NTNU
Norwegian University of
Science and Technology

SpaceMint

A Cryptocurrency Based on Proofs of Space

Trond Hønsi

Master of Science in Communication Technology
Submission date: August 2017
Supervisor: Danilo Gligoroski, IIK

Norwegian University of Science and Technology
Department of Information Security and Communication Technology

Title: SpaceMint: A Cryptocurrency Based on Proof of Space

Student: Trond Hønsi

Problem description:

SpaceMint, is a new cryptocurrency introduced in 2015. It is based on proofs of space, instead of wasteful proofs of work. Mining in SpaceMint is designed to have low setup and overhead costs, yielding a fairer reward structure for small and large miners. Miners in SpaceMint dedicate disk space rather than computation.

The goals for this master thesis is to study in details the concepts of SpaceMint, SpaceMint mining and to create a proof of concept environment for SpaceMint mining.

Responsible professor: Danilo Gligoroski, ITEM

Supervisor: Danilo Gligoroski, ITEM

Abstract

While cryptocurrencies are ravaging the world, and people are drooling over them like vultures. Eyeing the opportunity to earn a few bucks, without any thought of the toll that it puts on the environment. Mother earth is screaming for a break, from the everlasting destruction that humanity brings.

This paper will present two cryptocurrencies, one which is the most known in the world, and a new solution. This new solution, suggests at decentralized currency which is offering all the functionality bitcoin has, but with far less energy consumption. The paper will also demonstrate a proof of concept of this coin, and discuss if there is a future for the new guy.

Mens kryptovaluttaer herjer og folk tjener millioner på trading og mining av bitcoin og andre altcoins, skriker værden etter fred og ro fra det evige miljøsvineriet som er menneskeheten.

Denne oppgaven tar for seg 2 kryptovaluttaer, den ene som er mest utbredt(bitcoin) og et nytt forslag. Det nye forslaget foreslår en ny myntenhet som skal kunne tilby alle godene som bitcoin har, bare at de kan skaffes med bruk av mye mindre elektrisitet. I løpet av oppgaven vil det bli presentert en utviklet demo av valuttaen, som så blir drøftet.

Preface

This Master's thesis is was done in the months between March and September at NTNU in the year 2017.

SpaceMint, is a new cryptocurrency introduced in 2015. It is based on proofs of space, instead of wasteful proofs of work. Mining in SpaceMint is designed to have low setup and overhead costs, yielding a fairer reward structure for small and large miners. Miners in SpaceMint dedicate disk space rather than computation.

The goals for this master thesis is to study in details the concepts of SpaceMint, SpaceMint mining and to create a proof of concept environment for SpaceMint mining.

Contents

List of Acronyms

BTC Bitcoin.

CC Cryptocurrency.

CPU Central Processing Unit.

GPU Graphics Processing Unit.

HDD Hard Disk Drive.

IP Internet Protocol.

NTNU Norwegian University of Science and Technology.

PoS Proof of Space.

PoW Proof-of-Work.

TCP Transmission Control Protocol.

UDP User Datagram Protocol.

Chapter 1
Introduction

Since its invention in the early 1980s, there has been a lot of divided opinions on digital currencies. From David Chaum's Digital cash conceived in 1983[1] to the numerous altcoins we see today, the topic of a sustainable digital currency has come a long way. These altcoins are all part of the very new term Cryptocurrency (CC), and is one of the new technologies meant to shake up the world as we see it.

A cryptocurrency is a digital asset designed to work as a medium of exchange. It uses cryptography to control its creation and management instead of relying on a central authority or a trusted third party(decentralized).

Something similar was first conceptualized as digitial cash in the 1980s, where a central server was entrusted to prevent double spending [1]. But due to failure to achieve compatibility between centralization, anonymity and the prevention of double spending, its viability was put into question.

Three decades later in November 2008 a paper, written by Satoshi Nakamoto, was sent to a mailing list in the cryptographic community. It described a peer to peer electronic cash system where payments would be sent directly from one party to the other without going through a financial institution. Three months later, in January 2009, the Bitcoin network was created and the first coins came into existence. This thesis will explain how bitcoin came to be, and will shortly describe a new contender to the bitcoin which proposes a more environmental friendly way to handle transactions.

A basic proof of concept of this new coin called SpaceMint will then be presented, and is the main purpose of this thesis. By using a method that uses far less power to generate decentralized value, it might pose a threat to the bitcoin, the Goliath of CC.

Chapter 2
Literacy

2.1 Bitcoin

Bitcoin is the first digital currency that managed to achieve full decentralization. The reason for its origin; to diminish the cost of transactions due to mediation, thereby removing the practical limits on transactions. Henceforth making small casual transactions between any party in the world possible.(Bitcoin: A Peer-to-Peer Electronic Cash System).

