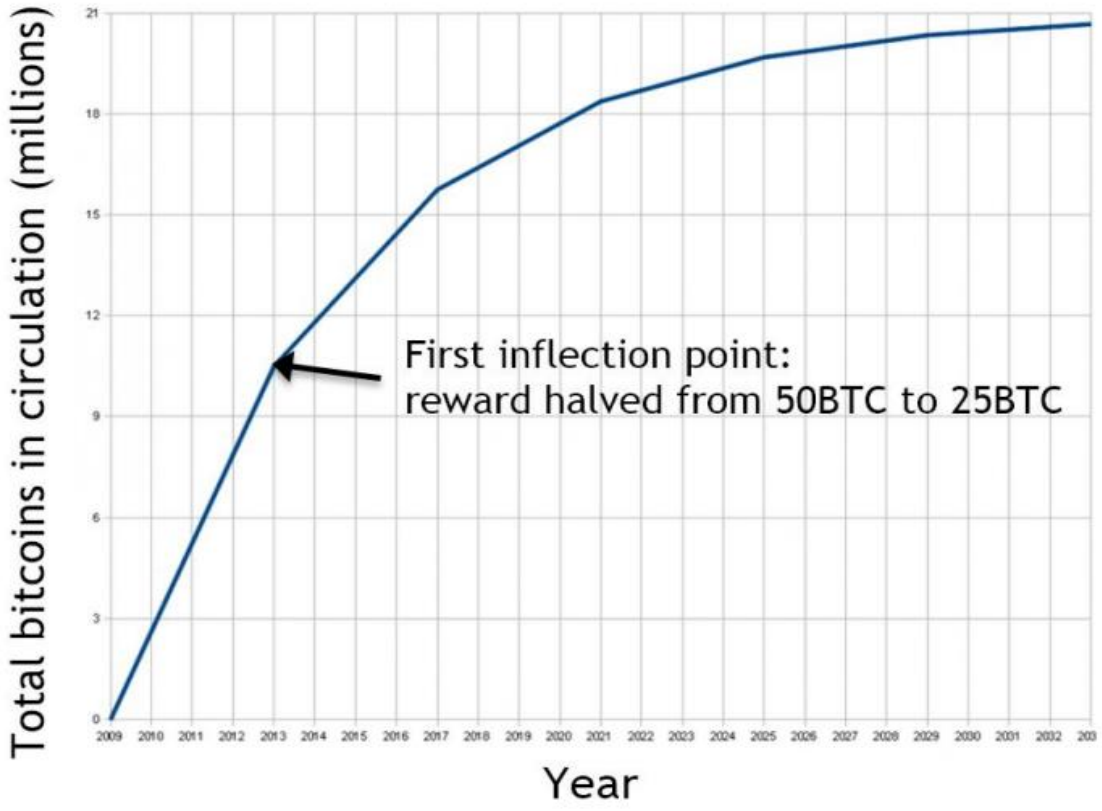


**Figure 4 The block reward is halved every four years restricting the aggregate supply of bitcoins to 21 million (Narayanan, et al. 2016).**

**Some basic mining steps:**

A bitcoin miner validates the new transactions and store and broadcast the blockchain. In order to mine a block, a miner has to do the following steps:

1. Join the bitcoin network and become a node.
2. Pay heed to all transactions nodes are making over the network and validate them

3. Notice new blocks and transactions in those blocks, validate them and maintain a blockchain
4. Assemble the validated block based on the transactions
5. Find a nonce that will make a block consider valid by doing some computational effort.
6. Assure that other miners accept the new block and start mining it.
7. A profit will be rewarded to the miner in the end.

Among all these steps, finding a valid block is the basic step which miner performs, that is useful to a bitcoin network.

**Some default mining strategies:**

There are several strategies that we need to keep in mind for mining, such as:

1. Getting good hardware, cheap electricity and run as fast as possible.
2. The transaction selected to be included in the block must have fee above than the minimum transaction fee.
3. Deciding which block to mine on the top of i.e. selecting the longest current valid chain.
4. Announcing the new block immediately after finding it.

Miners are free to implement any strategy, its not necessary for them to apply the default strategies. They can follow their own strategy besides the default one, which might result in more profit.

### 1.4.6.2 Mining data structures:

For a block there are two hash based data structures, one is the blockchain in which the header points to the previous block's header and other is the hashed based binary merle tree. A miner solves the problem with a known input of the current blockchain and creates a hash target. The SHA-256 Hash function (Secure Hash Algorithm) is used for hashing the transactions. (Narayanan, et al. 2016)Using computer's resources multiple combinations of inputs are guessed by the miner to find the hash target that solves the problem. A miner who solves the problem first, gets the reward and incentive. A merkle tree is used for data integrity in a blockchain. Since a block contains multiple transactions containing data, each transaction is passed through a hash function separately in the first step to get the unique hashes. Then transactions are combined in pairs and passed again through the hash function. All transactions

are hashed till a single root hash is found, which forms a complete merkle tree. A merkle tree is used for finding any changes in the tree by rerunning the transactions and comparing the results with the root hash.

Each block's transactions are combined with the previous time stamping and is hashed, which is further combined with the hash of the next block to generate another hash. In this way it forms a chain of blocks with each additional timestamp reinforcing the previous ones. The proof of work is performed by the miners, in which we scan a value, that when passed through the hash function, produces an output (hash) containing a combination of zeros. The miners look for a specific number of zeros that are considered valid. They hash the previous hash, the number of transactions in the block and the nonce in the block. If they couldn't find the correct hash, they increment the nonce and take the hash again. They keep on doing it until a correct hash is obtained, which requires a lot of computational effort. (Narayanan, et al. 2016)

#### 1.4.6.3 Mining difficulty target:

The hash of any valid block has to be lower than the difficulty target value, which now-a-days requires 64+ leading zeros. Based on miners' efficiency in last two weeks, the difficulty target is calculated every two weeks. It is computed as

**next_difficulty= previous_difficulty*(2 weeks)/(time to mine 2016 blocks)**

Here time to mine 2016 blocks is the expected number of blocks in 2 weeks at 10 minutes per block. It is a fixed constant chosen for the bitcoin from the beginning. Every 2 weeks the difficulty is reset. The mining difficulty increases over time. This increase is not constant or exponential; it depends on the number of miners taking part in mining the blocks in the market. As the number of miners who hash the blocks increase, the blocks are found faster and the difficulty increases. Every 2 weeks the time to find a block is reset to 10 minutes, since the difficulty remains the same so the miners mine the block faster. After 2 weeks the process starts all over again.

The mining success is measured in terms of Goodput. It is the product of the throughput i.e. time to mine the mine the blocks and the success rate i.e. how often the computation have errors.

Goodput= throughput * success rate

A number of transactions are mined on a daily basis. We can check the latest blocks mined on blockchain.info that shows the live transactions that are mined. The most recent block is

shown on the top the list. Each block contains some statistics regarding to it such as number of transactions, the height of the blockchain, its timestamp, and hash of the block, hash of the previous block, the merkle root and so many other things.

#### 1.4.6.4 Mining Energy:

Since mining is energy intensive, high computational hardware is required for it. Mining requires megawatts of power and high operating costs of electricity (Narayanan, et al. 2016) and (Tschorsch and Scheuermann 2016). Starting from CPUs, then GPUs, FPGAs and now ASCIs are used. Evolution of mining diagram

Mining requires energy. Energy in terms of mining can be classified as Embodied Energy that is physical energy used to engineer mining equipment such as chips etc.; Electricity, that is used to perform heavy computations and Cooling, which is required to keep the equipment protected to ensure that it does not malfunction.

#### 1.4.6.5 Mining pool:

Mining is a random process of uncertainty; no one knows when the miner will find a new block. Being a small miner, there is a high probability that the expected number of blocks the miner will find for his first year is 0, which is so low. For a small miner, there is a big chance that he makes no money at all. So a number of miners join together and form a pool in which every miner mines the block having the same coinbase recipient known as pool manager. A pool manager manages the rewards, no matter who finds the block and distributes them among the candidates of the pool depending on the amount of work done by the candidate. The amount of work done by a miner is known by outputting the shares that are near-valid blocks. The more the valid blocks or shares are found by the miner, the more revenue a candidate is going to be rewarded with. A pool manager broadcasts the block to work on to all candidates. They start mining the block and the first miner who finds the valid block publishes that block in the pool. Mining pools lower the uncertainty of the mining, allowing small miners to get reward even if they are new. But it denies the concept of bitcoin decentralization and limits the miners from running the full nodes.

### 1.4.7 How my work related to Knapsack Problem:

Bitcoin transactions are more complex than we think. When someone asks for bitcoin transaction, their bitcoin wallet and bitcoin network perform certain steps to insure bitcoins have been transfer legally. A transaction is a digitally signed statement or declaration among two entities, based on the transfer of the bitcoins. The transaction is comprised of three important things: 1) Transaction Inputs 2) Transaction Outputs 3) An amount. Transaction has been explained in detail in section 1.4.4 earlier.

Transaction has some size, which generally depends on numbers of **inputs** and **outputs**. If these numbers are larger transaction size will be larger. These transactions are broadcasted over the bitcoin network. Miners receive these transactions and after verification these transaction are included in Memory pool (Mempool) of Miner where transaction waits till they are confirmed. By confirmation we mean transactions are included in **Block** and are removed from Mempool. On an average Mempools have over 100,000 transactions. Whereas Block has finite size of **1MB a**nd only some of the transactions can be included. Those who have requested for transaction, they give some incentives to miners so that they include their transaction in the block. This incentive is called **Transaction Fee** (Bonneau, Bitcoin mining is NP-hard 2014**).** So we can say that:

1. Mempool has **set of transactions**.
2. Each Transaction has some kind of **Size**.
3. **Transaction Fee** is paid to Miner if transaction is to be included in the Block.
4. **Block** having finite capacity.

