CES Seminar WS 13/14

Bitcoin
How it Works and What's Behind it

Lukas Bischoff

November 25, 2013

Contents

1 Introduction **3**

2 Cryptographic Basics **4**

2.1 Public-Key Signatures 4

2.2 Cryptographic Hash Functions 4

3 Sending and Receiving Money **6**

3.1 The Structure of Bitcoin 6

3.2 Transactions 6

3.3 Example 7

3.4 Special Claiming Conditions 8

3.5 Anonymity 10

4 Double Spending **10**

4.1 The Block Chain 10

4.2 Proof of Work 11

4.3 Double Spend Attacks 12

5 Bitcoin Mining and Mining Pools **13**

5.1 Bitcoin Mining 13

5.2 High Performance Mining Examples 13

5.3 Mining Pools 14

6 Conclusion and Outlook **15**

7 References **18**

1 Introduction

This paper was written in the course of the CES Seminar at RWTH Aachen University in winter 2013/14. It takes a closer look at the digital cryptocurrency Bitcoin explaining possible attacks and how they are prevented by pure mathematics.

Bitcoin is the implementation of a digital currency functioning without any kind of central authority. The underlying principles were first introduced in 2008 under the pseudonym "Satoshi Nakamoto".[1] The real identity of the person behind Bitcoin remains unknown to this day. Just like any other currency Bitcoin does not represent anything in the physical world. It is merely a list handling accounts and numbers that have a value, because people are willing to trade goods for a higher number next to their account.

Thinking about desirable features of money one can come up with a list of six key properties.

- universally recognized value
- available to everybody
- easy to transfer
- anonymous (while this might not be desired by governments, many individuals would certainly deem it desirable)
- light and portable
- hard to copy or forge

At first sight established currencies like the Euro or US Dollar seem to comply with all six points, but on closer inspection they are neither completely anonymous (different serial number on every bill) nor light and portable in all situations (1kg of US Dollar bills can represent a maximum of about 0.27 million Dollar which is certainly not bad for an individual but does become cumbersome in large business transactions). Lastly copying and forging is hard to the effect that it is difficult to do it in large quantities without getting caught since it is a crime taken very seriously, but the forging itself is perfectly possible. Bank wire transfers and credit cards perform a bit better, but they are perfectly traceable since they work through a central instance and they are not available to everybody (8% of households do not have a bank account in the US).

In the case of Bitcoin it can be said that while it certainly is light, portable and available to everybody, it cannot (yet) be considered to have universally recognized value. There are only very few stores and exchange office around the world accepting Bitcoin. Only time can tell whether this will change or not. As for the remaining three points it will be a part of this paper to explain as to how far Bitcoin complies with them.

2 Cryptographic Basics

Before we go any further, there are two basic cryptographic functions that need to be known. As the mathematics behind them is fairly complicated, this paper will only explain what they do rather than how it is done. However, in both cases links to papers explaining the mathematics are provided.

2.1 Public-Key Signatures

Just like real life signatures, digital signatures are a way of proofing ones identity. The algorithm used to do so in Bitcoin is called ECDSA (Elliptic Curve Digital Signature Algorithm) which, like the wider known RSA algorithm, creates a pair of different but connected keys. The first of the two is called "private key", only known to the person who created it and used to sign messages by performing a series of computations. The second, "public key", is known to everybody in the network. It is the only existing key that can reverse the series of computations done with the corresponding private key in the course of signing a message. Given the public key, it is infeasible to compute the corresponding private key. To sign a message one uses the private key to create a signature from the message. The recipient of a message and the corresponding signature can check whether the sending persons public key decodes the signature back into the message and thereby verify, that the author of the message is in fact the "owner" of both the private and public key. Detailed information on the mathematics behind this process are provided in the references.[2]

2.2 Cryptographic Hash Functions

Hash functions take inputs of arbitrary length and return a bit string called "digest". This digest has a fixed size and is highly sensitive to even tiny changes in the input as the following example will demonstrate.

SHA256("")
0x e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
SHA256("The quick brown fox jumps over the lazy dog")
0x d7a8fbb307d7809469ca9abcb0082e4f8d5651e46d3cdb762d02d0bf37c9e592
SHA256("The quick brown fox jumps over the lazy dog.")
0x ef537f25c895bfa782526529a9b63d97aa631564d5d789c2b765448c8635fb6c

Cryptographic hash functions furthermore need to satisfy the following properties.