To make this possible Bitcoin presented a new technology on its release, the blockchain. A public ledger keeping track of all previous transactions that have been made in the network, while also registering all new additions to come. To understand the cryptocurrencies and the reason for their values, its properties need to be explained further:

- Transactions
- Timestamp Server
- Proof of Work
- Network
- Incentive
- Privacy
- Integrity

2.1.1 Transactions

The coin in Bitcoin (BTC) is an electronic coin formed by a chain of digital signatures. Whenever a coin is spent, the coin is transferred to the new owner. This is done by

appending the chain with a digital signature of the hash of the previous transaction and the public key of the next owner. See figure(...from bitcoin)

2.1.2 Timestamp Server

To handle the data BTC uses a timestamp server. A procedure where a block of items to be timestamped is hashed and then widely published throughout the bitcoin-network(See subsection Network). This ensures the existence of the data at the time of creating the block. By including the timestamp of the previous block in the timestamp of the present block, the server forms a chain, making every new addition reinforce the previous ones. See figure below.

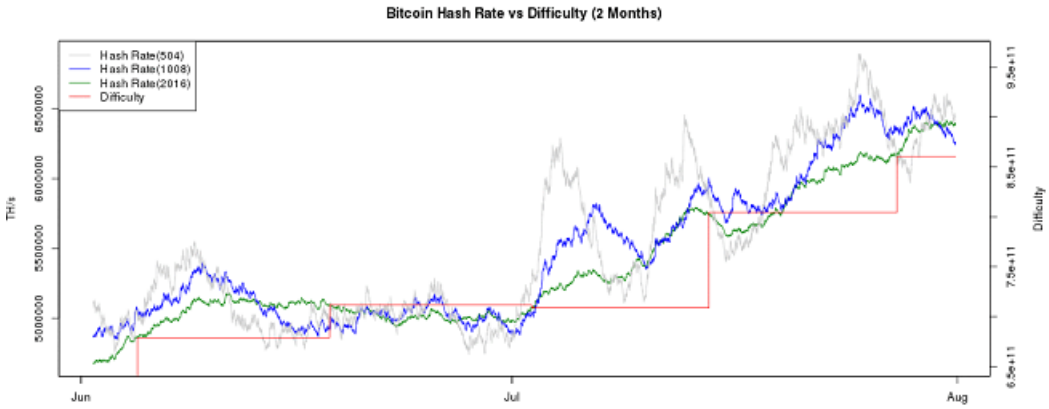
2.1.3 Proof-of-Work

To make a peer-to-peer distributed timstamp server possible, BTC uses a proof-of-work model described in Adam Back's Hashcash(Hashcash - A Denial of Service Counter-Measure). A method involving scanning for a hash value, using a cryptographic hashfunction, where the hash begins with an x number of zero bits. This carries the property of requiring an exponential amount of work with the number of zero bits, while verification can be done by executing a single hash.

The work done on the timestamp network is, by increasing a nonce in the block to a value that gives its hash the required amount of zero bits. (See figure) Thereby after the required Central Processing Unit (CPU) effort has been expended to satisfy the Proof-of-Work (PoW), the block value is determined, and cannot be altered. Altering the block would demand the work to be redone, and as new blocks are appended to the chain, the work of all new blocks also has to be redone.

By using this model, every major decision is determined in a one-CPU-one-vote matter. The longest chain makes the decision, which is the chain that has had the most work invested in it. Only by having one party control more than 50% of the total processing power, will the main chain be compromised. Which is called the "51% attack". This was close to happening in 2014. A mining pool, a group of miners sharing the problem and reward, was close to reaching 51% of the total hashing power. The community responded in the best manner possible, by the miners freely leave GHash.io to prevent disaster [2].

On release, the PoW was intended to be done on a standard CPU to use up left over processing time. However, as bitcoin rose in value, the interest in mining it also rose. Mining bitcoin then evolved to using the Graphics Processing Unit (GPU), which later evolved to using ASICs. A specialized hardware reaching sky high hash rates at low energy cost. Today, profitable mining of BTC is only achieved by access to cheap electricity and is mostly done in China[3]. A list of hashrates on standard



computer hardware can be found on[4]. Here the most powerful equipment reaches a few hundred GH/s, while most ASICs operate with speeds 10 times that.

To handle the increase in hardware speed, and increasing mining interest, the PoW difficulty is determined by a moving average number of blocks per hour. If the blocks are generated too fast, the difficulty is increased. This is done to keep the block generation occurrences at 1 block every 10 minutes. From the figure you can below you can see a clear connection between hashrate and difficulty.[5]

2.1.4 Network

For the Bitcoin network to run as smoothly as possible, every node need to follow a few predetermined steps:

1. New Transactions are broadcast to all nodes.
2. Each node collects new transactions into a block
3. Each node works on finding a difficult PoW for its block
4. When a node finds a PoW, it broadcasts the block to all nodes.
5. Nodes only accept the block if all transactions in it are valid and not already spent.
6. Nodes express their acceptance of the block by working on creating the next block in the chain, using the hash of the accepted block as the previous hash.