From the above mentioned points we can easily conclude that the problem under consideration is similar to knapsack. The knapsack problem is one of the examples of combinatorial optimization problems. Its name was originated from a situation where the number of objects having different values and weights, finding the most profitable set of objects that can be placed in a fixed-sized backpack.

The formal definition of Knapsack Problem is:

Knapsack of fixed-sized volume W greater than 0 and set of objects (set I) {1, 2, 3… n} having value (profit) <v1, v2 …> and weight (size) <w1, w2 …> as parameters. We wish to find the subset T (T⊆I), such that

Maximizes $\sum_{i\varepsilon T} vi$

Subject to $\sum_{i\varepsilon T} wi \leq W$

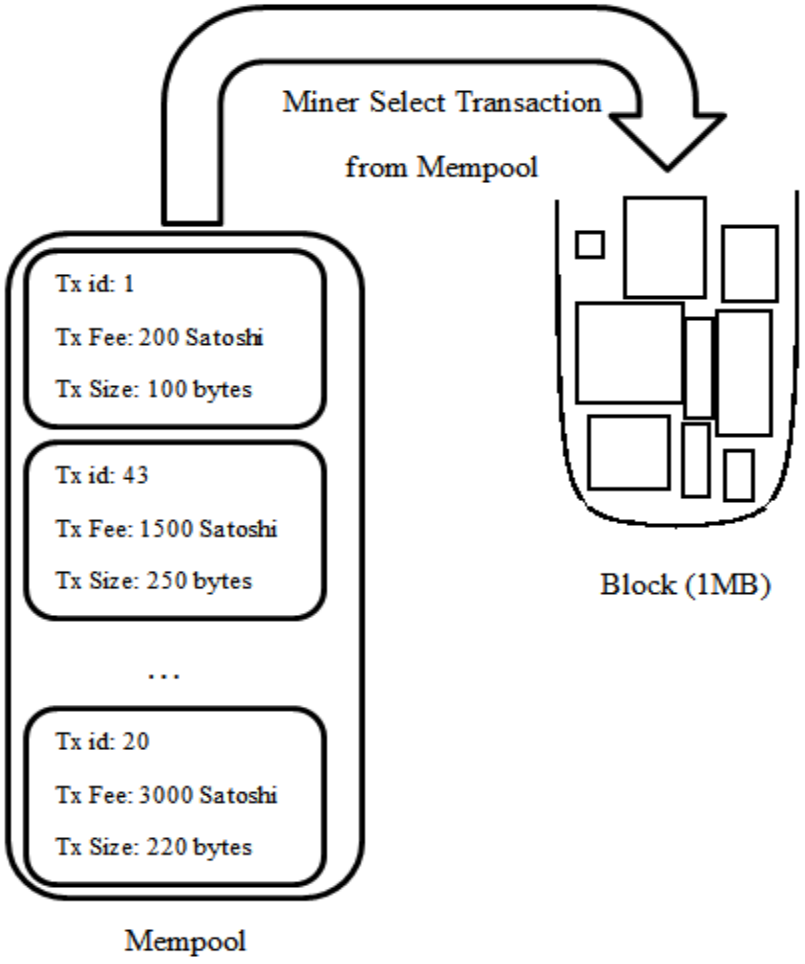


**Figure 5 Knapsack problem: Transaction Fee as value, Transaction Size as Weight and Block Capacity as Knapsack**

In this thesis we aim to optimize Miner's Reward ($\sum_{i\varepsilon T} TxFee$).

# CHAPTER 2:

# LITERATURE REVIEW

# Literature review

According to (Tschorsch and Scheuermann 2016) and (Narayanan, et al. 2016), Satoshi Nakamoto, the inventor of Bitcoin, never intended to introduce a new technology; instead it was supposed to be a purely peer-to-peer version of electronic cash. When transactions occur, it is recorded with all nodes over the network in a block. it is then verified using mathematical operations by all nodes. The node that solves the operation for verification first, broadcasts the solution and transaction over the network. All other nodes verify the transaction and solution and approve the block. That first node is rewarded with some amount of currency. to Dr. Saifedean Ammous (Ammous 2016) discussed the applications of the blockchain such as, Digital payments, in which decentralized currency system is used instead of the centralized one, it removes the third party intervention and involves more processing power and time. Smart Contracts, that uses blockchain currency through multi signature wallet and time-programmed payments. Database and Record management, which works only for native blockchain's currency. Digital payment is the most successful one. Furthermore blockchain technology experiences some obstacles also such as Redundancy, Scaling, Regulatory Compliance, Irreversibility and Security.

Florian Tschorsch and Bjorn Scheuermann in (Tschorsch and Scheuermann 2016) discuss the bitcoin's central concepts, its security threats, privacy and other key observations in their research paper. This paper discuses bitcoin in terms of decentralized digital currency, where every node is bank itself with a block-chain acting as a distributed ledger in contrast with the traditional the centralized digital currency, exemplifying the banking model, in which banks play the role of issuing unique serial numbers to the users and maintains the ledger. The concept of double spending is briefly explained in which, same coin is transferred to two different nodes or users, that results in inconsistency of the blockchain. It is avoided by verifying the authenticity of transactions by all users. In order to prevent network attacks, Proof-of-work schemes are being used. These schemes involve computing cryptographic puzzles through SHA-256 hash function. A specific term difficulty (discussed) is used to find how difficult it to find a hash of a target is. Bitcoin also provides rewards to the users in terms of transaction fees and mining. Initially Bitcoin reward was 50BTC, which is reduced to half every four years and now-a-days it is 12.5BTC. Whereas the transaction fees is most likely to increase in future. The bitcoin

transactions include transaction id, inputs and outputs. Each output is used as an input forming a chain of blocks which helps in tracing the transactions back. This output refers to a script, which consists of inputs, outputs and a number of arguments. To make bitcoin secure, a virtual wallet is assigned to each user which contains private and a public key. This wallet can be a software wallet, having a locally running bitcoin instance, which is much prone to attacks; a hardware wallet, containing an offline operating device, making it harder to gain access to attack; a paper wallet, possessing keys in a form of document, similar to cash money; and brain wallet, using passphrase through hashing for private and public key. In order to increase security of wallet, multi-signature transactions are used, that are signed independently. Elliptic curve cryptography is used to secure transactions (Matonis 2014) and (Maxwell 2013). Furthermore change in transaction id without invalidating it refers to transaction malleability, which is somehow related to double-spending with the difference of attacker, who in this case is the receiver party. Solo mining results in significant but infrequent rewards, so miners combine together to form a pool in which they work jointly get continuous small rewards. The bitcoin network is unstructured peer-to-peer overlay network, whose aim is rapid transaction processing and information distribution rather than finding specific data items or files. In terms of scalability, the size of a block is limited to 1MB, to limit the transactions along with the growth rate of block chain, to prevent the block chain extension. More resources will be required for increased scalability. For higher transaction rate, high internet speed with a super peer-based overlay structure is required. The author explores all these points which demonstrate that bitcoin is evolving and developing constantly.

According to Joseph Bonneau, the building block of bitcoin is “proof of work” (Bonneau, Miller, et al. 2015). It is proposed in (Dwork 1992) that bitcoin is basically meant for combating spam emails, but it is never used for this purpose. Bitcoin is used as a currency in terms of transaction for the first time in May 2011. After that bitcoin’s demand has increased many folds and its price keeps on rising. Bonneau discusses the history of bitcoin in the start, covering the entire Bitcoin system including transaction format, scripts, and Nakamoto consensus; block confirmation, mining details including rewards and fees, and pools, network topology, communication protocol. Under stability of bitcoin, transaction validity rule and consensus protocol has been discussed. Incase if the key is lost by the client, he has to suffer irrevocable losses. For the client-side security, Simplified Payment Verification (SPV), Key Management is

done. SPV involves the processing of blockchain's proof-of-work solution. In Key Management, keys are stored and secured in many ways such as key storage on device, offline storage split control, password protected, hosted and password derived wallets. Furthermore, the author discusses the alternative consensus protocols, Bitcoin's modification, its privacy and its functionality's extension.

Miles Carlsten in his paper discuses that there are separate incentives mechanisms for Bitcoin miners: block reward and Transaction fee. According to the writer, in the following years the transaction fees will dominate over the block reward as its keeps on increasing and the block reward is decreasing. (Carlsten, et al. 2016). Joseph Bonneau discuses in his article that selection of transition in bitcoin block is similar to knapsack (Bonneau, Bitcoin mining is NP-hard 2014)

Johnson and Eyal analyzes the mining pool's attack on the other competing pools in different ways (Johnson, et al. 2014) and (Eyal 2015)

**Optimization Algorithms**

Knapsack is a century old NP hard problem. There are many optimization algorithms for solving Knapsack problem. But the algorithm that produces approximate optimized profit in near real time is considered up to the mark. In our scenario, time complexity is essential.

The best approach for solving the knapsack problem in terms of profit and number of items is dynamic optimization algorithm (Kellerer, Pferschy and Pisinger 2003), since it produces exact solution. Some Examples of dynamic optimization algorithm are Exact solution for (RKP), Mixed ILP (MILP) and Branch-and-bound algorithm. The main idea of Dynamic Programming (Time Complexity O (nW)) is to avoid repeated work by remembering partial results (Rolfe 2007).

Monaci discusses the exact solution algorithms for about 5000 instances. According to them, the time complexity of the algorithm discussed is O (Γ nc log n) and space complexity is O(n + Γc log n) (in this scenario we need to copy previous entry in order to produce a new entry) (Monaci, Pferschy and Serafinix 2013).