- it is easy to compute the hash value for any given input (this property applies to all hash functions, not just cryptographic ones)
- it is infeasible to generate an input that has a given hash
- it is infeasible to modify an input without changing the hash
- it is infeasible to find two different inputs with the same hash

To give an idea of what that means in the case of a 256 bit digest like it is used by Bitcoin, here are two examples.

- Every position in the digest can either be 0 or 1 so there are $2^{256} = 1.2 * 10^{77}$ possible digests. For comparison, there are an estimated $7.5 * 10^{18}$ grains of sand on the earth.
- To have a 1% chance of a hash collision, meaning that two different inputs hash to the same digest, you would have to try an average of $4.8 * 10^{37}$ inputs. This is especially noteworthy since this so called birthday attack exploits the mathematics behind the birthday problem that cause the chances to rapidly rise with the number of tries.[3]

These examples of cause only hold true for perfect hash functions which the one utilized by Bitcoin, SHA256, most likely is not. However, no collision has been found so far and a single collision would in fact not even be a serious thread to the safety of Bitcoin. For that there would have to be a fast method to find collisions for a considerable set of inputs. Detailed information on the mathematics behind hashing may again be found in the references.[4][5]

3 Sending and Receiving Money

3.1 The Structure of Bitcoin

The Bitcoin network consists of millions of computers around the world without any central authority. Each of these computers has its own copy of a digital list called "ledger" that lists all transactions of Bitcoins (BTC) ever made (downloading and checking the ledger for the first time can take up to 24h but it only needs to be done once). In contrast to other digital payment services (eg Paypal) that keep track of account balances, too, no other data is saved in Bitcoin. For a fictive person Alice to be able to send 50 BTC to Bob she must reference one or several past transactions where she herself received at least 50 BTC. These referenced transactions are called inputs to a transaction. If she cannot reference the exact amount of 50 BTC, her transaction will result in 2 transactions (outputs), one of 50 BTC going to Bob, the other one with the remainder back to herself. This is due to the fact that every transaction redeemed as input to another transaction will be marked as "spend" in the ledger. Since there is no way of marking a transaction partly spend, it has to be used up completely.

3.2 Transactions

Let's take the previous example of Alice wanting to send money to Bob and look a bit more closely at what happens. She will broadcast a message to the network, stating her input(s) and output(s). This message is received by other nodes who update their ledger accordingly by adding the outputs as new entries and marking the used inputs as spend. Afterwards they pass the message along till it has spread through the entire network. To keep people from utilizing money they do not actually own, a signature has to be included with every input, to prove that one is the actual owner of that money. Before we can understand this signature we have to recall the public key signatures mentioned before and know, that the addresses money is transferred between in Bitcoin are the sending and receiving persons public keys. With every input Alice is utilizing she includes a signature, calculated as a function of the input itself and the private key, only known to her, that corresponds to the public key the money in the input was received under.

signature = f(input, private key)

Every node in the network receiving her transaction can then check the legitimacy of

all used inputs by using another function depending on the input, the signature and the aforementioned public key.

$$\textbf{true} \stackrel{?}{=} \textbf{f(input, signature, public key)}$$

Since the signature to an input is dependent on the input itself, it can neither be modified while passing the transaction along, nor can the signature be reused for another transaction. This way of owning money also means, that every day funds are lost forever as people lose private keys due to hard drive crashes and other malfunctions.

Recalling our list of six desirable features, "easy to transfer" can now be considered complied with. The only thing needed for Bitcoin transfers is a computer with internet access, the Bitcoin software and the receiving persons address/public key.

3.3 Example

The following picture shows a real example of how a transaction looks like.

Transaction

Short link: http://blockexplorer.com/t/2uR2474Kdt
Hash[?]: 323c3a7b9ee2db1d358d256a869996fdf9ae83d83fd5ed63983385cad75d0cf6
Appeared in block 225530 (2013-03-12 19:02:43)
Number of inputs[?]: 3 (Jump to inputs)
Total BTC in[?]: 2.55669839
Number of outputs: 2 (Jump to outputs)
Total BTC out[?]: 2.55619839
Size[?]: 620 bytes
Fee[?]: 0.0005
Raw transaction[?]