For the currency to be sustainable, all nodes in the network need to agree on the same chain. This is done by always mining on the longest chain. If multiple nodes

were to broadcast different versions of the next block simultaneously, the nodes will work on the one they received first, and we have a fork in the network. The fork will seize to exist when a block is found on either of the chains, and all the nodes will then again switch to the longest one. When making new transactions, a node need only to reach many nodes not all. The transactions will end up in a block if not in the first, it is guaranteed to end up in one of the next. Broadcasting blocks is also resilient against dropped messages. If a node does not receive a block, it will request it when receiving the next one and seeing it missed one.

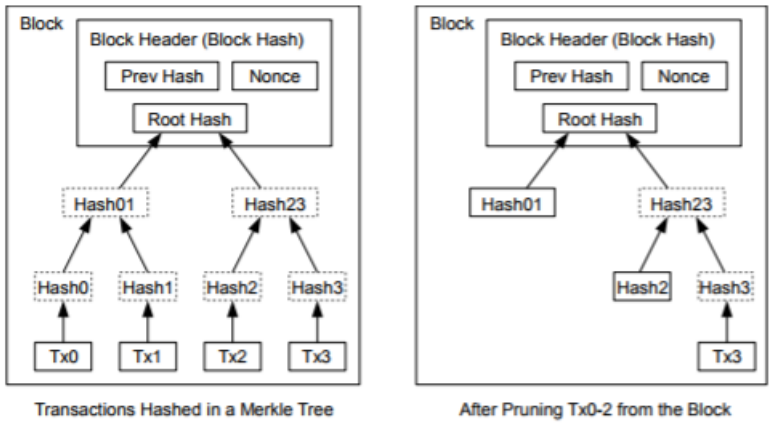
2.1.5 Incentive

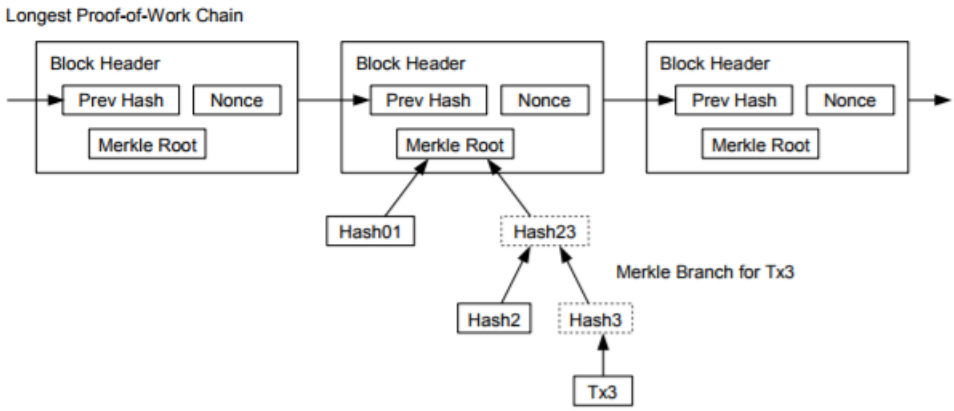
To keep the parties interested in mining BTC, the first transaction in a new block is always reserved for the block creator, called the generation transaction. This transaction keeps record of which Bitcoin adresses or scripts that are entitled to receive the block finding reward. The amount of Bitcoins generated per block started at 50, and is halved every 210,000 blocks[6]. With the current block count of over 420,000, this reward has already been halved twice, thus the mining reward now being 12.5 coins for every new block[7]. Another incentive for the miner is a transaction fee that may be included in the transactions made for a purchase. This fee is voluntary, but it is up to the block creator to choose if he wants to include the transactions in his new block. This fee is called a miner's fee, and is normally seen as a way to pay for miners to put your transaction on the block. Due to space on a bitcoin block being scarce, this is normally controlled by supply and demand. The market asks for a tradeoff between confirmation time and cost. Users wanting fast confirmation normally pays more than more patient parties[8].

2.1.6 Space management

To handle the massive amounts of data that eventually show up, there needs to be a procedure to save disk space. This is done by discarding previous transactions on a coin after reaching a certain blockchain milestone. Keeping the transactions hashed in a Merkle Tree(Merkle Hash Tree based Techniques for Data Integrity of Outsourced Data) and only inlcuding the tree root in the block, makes this possible without breaking the block's hash. Old blocks can then be compacted by pruning the merkle trees.(see figure)

By doing so, one only need to store the block headers in memory, which is around 80 bytes each. In June 2017 the number of blocks mined passed 470 000[9], making the total size required to carry all block headers close to 36 MB. Compared to the complete block chain which today has a size of over 140 GB[10].