Caprara talks about Mixed Integer Liner programming (MILP) in their paper. MILP uses n + 1 continuous variables and constraints and n binary variables, so the computational complexity of

the problem does not increase even by enforcing robustness (Caprara, et al. 2000). MILP cannot be used in our case as transaction is ACID in nature.

Bertsimas (Bertsimas and Sim 2003) introduces another class of ILPs which is later improved by (Lee, et al. 2012). Its time complexity is O(nT) whereas the nominal knapsack problem solving time is O(T). According to observations, the Branch-and-bound algorithm is worst from all above (Monaci, Pferschy and Serafinix 2013)

Genetic Algorithms (GAs) is basically a probabilistic optimization algorithm inspired by evolutionary process and genetic inheritance. A Genetic Algorithm maintains a population of candidate solutions for the problem at hand, and makes it evolve by iteratively applying a set of stochastic operators (Selection, Recombination, and Mutation). The two of the major primary issues of GAs are local optimum and time complexity (Tharanipriya and Vishnuraja 2013) and (Zou, et al. 2011). Different researches are being carried out in order to get rid of these problems. Teo and Srinivas propose adaptive mutation techniques which try to solve local optimum. For the better results, self-adaptive mutation GA changes the area of search space by changing mutation rate. Furthermore, we can minimize the time complexity by solving the local optimum (Teo 2006) and (Srinivas and Patnaik 1994).

Moreover, according to (Tharanipriya and Vishnuraja 2013) and (Zou, et al. 2011) a Hybrid Genetic Algorithm can improve the algorithm in terms of speed, performance and precision. Hybrid Genetic Algorithm is basically a combination of Genetic algorithm and Greedy algorithm. It also solves the zero-one Knapsack problem, in which each item is either selected or not.

Since the algorithms discussed earlier are not time efficient. So our focus is on approximation algorithms. Polynomial Time Approximation Scheme (PTAS) and Fully Polynomial Time Approximation Scheme (FPTAS) are some examples of approximation algorithms which has a control over accuracy. PTAS take error parameter ε; smaller the error parameter ε, the slower the approximation algorithm runs. The more running time we provide to the algorithm, the better results we get (Gopalan, et al. 2011). Since the dynamic algorithm takes more running time than the Greedy algorithm so the results produced by the dynamic algorithm are better. In dynamic algorithm, we have exponential time with exact solution, while in greedy algorithm we have

nlogn time with the solution, somehow lower than the dynamic algorithm (Gopalan, et al. 2011) and (Caprara, et al. 2000).

# CHAPTER 3:

# DESIGN AND METHODOLOGY

# Design and Methodology:

Our design activity can be categorized in to following parts: Full Bitcoin Node Synchronization, Data collection, Preprocessing (Mempool Image Creation), and Algorithm for profitable Block creation. Figure 3.1 shows the whole activity in the form of Block diagram. Data used in this thesis is collected by editing Bitcoin Core code taken from github[3].

Three files are maintained for data extraction i.e. files containing:

- Transaction entering in Mempool.
- Transaction exiting from Mempool.
- Block hash, timestamp and transactions fee.

These files are input for Mempool image creation. In this process we replicate Mempool of the Miner which creates the Block in the first place. After that our proposed optimization algorithm selects a subset of transaction from the Mempool such that profit is higher than Miner's transactions fee which is achieved while creating the Block in the first place.

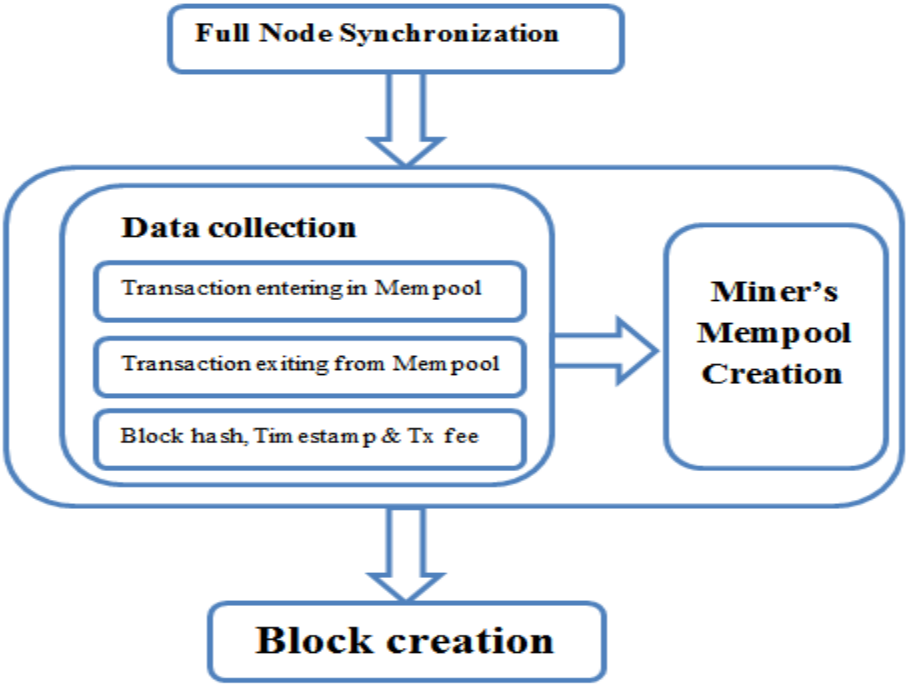


**Figure 6 Block diagram for Complete Algorithm**

[3] https://github.com/bitcoin/bitcoin

## 3.1 Full Bitcoin Node Synchronization

In trusted environment, we need to rely on third party (Bank etc.) to keep everybody's record. Centralized system has complete control and has the right to act according to its own interests at the expense of users. It acts according to its own defined priorities. The main idea behind decentralized peer-to-peer network is to enforce trustlessness. By term trustlessness we mean distributing trust among participating nodes (computers). When a transaction is requested, it is circulated to these nodes over the network. In a Bitcoin Network there are two types of participating nodes: 1) Full Node 2) Miners. Every node receives, validates and broadcast the validated transactions and stores it in their Mempool[4]. A complete ledger is stored by the Full Node, whose size exceeds 160 GB. There is a special node called Miner. It not only validates the block but also mines it.

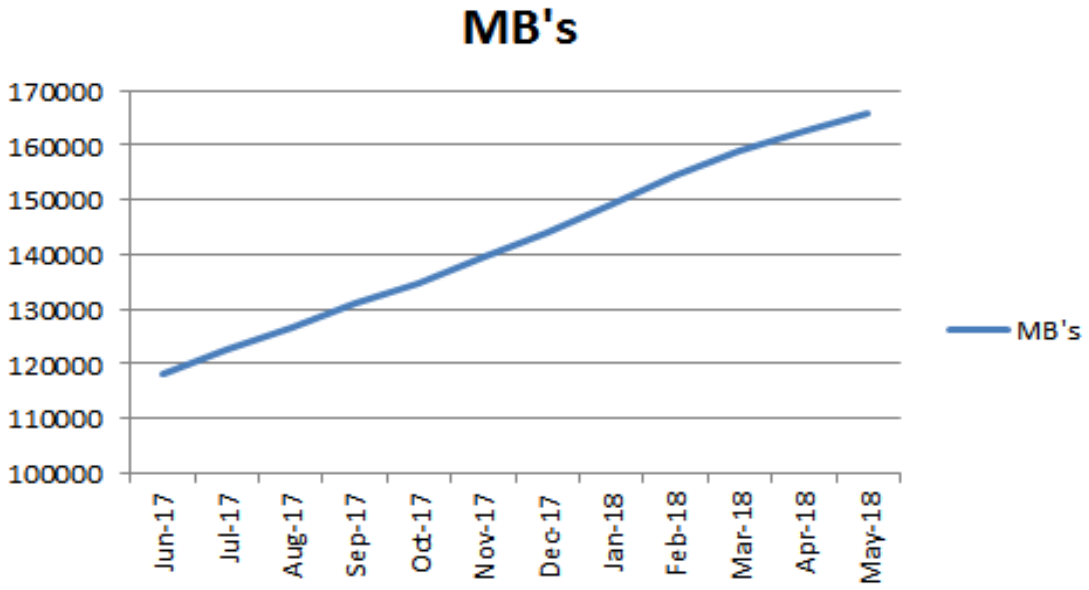


**Figure 7 Blockchain Size (source blockchain.info). 160 GB of Blockchain should be stored by fully validating node.**

Synchronization of the data files takes a quite long time (almost about 2 to 3 weeks). After synchronization, Full Node is ready to verify all the transactions and blocks.

[4] http://wiki.p2pfoundation.net/Blockchain

A Bitcoin Full Node is run for the data collection, which downloads the previous complete history (ledger). Every node has its own copy of ledger. All the transactions received by Full node and Miners are the same with a slight time difference because of peer-to-peer network. Miners create blocks from their own list of transactions which is called Mempool. We replicate the Miner's Mempool.

## 3.2 Data collection

Occasionally some statistics are needed for arduous and stress testing. But the synthetic data is terrible - it does not have a realistic distribution, and it's not easy to understand if the results are reasonable and accurate. However the real or natural data is preeminent. It has realistic distribution, and is easy to understand.

In order to cross check, the optimized block's fee is compared with the actual Bitcoin block's fee created at that instance of time at which the actual block was created. For this real time data collection is needed instead of dummy data.

When two parties exchange the data, this process is called transaction. Transaction has three sections: (1) data, money (bitcoins) and contracts etc. (2) The one who transfers bitcoins. (3) The one who receives bitcoins. Transaction and their fields are discussed earlier in detail.

When someone request for transaction, that transaction is spread across network like a gossip. There is a group of people (participating computers i.e. nodes), and when one of them hears anything, they convey it to the few (8 or more Nodes) more people (connected with), eventually everyone knows about the original information (transaction).