Inputs[?]

Previous output (index)[?]	Amount[?]	From address[?]	Type[?]	ScriptSig[?]
c16fed5ed784...:34	1.00044755	13cJazg9tcSp3RDR4e5H9ZSVqZFJSykhLG	Address	3045022100b105f528bb32dbae8e0be2f4d58d4 04239bbce9ba6d15e2de0306351d943bfe4906d
9410833d9bd1...:49	1.00125084	13cJazg9tcSp3RDR4e5H9ZSVqZFJSykhLG	Address	3046022100846b9a434c3839a205d5eb53c931 04239bbce9ba6d15e2de0306351d943bfe4906d
58b5da277d96...:0	0.555	1MGZ1gQKbAhphSpxTDsvVU8zAdbNhSvhwM	Address	3046022100b345aa27b697e81781e3c6d0d378 044c9edc7e6e3a7bb39d7d5acc909d19c7f25fe2

Outputs[?]

Index[?]	Redeemed at input[?]	Amount[?]	To address[?]	Type[?]	ScriptPubKey[?]
0	Not yet redeemed	0.00019839	1NqLFwuWW1H2Rcx7beb3NmWfEwXHR4v86M	Address	OP_DUP OP_HASH160 ef7eaee28a86815fd02e62ee991190d78d45b4cf OP_EQUALVERIFY OP_CHECKSIG
1	bab3237fd502...	2.556	1H4YcFhqTweqnTYFMzbuAibevX3hwVinn6	Address	OP_DUP OP_HASH160 b02da532aede99344513edf86386ec59afacfbc1 OP_EQUALVERIFY OP_CHECKSIG

Figure 1: Transaction Example [6]

To make it a bit easier to digest, let us put some names behind it. Alice wants to send 2.556 BTC to Bob. To do so she uses three previous transactions to which she owns the private keys. Since these three input transactions add up to 2.55619839 BTC and the fee of this transaction is 0.0005 BTC, the remainder of 0.00019839 BTC is transferred back to Alice. This might actually not be the case and there is no way of knowing due to the aforementioned anonymity but it is very likely that output 0 is in fact back to Alice. The 0.0005 BTC fee will be explained later on in this paper. The other values are:

- **"Previous output (index)"** is the hash of the transaction in which this input was created as output, as well as the index of the output in the output list of that transaction.
- **"From address"** is the public key Alice was using to receive the money. One can notice that the first two inputs were received using the same public key while it was different for the third.
- **"ScriptSig"** is the signature proving that Alice is the owner of the private key corresponding to the public key and therefore allowed to redeem this money. Even though the "From address" is the same for the first two inputs, the signature differs since it also depends on the "Amount" and the "Previous output (index)".
- **"Index"** is the index subsequently quoted by Bob in "Previous output (index)" when he wants to spend this money.
- **"Redeemed at input"** shows whether this output was already used by Bob as an input to another transaction and if so, the hash of that transaction.
- **"To address"** is the address (public key) the money is send to. It can be assumed, that the "To address" of output 0 is Alices and the one of output 1 is the one Bob told her to send his money to.
- **"ScriptPubKey"** is explained in the following chapter.

3.4 Special Claiming Conditions

So far we only looked at money being send from one person to another who can then spend that money using the connected private key. It is however possible to construct situations, where money is sent to a person but to use this money, the person needs

a password on top of the private key. There are even ways to construct escrow based transactions where e.g. 2 out of 3 signatures are required. An example where this is a desired feature is a broker managing money for a married couple. The broker is not able to invest the money without the consent of either the husband or the wife but not both are needed. To understand how this is realized we have to take a closer look at the ScriptPubKey and what it does. It contains operations (everything starting with "OP" in figure 1 and the following table) and constants. If funds are utilized, the network proves their legitimacy by running through the ScriptSig and the ScriptPubKey connected to these funds, successively copying and applying the constants and commands to a stack. Taking our example of money being send from one person to another with no further strings attached this would look like the following (operations in blue, constants in orange):