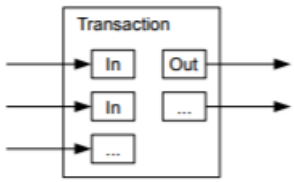


2.1.7 Payment Verification

Verifying payments in the network is easily done, as long as it is run by honest nodes. For a user to check a transaction, it only needs to be in possession of the block headers, and is not required to run a full network node. The block headers of the longest BTC chain can be gathered by querying the network. By doing this, the Merkle branch linking the transaction to the block can be obtained. However, the transaction cannot be checked, but by linking it to the chain, the transaction is accepted by the network and confirmed by further appended blocks. (see figure)

2.1.8 Coin Handling

To save the system for overflowing the system with transactions handling each coin individually, it is made possible to combine or split coins in a single transaction. A transaction takes multiple inputs, but have only two outputs. The transaction can



then combine many small payments to a bigger party, and then send the change back to the sender in the second output. See figure below.

2.1.9 Privacy

While banks and other financial institutions are required by law to reveal information about their clients if the proper authority demands it, (https://www.dnb.no/en/about-us/protection-of-personal-privacy.html) this is not a case when dealing with BTC. All transactions are allready public in the blockchain. However, the bitcoin-paper describes that privacy can still be maintained. This is made possible by regulating information flow on a different point, by keeping the public keys anonymous. The public ledger tells tales of a public identity sending another identity some coins, but by not knowing the who the owner of the identities privacy can be maintained.

2.2 Spacemint

SpaceMint is a proposed altcoin(http://www.investopedia.com/terms/a/altcoin.asp) from the paper Spacemint: A Cryptocurrency Based on Proofs of Space by Albert Kwon et al. It builds on the blockchain-technology from bitcoin, but instead of the wasteful PoW, uses a new protocol based on Stefan Dziembowski's Proof of Space (PoS)(https://eprint.iacr.org/2013/796.pdf) Although many of the cryptocurrency attributes are reused, some new functionality has been added to make SpaceMint viable.

2.2.1 Proof of Space in SpaceMint

A PoS-scheme is set up by two parties, a prover P and a verifier V. Here the prover P is trying to convince the verifier that he is storing some amount of data S_y. The verifier on the other hand needs to store a commitment valueγ which he can use for when he needs to send a challenge to the prover.

For SpaceMint the prover stores a special directed acyclic graph[11] called a hard-to-pebble graph for its committed space, where each node contains a hash. The algorithm for committing space is listed as Algorithm 1, section 3 in the SpaceMint

article [12]. Here the prover generates a unique nonce μ that he uses to compute the commitment value γ from the hard-to-pebble graph. γ is a Merkle-tree root to the graph stored on the prover's harddisk, and is sent along with μ to the verifier. The total space S_y which then is stored is $N = 2 * n * L(graph + Merkletree)$ Where n is the number of nodes, and L is the length of the hash function used.

There are two additional algorithms described for use in SpaceMint; Algorithm 2, the commitment verification, and Algorithm 3, Prove Space[12]. In Algorithm 2, verifier V samples a tuple of challenges $c = (c_1, ...c_{k_v})$, which are sent to the prover in a challenge function Chal(n, k_v, \$). P then computes openings $b := (b_1, b_2, ...)$ of all the nodes that are being challenged and all their parents. These are sent to V through answering function Ans. The verifier then uses this information in a verfication function Vrfy(μ, γ, c, a), which returns 1 if prover provided the proper answers. The third algoritm, Prove space, is a more simplified version of Algorithm 2, and is about just a way to prove that the space is still being held after a certain amount of time after the commitment verification has taken place.

These 3 algorithms form the required methods for the two phases of PoS in SpaceMint:

- **Setup Phase**
  - The space is committed through Algorithm 1-
  - The commitment is verified through Algorithm 2.
- **Online Phase**
  - The space is continuously checked that it is still at the prover through Algorithm 3.

2.2.2 The Spacemint Protocol

Like every other CC, SpaceMint also builds on the technology that BTC proposed. Some work is done for a block to be added to a blockchain, and then the finder of the block is rewarded with some coins. However, some aspects of the SpaceMint protocol acts differently than your traditional PoW-schemes.

2.2.3 Mining

The process which keeps the system running. For transactions to be reccorded and added to the blockchain, new blocks need to be added to the ledger. Mining, as in bitcoin, revolves around solving a problem for the network. SpaceMint has two ways of rewarding miners for adding blocks and transactions to the blockchain. The first

one is handing out fresh coins for finding and adding the block to the chain. The size of the reward, as in bitcoin, is decided by some standard. The second one is a mining fee, where the sender can add a chosen amount which will be granted to the miner adding the transaction to the new block. To begin the process of mining, there are as in the PoS-concept described in [12], two phases to go trough.