One of the nodes (participating computers) known as miner receives transaction and applies series of validations checks. Some of basic validations checks that a miner applies are listed below[5]:

- Is transaction assembled correctly?
- Is input's and output's lists are empty or not?
- Size of transaction should not be greater than MAX_BLOCK_SIZE (1 MB)
- Reject the transaction if pool already contains transaction.

[5] https://en.bitcoin.it/wiki/Protocol_rules

- Throwaway the transaction if each input, referenced output occurred in other transaction in Mempool.
- Identify orphan transaction and adding to orphan transaction.
- Throwaway the transaction if each input, referenced output never existed or already has been spent.
- Do not accept if the sum of output values is greater than sum of input values.
- Transaction cannot to include in to empty block due to too low transaction fee. Reject that transaction.

If the Transaction survives from above validation checks, it is ready to be added to Mempool and broadcasted to connected nodes[6].

### 3.2.1 Signal Handler

Signal is one of the methods of inter-process communication (IPC), which notifies process or to thread within the same process asynchronously. When a signal is triggered, receiving process halts normal flow of execution and registered Handler is executed after wards normal flow is resumed where it was interrupted. In bitcoin Core (*validationinterface.cpp* and *validationinterface.h*) code there are lots of Signals available. Signal handler is registered for three Signals

1. TransactionAddedToMempool
2. TransactionRemovedFromMempool
3. BlockConnected

#### 3.2.1.1 TransactionAddedToMempool

When the validation is been done and Transaction is ready to be added to Mempool. TransactionAddedToMempool signal is triggered. TransactionAddedToMempool signal writes Transaction and Time at which Transaction received to file mempool_entry.json. Transaction received time to Miner is unknown since Bitcoin has decentralized structure so transactions

[6] https://blog.kaiko.com/an-in-depth-guide-into-how-the-mempool-works-c758b781c608

received time to Miner or to us will be slightly different e.g. Transaction of hash 6ebedfb2baa1e6f7e16b281b064c8eb1b5b0ad196c37db134118d883054e8ce5 received time to Miner (BTC.com) is 1503632513 (UNIX timestamp) whereas this transaction is received by us at 1503632514 having difference of 1 second.
Mempool_entry.json will contain all the incoming transactions (unconfirmed transaction) with respective timestamps.

#### 3.2.1.2 TransactionRemovedFromMempool

Mempool contain all unconfirmed transactions, where each transaction waits for confirmations from Miners. Transaction does not last in the Mempool for long time; it is eventually removed from Mempool. Reasons due to which a transaction is removed are:

- The transaction conflicts with a confirmed transaction (transaction already included in a block).
- The transaction's unconfirmed ancestor conflicts with a confirmed transaction.
- The owner of transaction replaces its transaction by Replace-by-Fee Signaling[7].
- When the Mempool ran out of its size limit. It sorts the Mempool by fee ratio size. Remove transactions from bottom of the list.
- Mempool do not contain transaction indefinitely after sometime (by default 14 days) Mempool throws it out.
- Transactions already included in a block or when the Miner includes certain transaction into Block then transaction is removed from Mempool.

Due to these reasons transaction is eventually removed from Mempool. Removal of transaction generates TransactionRemovedFromMempool Signal. Handler of this signal captures transaction and write that transaction and time of removal into mempool_exit.json log file.

[7] https://github.com/bitcoin/bips/blob/master/bip-0125.mediawiki

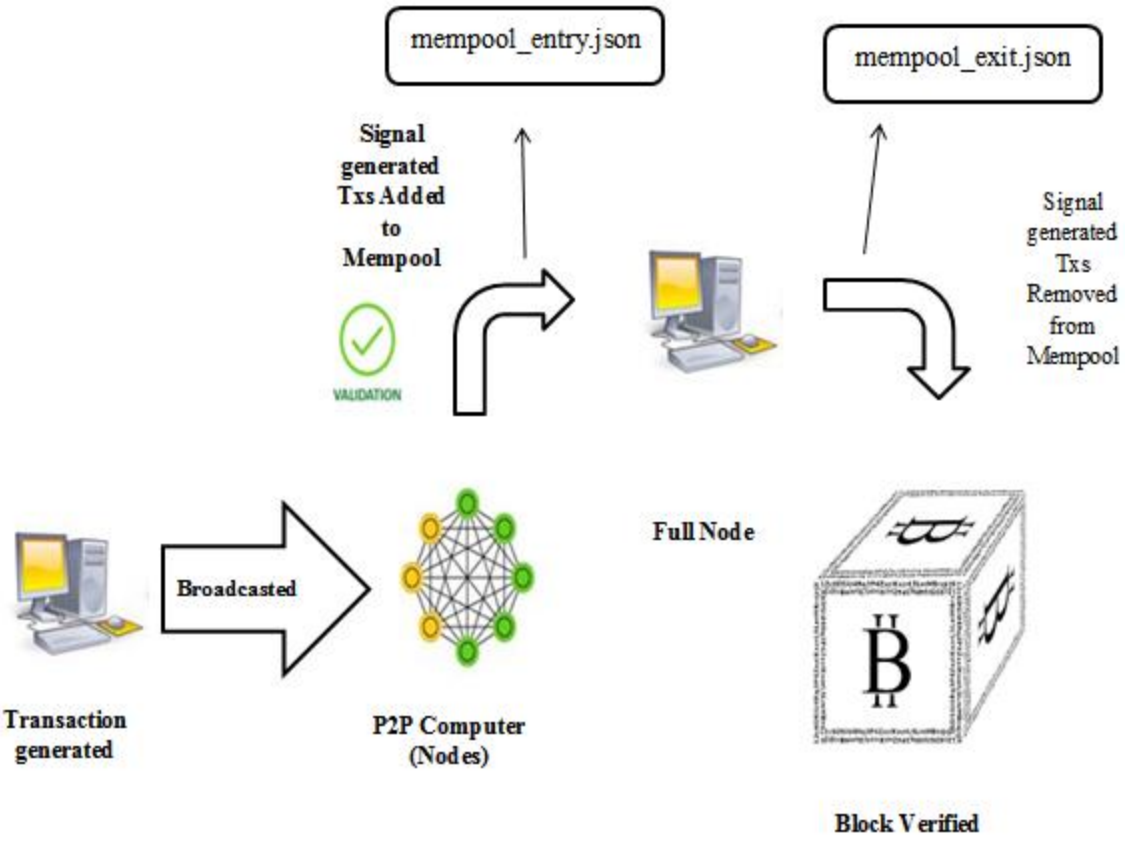


**Figure 8 mempool_entry & mempool_exit json files generation process**

### 3.2.1.3 BlockConnected

BlockConnected Signal handler notifies Block is created and provides a list of transactions evicted. So handler writes a hash of that Block and creation time is log into block_time.json file.

Now we have mempool_entry.json file which contain all incoming transactions, mempool_exit.json file which contain all exiting transactions and block_time.json file which contains block, their timestamps and transaction fee.

## 3.3 Preprocessing (Mempool image creation)

In order to create candidate Block, transactions is to be selected in such a way that their Block transactions Fee is maximized. For this these transactions will be chose from the pool similar to the Miner's pool of transactions.

Our preprocessing algorithm takes mempool_entry.json, mempool_exit.json and block_time.json as input and generates Miners Mempool at that instance of time at which the actual block was created. For this purpose Sets difference (A - B) is used.

In mathematics, a Set is a collection of distinct items. In our algorithm transactions are the distinct items for the Set. Preprocessing algorithm generates two Sets:

- Mempool_entry_Transactions is the Set whose members are all the incoming transactions extracted from Mempool_entry.json file till the time of Block creation.
- Mempool_exit_Transactions is the Set whose members are all the exiting transactions extracted from Mempool_exit.json file till the time of Block creation.

After Sets creation, Mempool of Miner is generated by using Sets difference (A - B). As shown in the Venn diagram.

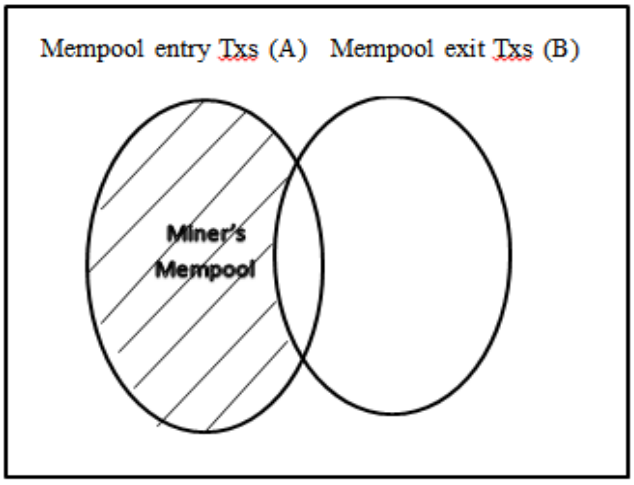


**Figure 9 venn diagram A-B**

The set difference A-B is defined by

$$A - B = \{x: x \in A \text{ and } x \notin B\}$$

Mempool image contains all the incoming transactions but it is possible that some of the transactions have been thrown out of Mempool by Node (Miner) due to the above mentioned reasons. So a transaction belongs to Set Mempool_entry_Transactions but does not belong to Mempool_exit_Transactions Set. Those selected transactions are the transactions that are included in the Miners Mempool.

## 3.4 Algorithm for profitable Block creation

Some of the major challenges a miner has to face involves: (1) Speedy mining hardware, (2) Economical electricity. And (3) start trying your luck by competing with other miners. But there are some other interesting strategies every miner should consider before they select which blocks to work on for mining. Some of strategies are mentioned in (Bonneau, Miller, et al. 2015) , (Carlsten, et al. 2016) and (Xu, et al. 2016) are:

i. There are lot of outstanding transactions in Mempool waiting to be included in the block whereas block size (1 MB ) is limited and limited number of transactions could be include in the block. So miners have to decide which transactions should be included in a block. Miner by default includes any transaction having transaction fee higher than some minimum transaction fee.
ii. As we know Bitcoin is decentralized peer to peer network, it's quite possible that more than one miner mine blocks simultaneously (or with few time difference) and broadcast it over network. So we can say that temporarily a fork has been taken place. in the following case, Miner have to take decision which block to choose and extends the chain because both will be the longest chain in history.
iii. Miners decide when to find and to announce new blocks.

Since default bitcoin client (Bitcoin Core) is being run by most of the miners (over 90%), so a default strategy is implemented. Default bitcoin client run by over 90% of fully validating nodes. It's not clear at what extent these default bitcoin clients have mining power but it is safe to deduce that they are in majority. So default strategies are been implemented by most to the

miners. What if miner changes the some of strategies? Can miner get more profit? (Bitcoin mining is NP-hard by joseph Bonneau and also find other references)

Miners before mining, select transactions from its Mempool and create 1 MB block, called pre Mining. The owner of the transaction payoffs miners some incentives (Transaction fee) so that Miner selects transaction for mining. These transactions have their own size (due to the number of inputs and outputs). As we know block size is finite (1 MB), limited number of transactions will be included in to block. Our algorithm will optimize selection of transitions for block creation such that profit is maximized.

Our algorithm works in the following way:

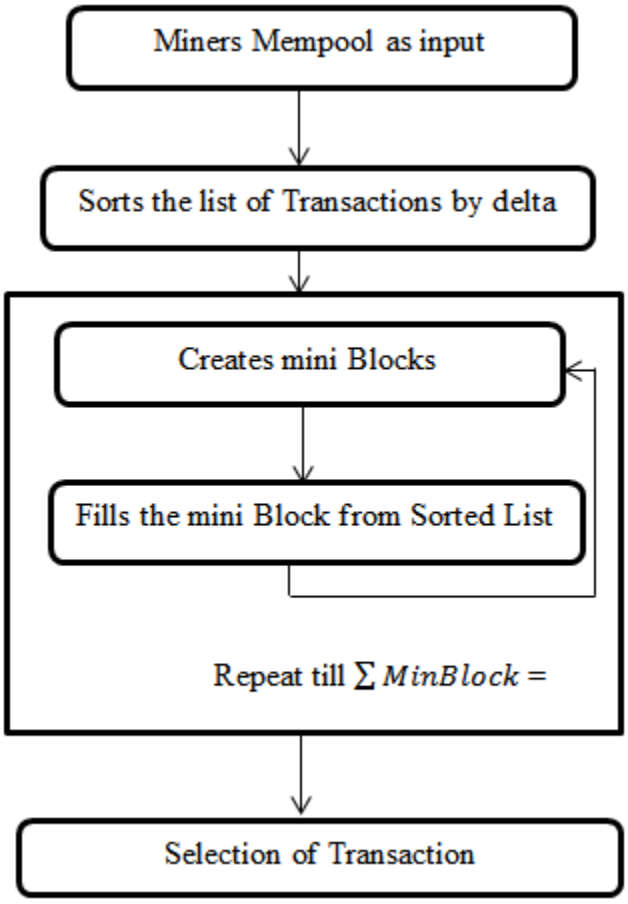


**Figure 10 Block diagram for Proposed Algorithm**

### 3.4.1 Algorithm Proposed

1. **Miners Mempool as input**

   The algorithm takes Miners Mempool (generated by Set difference (A - B)) as input. The average number of transaction in each Mempool is 100600 (approximately). Transaction has multiple parameters (as discussed above in Transaction section). Due to which size of transaction is quite large. So we only extracted Transaction Hash, Transaction fee and Transaction size.

2. **Sorts the list of Transactions**

   In this section we sort the list in descending order by delta **Δ**= tx fee / tx size. The aim of this is to find which transactions yield maximum profit with respect to its size. Transaction will lie higher in the list when transaction fee is greater or transaction size is smaller.

3. **Creates mini Blocks**

   Mini Blocks of size lambda λ = **μ** * Block Size was created. The block Size taken is the actual size of a Bitcoin Block. Nowadays it's 1 MB. **μ** is coefficient whose value is 0.25. In fact size of mini Block is 205KB instead of 1 MB. These mini Blocks will add up and generate 1 MB Block.

4. **Fills the mini Blocks**

   Now fill the mini Block using delta **Δ** and if the Transaction size is greater than Mini Block size skip that transaction and again start filling that mini Block. Continue till no further transactions are left to check.

The list of transaction now contain remaining of transactions that are not selected in first place And their places in List are not changed.

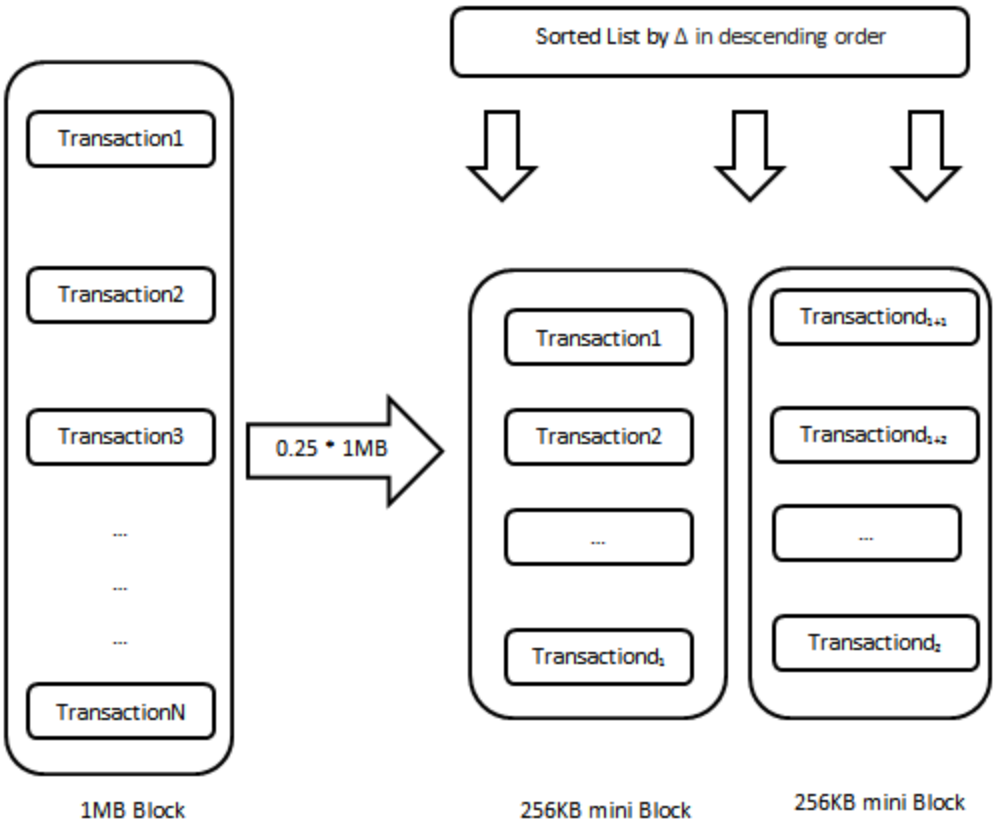


**Figure 11 mini Block creation**

5. **Repeat step 3 and 4**

   The mini Block size is 25% of actual Block so four mini Blocks will be enough to create 1 MB Block but due some internal fragmentation there will be some space left. So we created one extra mini Block. So steps 3 and 4 repeated 5 times.

6. **Selection of Transaction**

   All the Transaction within top four mini Blocks are selected. And remaining space is filled with the transactions having the highest fee in the fifth block. This is done by sorting the transactions in fifth mini block in descending order of transaction fee. Then pick those transactions which can fit in Block.

# CHAPTER 4:

# EXPERIMENTS AND RESULT

# Experiments and Results

## 4.1 Dataset

Node is first synchronized and transactions are logged for two full weeks. Because Bitcoin protocol says that by default transaction can survive for two weeks in the Miner's Mempool. After two weeks the Node's Mempool will be similar to the Miner's Mempool. Our results will be compared with data collected from blockchain.info. This data contains Block's from number 501073 to 501333, whereas block number 501124, 501252 and 501304 are not included in the dataset because these are coin based transaction (only one transaction included in the block which were created by miner). "1" corresponds to 501073, "2" corresponds to 501074 and so on. Dataset includes Block Number, Block size, Block fee, Number of Transaction in a Block and Unix Timestamp at which a block is mined. Almost two day of data (249 Blocks). Average Number of transactions in Mempool are 100600.

Unix Timestamps are used for Mempool image creation. 249 sets of Mempools are created. These Mempools are the inputs for greedy algorithm and algorithm proposed in this thesis. These algorithms are run on 2.7 GHz Intel Core i5 having 8 GB RAM using java code.