Initialization. SpaceMint uses a secure signature scheme for handling identification over the network, and consists of the following functions; `SigParamGen, SigKeyGen, Sign, SigVerify` When a new miner wants to join the network with a contribution of N bits disk space, he first needs to sample a public/secret key pair *pk,sk*. This is done with the signature key generation-function `SigKeyGen` and uses his public key as the nonce μ for Algorithm 1 to generate

$$(\gamma, S_\gamma) := Init(pk, N) \tag{2.1}$$

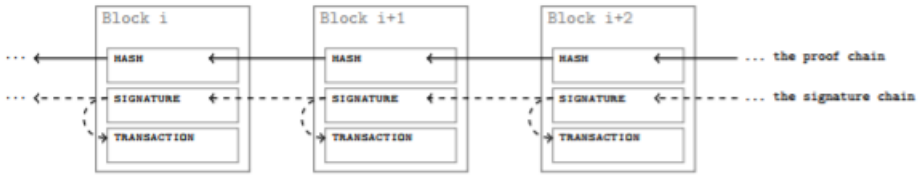
The new miner then stores the space S_y and the secret key `sk` and makes his space commitment (pk,γ public through a transaction described later. When this transaction has been added to the blockchain, the miner is allowed to start the second phase.

Mining. The online phase for which a person attempts to add blocks to the chain every time period. For every time interval the miner follows 5 steps to see if he will be able to make a new block:

1. Retrieves the hash of the current latest block and a challenge c which is derived from a function described later. This challenge is used to produce long random strings $\$_p$,$\$_v$
2. Calculates $(c_1, ..., c_{k_p}) := Chal(n, k_p, \$_p)$ as described in Algorithm 3.
3. He then computes proof $a = a_i, ..., a_{k_p}$ by using Algorithm 3 trough the answer function $a_i = Ans(pk, S_\gamma, c_i)$.
4. The answer is used to generate the qualiy of proof through the quality function $Quality(pk, \gamma, c, a)$ described further down.
5. If the quality is high enough to be the likely best answer for time period i, the challenge $c := (c_1, ..., c_{k_v})$ is computed along with their answers $b = (b_1, ..., b_{|}b|)$ as described in Algorithm 2. Then a block is created and sent to the network. (Note that it is possible to set $k_v << k_p$)

2.2.4 Blockchain format in SpaceMint

The blocks in SpaceMint consists of 2 items. (1) The transactions that have been published since the last block, and (2) the proofs a and b computed in step 5 above.



These blocks are put in a sequence of blocks $\beta_0, \beta_1, ...$ which forms the public ledger containing all the network transactions. For holding that information, the structure of each block β_i is divided into three parts, called "sub-blocks". Every sub-block keeps the index i which specifies the block's position in the blockchain, and some additional information which is connected to that sub-block's function.

- The first block is the Hash sub-block, and contains the following information:
  - the urrent block index i
  - the signature of the miner who made the block on the Hash sub-block with index $i-1$
  - the proof of space along with the miner's pk
- A transaction sub-block:
  - the current block index i
  - a list of transactions
- A signature sub-block:
  - the current block index i
  - the miner's signature on the current transaction sub-block
  - the miner's signature on the previous signature sub-block.

These sub-blocks form links that allways point to the previous block, and is illustrated in the figure below. The chain of hash sub-blocks is called proof chain, and the signature-blocks the signature chain. Also notice that the transaction sub-block is only connected to the its own block index.

In the figure you can see that the hash sub-block is only linked to the proof chain, while the signature sub-block and transaction sub-block are linked together. This creates the possibility of having many signature chains linked to the same proof chain, and consensus in the block chain can be hard to achieve. However, this is fixed whenever an honest miner adds another proof block to the proof chain, and the prior signature chains are immune to alterations.

2.2.5 Transactions in Spacemint

Although some of the transaction part was covered in bitcoin's presentation, SpaceMint also uses some other types of transactions which are to be included into the transaction sub block. There are 3 types of transactions that are used in SpaceMint, some more often than others:

Payment: This one is the most similar one to bitcoin. This function transfers coins from m benefactors to n beneficiaries, where every party is identified by a verification key supported by `SigKeyGen`. This transaction, *ctx*, holds 4 attributes; `payment, txId, in, out`.

- txID is a unique, arbitrary transaction identifier.
- in is a list of triplets $in = in_1, ..., in_n$ with n benefactors. Each input to the list is a triplet $in_j = (txID_j, k_j, sign_j$ where:
  - $txID_j$ is an identifier pointing to a past transaction
  - k_j specifies the beneficiary's public key to the transaction.
  - sig_j is a signature on $txId, txId_j, k_j$,***out***, which verifies the ownership that the beneficiary has on the coins being transferred binding them to the party.
- out is a list of beneficiaries and the amount received: $\texttt{out} = (out_1, ..., out_m$ where $out_i = (pk_i, v_i$
  - pk_i is the public key of the beneficiary, supported by `SigKeyGen`
  - v_i is the amount of coins paid to the beneficiary.

There are two requirements that these transactions need to fulfill in order to be valid:

1. A benefactor can not be named in two subsequent transactions, this to prevent double spending.
2. The amount of coins in the input is greater or equal to the amount in the output

The beneficiary can also be one of the benefactors in a transaction. This is done in order to transfer change back to himself.