## 4.2 Algorithm Analysis

The proposed algorithm is compared with Greedy Algorithm. Greedy Algorithm is the fastest algorithm to solve Knapsack problem (Della Croce et al., 2018). It sorts the list of transactions delta Δ= fee/size in descending order. Then selects transactions till no more transactions can fit in 1MB Block. So it's Time Complexity is $\Theta(n \log n)$. Whereas our proposed algorithm divides 1MB Block in five mini Blocks (because μ is 0.25) having size of 256 KB. Then it selects transactions from list (sorted in descending order by delta Δ= fee/size) and fits it each 256 KB mini Block till no more transactions fit in each Block. First four mini Blocks are selected by default whereas list of transactions in fifth mini Block is sorted by the fee of transaction, and

tries to fit transactions from fifth mini Block. The time complexity of our proposed algorithm is computed in the following way:

$n \log n + n + (n-d_1) + (n-d_1-d_2) + (n-d_1-d_2-d_3) + (n-d_1-d_2-d_3-d_4) + d5 \log d5 + C$

1. n is total number of transactions in Mempool.
2. $d_1$, $d_2$, $d_3$, $d_4$ and $d_5$ are number of transactions included in 1st, 2nd, 3rd, 4th and 5th mini Blocks respectively.
3. $d_1 + d_2 + d_3 + d_4 = C$ is number of transactions in 1 MB Block.

If $3*d_1 + 2*d_2 + d_3 = X$, and $d_1 + d_2 + d_3 + d_4 = C$

$n \log n + 5n + d_5 \log d_5 + X - C + C$

So the time complexity becomes:

$\boldsymbol{\Theta (n \log n + 5n + d_5 \log d_5 + X)}$

$n >> d_5$ i.e. $n \approx 100{,}000$ & $d_1, d_2, d_3, d_4$ & $d_5 \approx 1000$

$\boldsymbol{\Theta (n \log n)}$

## 4.3 Experiments and Results

Here Blocks from 501073 to 501333 are under consideration.

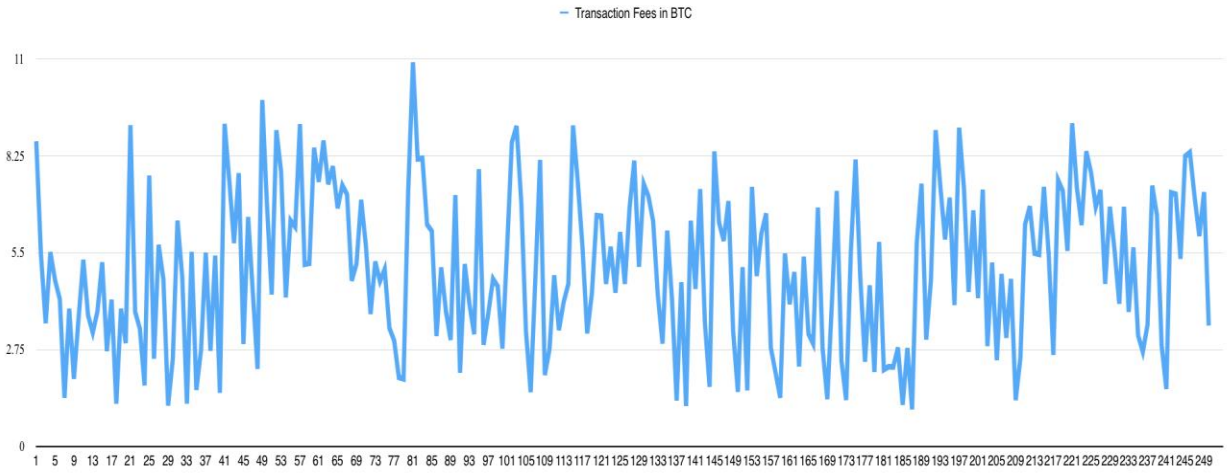


**Figure 12 Transaction fee in BTC**

### 4.2.1 Greedy Algorithm

When Greedy algorithm is applied on our dataset, it produces better results, which indicates that no optimization algorithm is applied earlier while creating a Block in Bitcoin network.

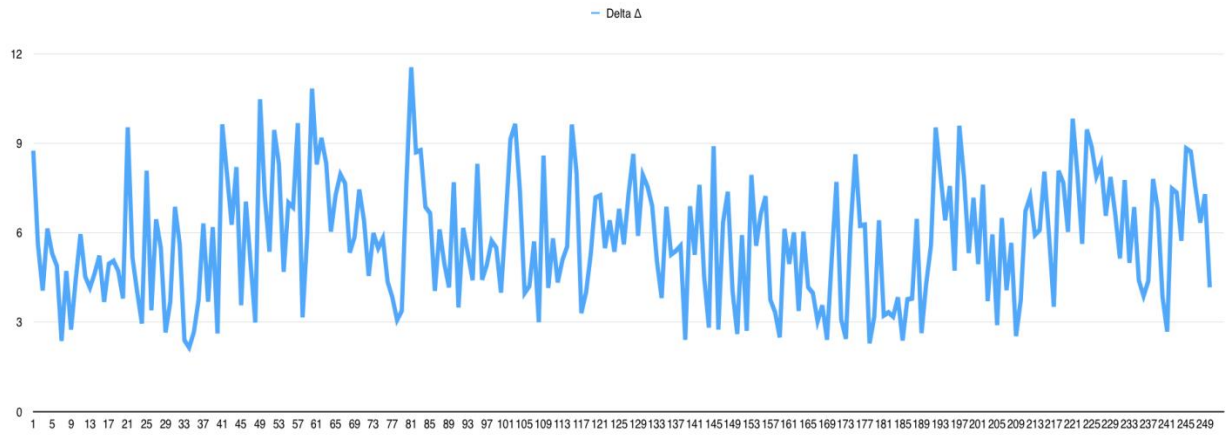


**Figure 13 Greedy Algorithm result**

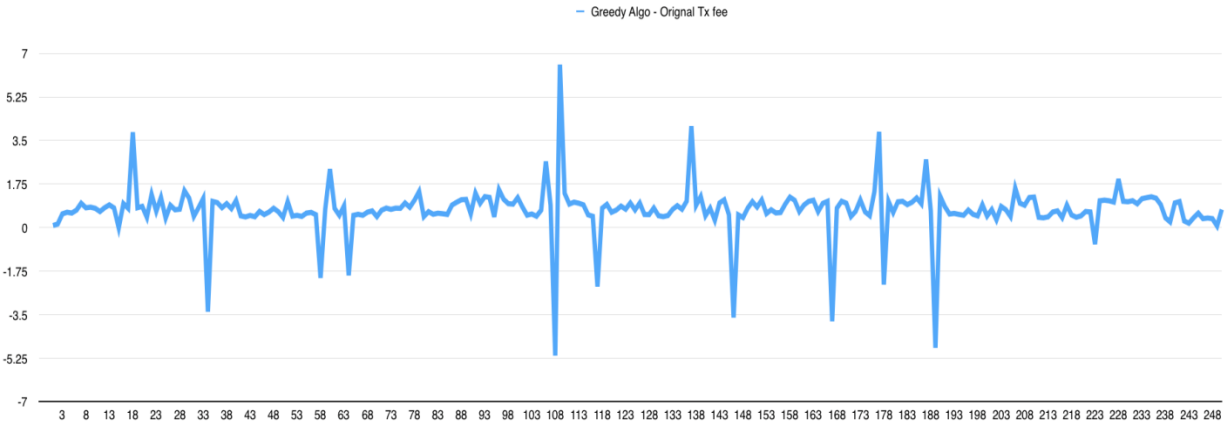


**Figure 14 Greedy Algorithm fee - Original Block fee**

Above graph is obtain by subtracting fee's obtain from Greedy Algorithm and Original Block. Out of 249, 239 times greedy algorithm performed better.

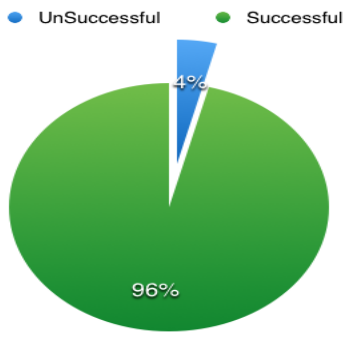


Figure 15

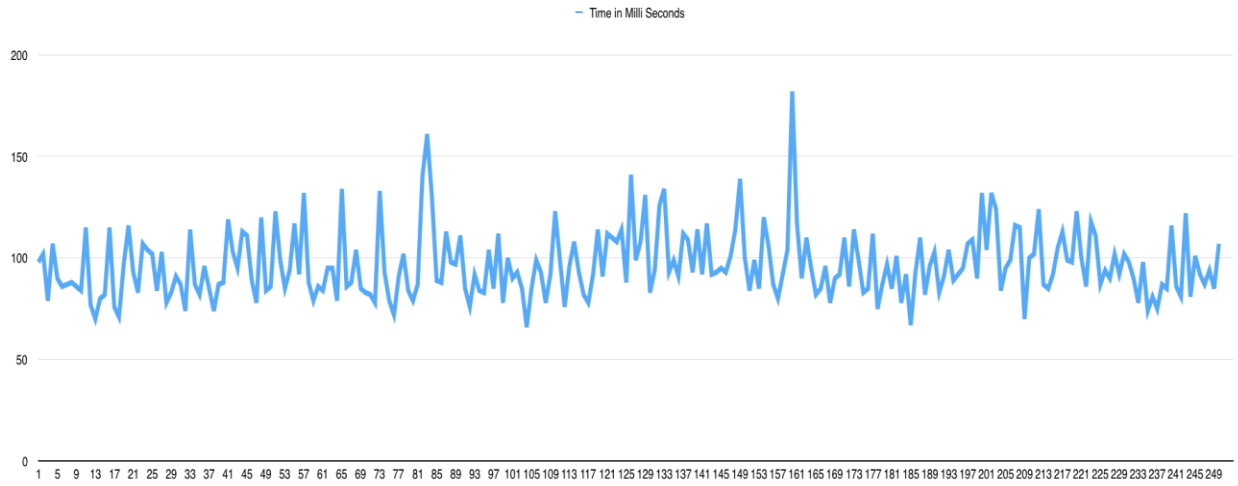


**Figure 16 Greedy Algorithm's time in Millisecond**

Average time taken by greedy algorithm is 97millisecond. So we conclude that greedy algorithm preformed 96% better in quite near real time.

Next graph (Figure 17) shows our proposed algorithm having different μ={0.02, 0.4, 0.05, 0.06, 0.08, 0.10, 0.12, 0.14, 0.15, 0.16, 0.18, 0.20, 0.22, 0.24, 0.25, 0.26, 0.28, 0.30, 0.32 and 0.35} with respect to Greedy algrothm. y axis shows number of blocks.

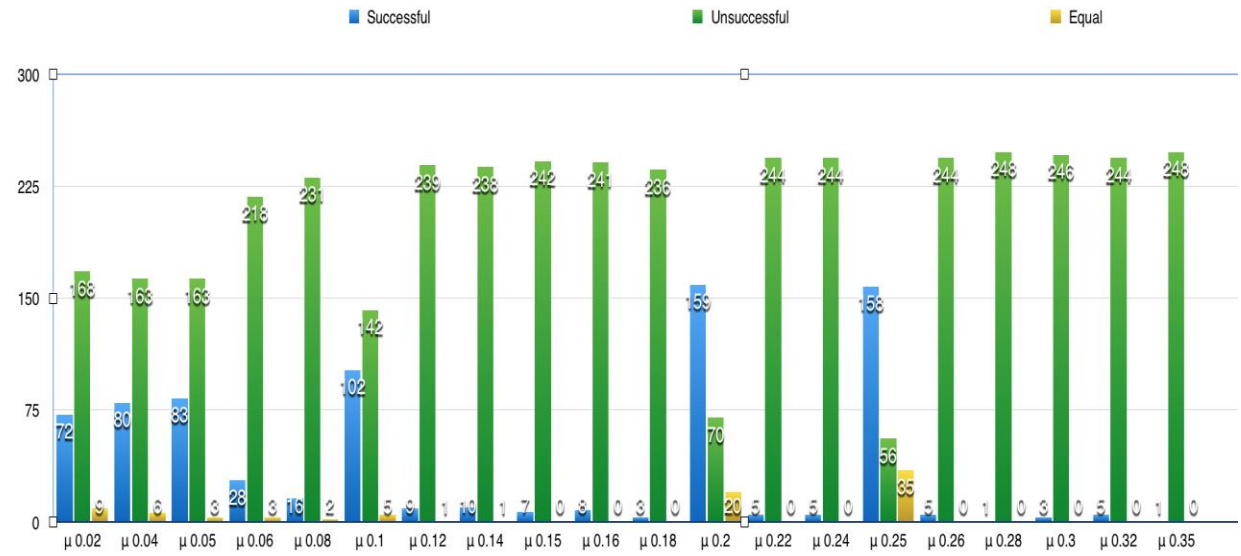


**Figure 17 μ ={0.02, 0.4, 0.05, 0.06, 0.08, 0.10, 0.12, 0.14, 0.15, 0.16, 0.18, 0.20, 0.22, 0.24, 0.25, 0.26, 0.28, 0.30, 0.32 and 0.35}**

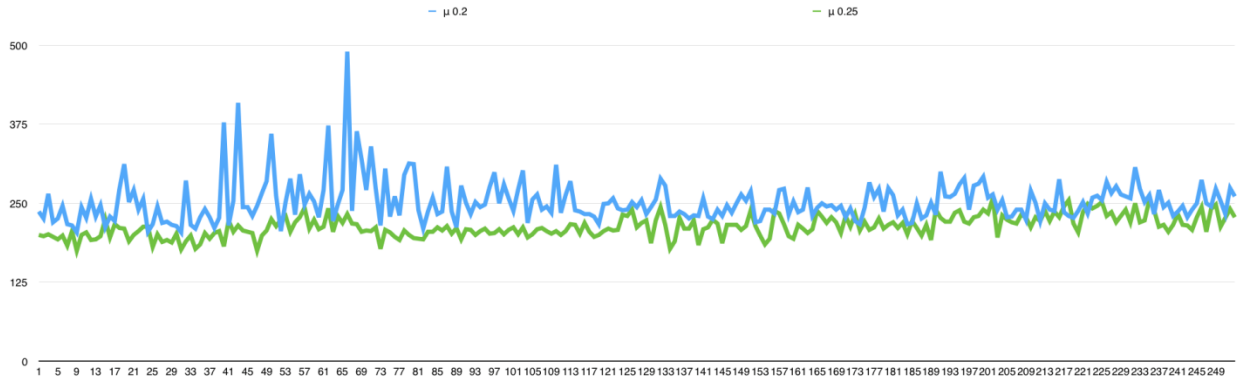


**Figure 18 μ= {0.2 and 0.25} Time in millisecond**

For μ 0.2 and 0.25 results are better than rest. But running time of μ 0.25 took less time as respect to μ 0.2. So we conclude when we take μ 0.25 it will optimize the Block Reward.

### 4.2.2 Actual Transactions v/s μ 0.25

When we compare fees calculated by our proposed algorithm (μ= 0.25) with actual Transaction fees, we get the below mentioned graph.

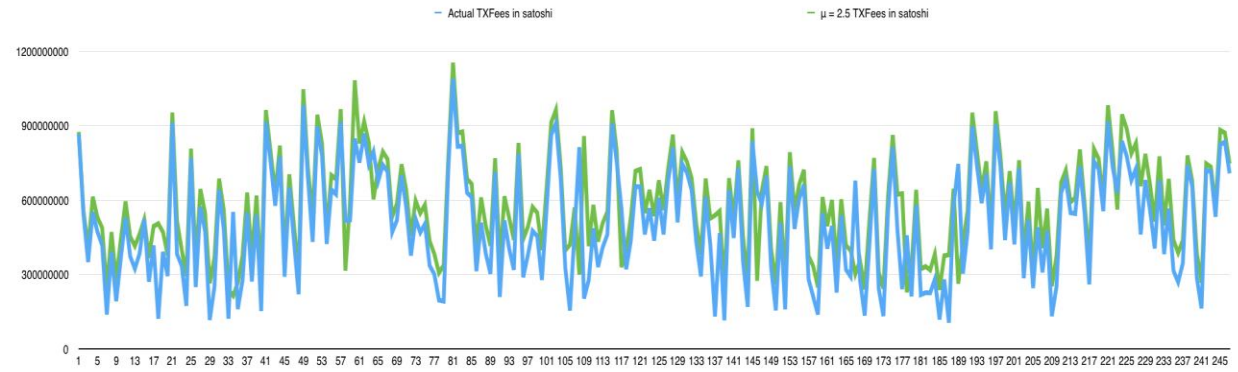


**Figure 19**

By carefully analyzing this graph, it shows that our proposed algorithm's performance is better in regard to maximum time.

Our purpose is not only to optimize Transaction fee but also the number of transactions in 1MB Block.

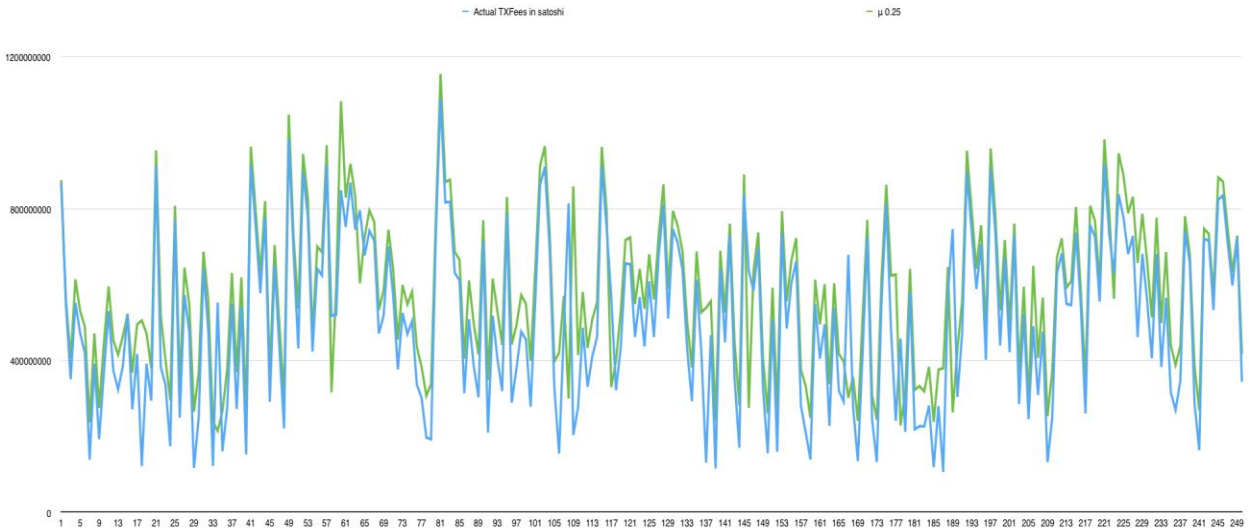


**Figure 20 Blue:Actual Transactions fee Green: μ 0.25**

Our algorithm wraps more Transactions. Originally on an average 1916 transactions were selected in 1 MB Block. For μ is 0.25 our algorithm on an average picks up 2560 transactions in 1 MB Block as shown in figure 21.