Space commitment The transaction for committing space. $ctx = (comit, txId, (pk, \gamma)$

Here the pk is the public key of the person committing space, γ is created in the initiation function `Init(pk,N`. ctx is thus the space commitment of a space of size N. The reason for having this commitment in the blockchain along with the miner's public key is to prevent cheaters from mining on several tasks at the same time using the same pk.

Penalty. A punishing transaction. $ctx = (commit, txId, pk, proof)$ There will always be miners trying to cheat the system. This is a transaction offering incentive to miner's willing to try and catch the dishonest parties. $ctx =$ *(penalty,txId,pk,proof* This transaction carries the familiar transaction id, and public key, but introduces a new attribute. *proof* is the evidence of behaviour that is punishable. The transaction penalizes miners that try to trick the system by mining on multiple chains. Notice here that the first item in every transaction, is a keyword which describes the type of transaction you're currently reading.

2.2.6 Quality of proof

Unlike proof of work where the goal is to be the first one to reach the correct answer, SpaceMint can have several correct answers where the best one is the winner. There is a quality measure on the PoS. For valid proofs $\pi = (pk_1, \gamma_1, c_1, a_1), ..., \pi_m = (pk_m, \gamma_m, c_m, a_m)$ there exists a `Quality(`π_i) for every proof there the ith proof has the probability to have the best quality in correlation to the ith miner's fraction of the total space. Meaning that every period, there will be a best quality option, and is decided by the size of the space committed. A miner allocating a larger space will have a greater probability to generate a high quality proof. The quality is computed by sampling a hash of answer a using a random oracle hash, and then pick out the largest value.

2.2.7 Quality of a chain

In order to prevent hard forks on the block chain, the system uses an value determining the value of the chain. This to determine which of two branches is the highest quality one. This is done in function `QualityPC` which calculates the quality of a chosen number of the latest blocks on the chain. Every block has a proof, and the quality of each block is calculated by the `Quality`-function. To prevent challenge-grinding attacks(mining on several challenges at once), the quality should be calculated by the product of the quality of many consecutive blocks. This is also done in a fair way where more recent blocks are more weighted into the chain than the older ones.

2.2.8 The challenge.

Bitcoin uses the hash of the most recent block to determine the challenge for its miners. For SpaceMint, and other PoSpace-based CC the input challenge c is a bit

harder to design. Although the challenge for SpaceMint is also generated from the chain, three alterations are made to prevent different forms of attacks. By using the most recent block for the challenge consensus on the chain would be slowed down. Many different chains could occur, where miners can get different challenges for different chains, and mine on several chains simultaneously. To counter this, SpaceMint uses block $i - \delta$ for a suiting Δ. The longer the delta, the lesser chance of multiple chains surviving. Another difference, is that only the proof-block is being hashed, so to prevent block grinding attacks. The last difference is that the same challenge is used for multiple blocks, up to δ consecutive blocks. This is done to prevent challenge-grinding attacks.

Chapter 3
Methodology

For this thesis, where creating a proof of concept of SpaceMint was the main goal, three phases had to be conducted. A researching phase, a developing phase, and a testing phase.

3.1 Research phase

This stage was the first couple of months of the project, and gave the foundation to follow through the rest of the project. Starting from scratch, knowing very little about the blockchain and the world of cryptocurrency, i dove in starting with the deep end. The first thing i did was finding the research paper about the proposed SpaceMint and read it from beginning to end. After finishing, and realizing that not much was gained, due too a lot of math and new expressions. It was clear that starting from the beginning and learn of the most basic concepts of cryptocurrency first, would be a much better approach.

I started with the research paper where bitcoin was first presented. This was a lot easier to read than the SpaceMint-proposition, and gave me a much greater picture of how these currencies came to existence and how they work. Some peculiar aspects were the sudden increase in value of both bitcoin and a new up and coming currency Ethereum[13]. While from January 1 bitcoin has almost tripled in value, this newcomer has risen over 3000% [14] [15].

After learning a lot about different types of PoW-based cryptocurrencies, where I even set up a mining rig for ethereum due to immense personal interest and curiosity, i returned to the PoS-based currency SpaceMint which was now much easier to grasp. A new release of the article was made available for me by mail, which had been shortened and polished. I also discovered that the bitcoin source code was available online [16] in $C++$, as well as a finished proof of concept for proof of space in the programming language Go[17] made by Kwon et al[18].

Most of the altcoins that are being made are based on the code from bitcoin with some alterations made. It became clear to me that trying to combine these two

systems to fit the description of the SpaceMint article would be the best possible way of making a proof of concept. Bitcoin had the network functionality needed in C++, and the mining functionality was available in Go.

I tried for a month to set myself into the bitcoin code to extract the network functions. However, after a couple of weeks i deemed that task too great, and decided to go with a different approach.

3.2 Development

Most of the mining functionality was available in the go-code and could almost readily be used in a network benefiting a cryptocurrency. My task was then to set different parties that could communicate with eachother, and run mining operations.

3.2.1 (

Technology) The programming language i chose to use was python. This is by far the language i have the most experience with.Also by using a technology called GoPy [19] the go-code can be converted into CPython[20] extensions that can be called from python.

Python also has a very helpful library called socket, which makes communication trough Transmission Control Protocol (TCP) or User Datagram Protocol (UDP) easy to set up. I went with sockets communicating trough TCP, so that i wouldn't have to deal with packet loss. It is however clear that on a much larger scale peer to peer network, UDP is to be used.

3.2.2 (

The network) The network i produced is a system where every node needs to run both a TCPserver, and a TCP client. The server is always running, ready to accept data about new blocks being added to the system. The TCP client is started every time a block is found on a node, and is then to transmit its findings to two other nodes in the system.

3.2.3 Mining

The mining procedure is mainly handled by the go code, although some functionality had to be added and changed. The foundation to build the graph, get challenges, build a block chain were all there, but not fully developed. So the overhead to build the chain, collect transactions in a block and then add the block to the chain needed to be added. With an extended chain, fetching new challenges for mining also needed to be implemented.

3.3 Testing

3.3.1 Parameters

The testing was done by using 2 different computers. An old laptop i had and a used desktop i bought to also use for ethereum mining:

- Desktop computer - 1 TB
- Laptop - 500 GB

The parameters i used for the mining system were the same as proposed and trialed in the SpaceMint article:

- Δ(Number of blocks back to get challenges from) = 50
- δ(Number of blocks for which challenge nonce will remain the same) = 10
- Λ(Discount factor of older blocks) = 0.9999
- `minspace` = 100(minimum discspace to mine with, in GB)

These are proposed in the article, and are meant to be the best options for preventing different types of attacks as well as being efficient for mining.

3.3.2 Setup

Just as bitcoin had its genesis block(the first block of the chain), the SpaceMint proof of concept also needs its own. While most altcoins just continue on the same chain as bitcoin, due to the differences in mining, SpaceMint starts its blockchain from 0.

To generate the foundation of the blockchain in SpaceMint, the initialization is first run on the desktop computer, where mining is continued until 50 blocks are generated. Initialization is started by calling the `init` function in the python class, which takes the chosen amount of space(here 750 GB) as parameter. The function runs as according to the steps below:

1. Prompts for a password.
2. Generates public and private key pair
3. Runs the init(pk,N)
4. Waits for the graph and Merkle tree to be built.

5. After completion, it runs space commit.
6. Spacecommitment γ is included with the public key and put into a spacecommit-transaction.
7. The commit-transaction is included with a payment transaction(for creating the first block) given index 0 and signed by the user.
8. The signature is added to the signature on an empty set, given the index 1. This forms the first signature block.
9. Adds the index 1, along with a proof = 1, and a signature on an empty set, which becomes the first hash sub-block.

After creating the first block the miner the tcp server is then started, and the node is now seen as a regular miner in the SpaceMint system. It can now start to mine using the regular SpaceMint protocol. Due the number of blocks being less than 50, i had to add the functionality so that it only mines on the previous block. This will only be the chase until the chain reaches 50 blocks. So except for that the hash value is collected from block i-1, the miner now runs as normal and according to the steps procedure described in section 2.2.3.

When the blockchain had reached its 50 blocks marker, which took just under an hour after block 1, i started up the laptop mining with 250 GB. This is done by initializing, and then sending the space commitment along with their public key to the desktop. The desktop stores their Internet Protocol (IP)-adsres and starts to transmit data about the blockchain to the laptop through its TCP client, starting with the current challenge block. Two nodes were now up and running, mining SpaceMint. I let this run for a week.

Chapter 4
Results

After managing to successfully run the system for a week, i stopped it to review results:

Miner 1	Miner 2
7563	2239
14678	4579

This is the number of coins every miner achieved in a measured every week over 2 weeks.

There are also some other remarks to take notice of. When miner 2 was finally ready to start mining, the chain had already reached over 550 blocks.

Chapter 5
Discussion

5.1 Working as intended?

For the first week, the proposed proof of concept generated an amount of a total of 9802 coins. This amount of coins equals the amount of blocks generated due to the incentive of generating a block being set to 1. This is much lower than bitcoin's incentive which started on 50, and halves every now and then, approximately every 4th year. However, bitcoin adds a new block to their network every close to every 10th minute, while the SpaceMint protocol suggests generating one block every minute. This leads to 60 blocks generated per hour, which is the regular confirmation time of bitcoin(6 blocks). With the Δ set to 50, this can be compared to the confirmation time of SpaceMint as well.