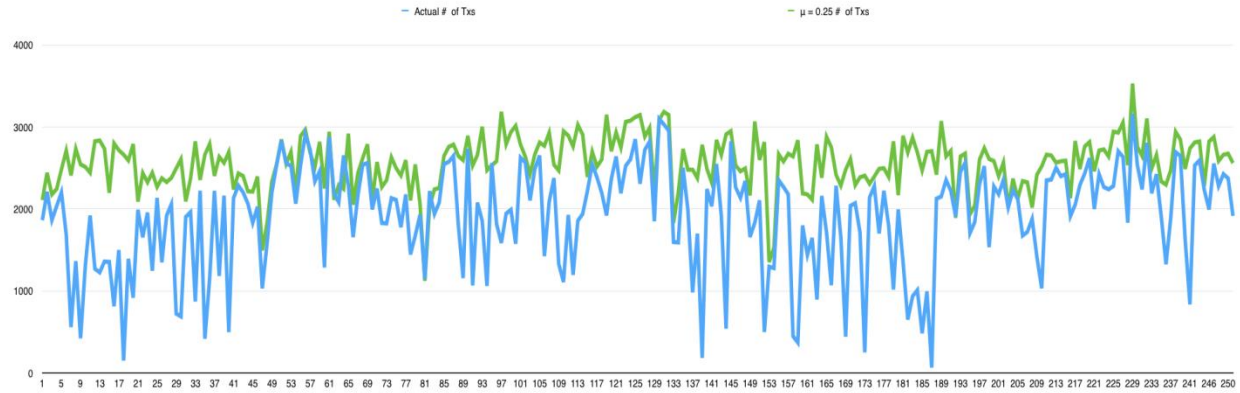


**Figure 21**

Undertaking this, our algorithm utilizes maximum space as shown in figure 22. It leaves less than 200 bytes space only.

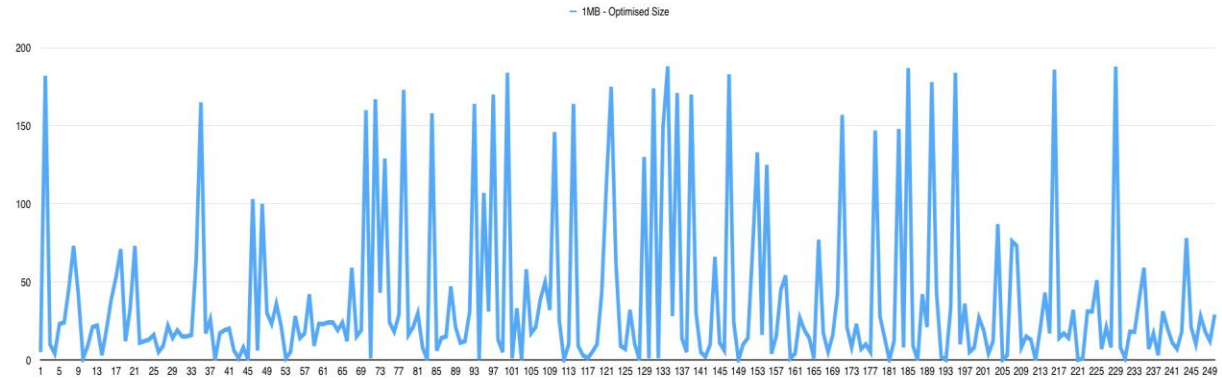


**Figure 22**

# Chapter 5:

# CONCLUSION AND FUTURE WORK

# Conclusion

## 5.1 Outcome of Research

Miners select transaction from their Mempool to create 1MB Block. The problem we are trying to solve is similar to the knapsack problem, which is NP hard. The properties of our problem are to maximize **profit** in **real time**. Our observations show that Miners have not used any pragmatic optimizing algorithm. Conferring our experiments, the results obtained in figure 13, 14 and 15 shows that Greedy Algorithm, which has time complexity of $\boldsymbol{\Theta (n \log n.)}$, produces better results. This demonstrates that if miners apply greedy algorithm as an optimization algorithm for the selection of transactions, it will be quite profitable.

Our proposed algorithm divides the 1MB Block in to five mini Blocks (256 KB each) and fills it with the list of transactions sorted by delta $\Delta$= fee/size in descending order. Our algorithm optimizes miner's fee 239 times out of 249 sets of Mempools. It also produces better results than Greedy algorithm despite the fact that time complexity remains same $\boldsymbol{\Theta (n \log n)}$. Our algorithm not only optimizes miner's fee but also utilizes maximum space of 1MB in near real time.

### 5.2 Future Work

#### 5.2.1 Limitations

- Dataset used in this study contains 249 Sets of transactions (Mempool images). Each set contain on an average of 100,000 transactions. Due to exponential increase of usage of Bitcoin, the transaction number will increase drastically.
- Algorithm proposed considers 1MB Block size.
- Only Bitcoin Transaction is under consideration.

#### 5.2.2 Future Work

Following are the recommendations to further extend this research:

- Larger Dataset (i.e. 1000 Sets of transactions) can be used to get better results.
- Block of larger sizes (2MB and 4MB Block) can be used.

- Litecoin, Dash, Ripple and Ethereum etc. transactions should also be considered.
- Our proposed algorithm optimizes Miner's Reward to few thousands satoshi. More precise algorithms can be used.

# Bibliography


Ammous, Saifedean Hisham. "Blockchain Technology: What is it good for?" *Available at SSRN*, 2016.

Bertsimas, Dimitris, and Melvyn Sim. "Robust discrete optimization and network flows." *Mathematical programming*, 2003: 49-71.

Bonneau, Joseph. *Bitcoin mining is NP-hard.* October 2014. https://freedom-to-tinker.com/2014/10/27/bitcoin-mining-is-np-hard/.

Bonneau, Joseph, Andrew Miller, Jeremy Clark, Arvind Narayanan, Joshua A. Kroll, and Edward W. Felten. "Sok: Research perspectives and challenges for bitcoin and cryptocurrencies." *Security and Privacy (SP).* IEEE Symposium on, 2015. 104-121.

Caprara, Alberto, Hans Kellerer, Ulrich Pferschy, and David Pisinger. "Approximation algorithms for knapsack problems with cardinality constraints." *European Journal of Operational Research*, 2000: 333-345.

Carlsten, Miles, Harry Kalodner, S. Matthew Weinberg, and Arvind Narayanan. "On the instability of bitcoin without the block reward." *In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security.* ACM, 2016. 154-167.

Chen, Qing, and Yuxiang Shao. "A hybrid genetic algorithm to solve zero-one knapsack problem." *In International Conference on Applied Informatics and Communication.* Berlin: Springer, 2011. 315-320.

Croce, Della, Federico, Ulrich Pferschy, and Rosario Scatamacchia. "Approximating the Incremental Knapsack Problem." arXiv preprint arXiv:1801.04801, 2018.

Dwork, Cynthia, and Moni Naor. "Pricing via processing or combatting junk mail." In Annual International Cryptology Conference, pp. 139-147. Springer, Berlin, Heidelberg, 1992. "Pricing via processing or combatting junk mail." *In Annual International Cryptology Conference.* Berlin: Springer, 1992. 139-147.

Eyal, Ittay. "The miner's dilemma." *In Security and Privacy (SP).* IEEE, 2015. 89-103.

Gopalan, Parikshit, Adam Klivans, Raghu Meka, Daniel Štefankovic, Santosh Vempala, and and Eric Vigoda. "An FPTAS for# knapsack and related counting problems." *In Foundations of Computer Science (FOCS).* IEEE, 2011. 817-826.

Johnson, Benjamin, Aron Laszka, Jens Grossklags, Marie Vasek, and Tyler Moore. "Game-theoretic analysis of DDoS attacks against Bitcoin mining pools." *In International Conference on Financial Cryptography and Data Security.* Berlin: Springer, 2014. 72-86.

Kellerer, Hans, Ulrich Pferschy, and David Pisinger. "Knapsack problems." *Springer*, 2003.

Lee, Chungmok, Kyungsik Lee, Kyungchul Park, and Sungsoo Park. "Branch-and-price-and-cut approach to the robust network design problem without flow bifurcations." *Operations Research*, 2012: 604-610.

Matonis, Jon. "The bitcoin mining arms race: Ghash. io and the 51% issue." 2014.

Maxwell, Gregory. *CoinJoin: Bitcoin privacy for the real world.* August 2013. bitcointalk.org.

Monaci, Michele, Ulrich Pferschy, and Paolo Serafinix. "Exact solution of the robust knapsack problem." *Computers & operations research*, 2013: 2625-2631.

Narayanan, Arvind, Joseph Bonneau, Edward Felten, Andrew Miller, and Steven Goldfeder. *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction.* Princeton University Press, 2016.

Ren, Ziwu, and Ye San. "A Hybrid Optimized Algorithm Based on Simplex Method and Genetic Algorithm." *In Intelligent Control and Automation* (IEEE) 1 (2006): 3547-3551.

Rolfe, Timothy J. "An alternative dynamic programming solution for the 0/1 knapsack." *ACM SIGCSE Bulletin 39.* 2007. 54-56.

Srinivas, Mandavilli, and Lalit M. Patnaik. "Adaptive probabilities of crossover and mutation in genetic algorithms." *IEEE Transactions on Systems, Man, and Cybernetics.* IEEE, 1994. 656-667.

Teo, Jason. "Self-adaptive mutation for enhancing evolutionary search in real-coded genetic algorithms." *ICOCI'06.* IEEE, 2006. 1-6.

Tharanipriya, P. G., and P. Vishnuraja. "Hybrid genetic algorithm for solving knapsack problem." *In Information Communication and Embedded Systems (ICICES).* IEEE, 2013. 416-420.

Tschorsch, Florian, and Björn Scheuermann. "Bitcoin and beyond: A technical survey on decentralized digital currencies." *IEEE Communications Surveys & Tutorials*, 2016: 2084-2123.

Xu, Xiwei, et al. "The blockchain as a software connector." *In Software Architecture (WICSA).* IEEE, 2016. 182-191.

Zou, Dexuan, Liqun Gao, Steven Li, and Jianhua Wu. "Solving 0–1 knapsack problem by a novel global harmony search algorithm." *Applied Soft Computing 11*, 2011: 1556-1564.