Back to the blocks generated, 9802, for the first week. Calculating the potential of a week, $max = 7 * 24 * 60 = 10080$. These numbers aren't too far apart, but seeing as 100% of the storage available for the system partook in the experiment, and the system should generate 1 block every minute, they should be equal. After 2 weeks of running the blocks count for 19257, while theoretically should be 20160. This can of course be all due to technical discrepancies, the computers weren't top notch.

Looking at the numbers in correlation with how much space every node made available, they seem to look promising. After the first week Miner 1 had 3.3 times the amount of blocks generated than Miner 2. After 2 weeks this number had decreased to 3.2. This seems to work according to plan, seeing as Miner 1 had over a 500 block head start when Miner 2 could begin. Miner 1 has 3 times the space of Miner 2, and should in theory have 3 times the probability of finding a block. The reason for the big head start is due to the long time to set up the space. In the SpaceMint article section 7 figure a, 250 GB was recorded to take somewhere around 500 minutes[12], 750 GB on the other hand takes over 12 hours.

5.1.1 Disconnection

A question that arises is, what happens if a client disconnects? The space is allready added to the block chain, and is counted to the total. There is no transaction that de-commits space, and if you're disconnected there is a possibility that you're sitting on the best quality proof but unable to add it to the chain. The SpaceMint creators seem to have thought of this by adding the discount factor δ. This decreases when more and more space is added to the chain, and if it remains constant it will rise slowly towards 1. It would seem that this will fix the problem, as for a while, lower quality blocks will be added, but when the discount slowly crawls its way towards 1, the quality will also grow.

5.2 SpaceMint as a sustainable cryptocurrency

In July 2017, the reccorded number of altcoins were over 900 [21], so a big question comes to mind. When there are so many coins trying to make a name for themselves, why should SpaceMint be any different. One could argue that it is more environmental friendly, and so therefore people will be incentivized to use it. But from experience, people want what ever can make them money. So far, the only two currencies that have come far enough to be established are bitcoin and ethereum. The most wasteful ones, at that. Is there really a future for SpaceMint in the world?

Chapter 6
Conclusion

All in all, the proof of concept seemed to be a success. The nodes mined according to the space they were offered, and except for a minor variance in block generation, the target of 1 block per minute did not seem too far off. However, the proof of concept is only a basic proof, and will not work on a full scale network with sever thousand nodes. This is due to the TCP connection solution i used. A node cannot be connected to several thousand members over TCP at once.

When it comes to the future of SpaceMint, i doubt it has a future. There is already a cryptocurrency based on proof of space called burst coin[22]. This coin has over 100 users already, which is a huge advantage. This is due to the fact that the value of a coin lies in its network. The point of being a cryptocurrency based on PoS is already gone. However, Burstcoin is vulnerable to time/memory trade off attacks, which is by adding computational power you can generate coins faster, which removes the environmental gain. SpaceCoin as proposed is immune to this, so they sit with a solution. In my conclusion it would be better to attempt to fix the current Burstcoin with an already established network, than to build a completely new one.

6.1 Future work

For this thesis, possible future work would be to try the SpaceCoin-system with more than 2 nodes to see how it would be affected, unfortunately i only had 2 computers to run on. Also, it would be advantageous to try a different approach to the communication between the nodes. Decentralized cryptocurrencies normally run on a peer to peer basis, which uses UDP and not TCP. I did not see it relevant to my small scale trial, but on a larger scale roll-out it should be developed.

References

[1] David Chaum. Blind signatures for untraceable payments. 1982.

[2] Steve Shanafelt. Miners flee ghash.io after near 51%*scenario*, 2014.

[3] Jordan Tuwiner. Bitcoin mining hardware, 2017.

[4] anonymous. Non-specialized hardware comparison, 2015.

[5] bitcoinwisdom.com. Bitcoin hasrate vs difficulty, 2017.

[6] anonymous. Block, 2016.

[7] anonymous. Current bitcoin block number, 2017.

[8] anonymous. Transaction fees, 2017.

[9] anonymous. Controled supply, 2017.

[10] bitinfocharts.com. Cryptocurrency statistics, 2017.

[11] anonymous. Directed acyclic graph, 2017.

[12] Joël Alwen Geog Fuchsbauer Peter Gazi Krzysztof Pietrzak Sunoo Park, Albert Kwon. Spacemint: A cryptocurrency based on proofs of space. 2017.

[13] BlockGeeks. What is ethereum? a step-by-step beginners guide, 2017.

[14] kraken.com. Trade ethereum, 2017.

[15] Arjun Kharpal. Bitcoin may have more than doubled this year, but rival ethereum is up 2,000 percent. here's why, 2017.

[16] bitcoin.org. bitcoin, 2017.

[17] Jason Kinsaid. Google's go: A new programming language that's python meets c++, 2009.

[18] Albert Kwon. spacecoin, 2016.

[19] go python. Gopy, 2017.

[20] anonyumous. Python extension programming with c, 2017.

[21] anonymous. List of cryptocurrencies, 2017.

[22] WoltLab Suite. Burstcoin, 2017.