Stack	Script	Description
empty	<sig> <pubKey> OP_CHECKSIG	stack is empty, ScriptSig and ScriptPubKey are lined up to be processed
<sig>	<pubKey> OP_CHECKSIG	<sig> is copied to the stack
<sig> <pubKey>	OP_CHECKSIG	<pubKey> is copied to the stack
true/false	empty	OP_CHECKSIG is executed, signature is checked

The procedure in this table is called pay-to-pubkey and could be used to do save transfers, however the commonly used script as it can be seen in figure 1 looks a bit different. It is called pay-to-pubkey-hash and a link showing the corresponding table can be found in the references.[7]

A transaction requiring a password to claim the money could be realized with the following ScriptPubKey:

OP_HASH160 6fe28c0ab6f1b372c1a6a246ae63f7 OP_EQUAL

The password given in ScriptSig of the input, utilizing these funds, would have to hash to **6fe28c0ab6f1b372c1a6a246ae63f7**. If no ScriptPubKey is specified, the money can be claimed by anybody using the ScriptSig **OP_TRUE**. The very first transaction in the Bitcoin network was in fact one, that could be claimed by anybody.

3.5 Anonymity

Using the TOR network to access the Bitcoin network, it is possible to hide ones IP address, revealing only the public key used during the money transfer. This key can be changed for every transaction, circumventing someone from linking several incoming transactions to the same person. They will however be linkable at the moment they are grouped to make a payment as it has to be proven that one is the rightful owner of all the money being utilized.

Looking at the possibility of creating a set of keys already in use by somebody else it can be said that this is extremely unlikely as the number of possible keys is 2^{160}. The Bitcoin software does not even consider that case even though it would enable the involved to spend each other's money.

This completes our list further as it shows that Bitcoin can be used completely anonymous.

4 Double Spending

The big problem remaining is the time it takes a message to run trough the network to reach all nodes. No node can be sure that the order it receives the messages in, is the order they were created. A malicious user Alice could therefore send money to Bob who would in return ship a product for it. Alice would then send another transaction, initializing the same funds she previously send to Bob, to herself. Since certain parts of the network did not yet receive the transaction to Bob they would count Alices transfer to herself valid rather than the one to Bob. This would lead to uncertainty within the network about which transaction came first and should be considered valid for future transactions.

4.1 The Block Chain

To avoid this problem and make the whole network agree on one version of the transfer history, Bitcoin groups transaction in blocks that are linked together to a block chain. All transactions placed in one block are considered to have happened at the same time. Every block references the hash value of the previous one to form the chain. Transaction requests broadcasted to the network but not yet present in the block chain are called "unconfirmed". Every user collects a set of unconfirmed transactions and forms a suggestion for what the next block in the block chain should be. He then suggests it to the

network but once again this alone would lead to uncertainty within the network about which block is next. He therefore has to suggest not only a block but also a value called "proof of work".

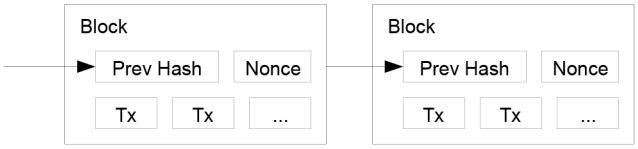

Figure 2: Visualization of the Block Chain, Tx are Included Transactions, Nonce is the Proof of Work

4.2 Proof of Work

The proof of work is a so called nonce, r, that is concatenate with the suggested block and then put through a cryptographic hash function. If the digest is smaller than a set target value it is considered to be a valid proof of work and the block accepted to be next in the block chain. Due to the aforementioned features of hash functions, nodes have no other option but to try out random values for r. Just like playing the lottery, it is possible to guess right the first time but on average it takes a certain amount of time. As of today the target is:

00000000000000AFC850000000000000000000000000000000000000000000

Since this is a hex value, the digest is required to start with 60 consecutive zeros. The chances of finding a corresponding hash in one attempt are:

0.000000000000000000595574176941242691045788615333833604 3

Current GPUs are able to achieve hash rates of around 500 MH/s (MegaHashes/Second) which results in about 73 years to find r. The whole Bitcoin community is however able to average 10 minutes to bring a new proof of work. As the network grows and faster hardware is released, the target is automatically retuned by the Bitcoin software every two weeks to keep the average guessing time constant at 10 minutes. The choice of 10 minutes was somewhat arbitrary, but very short times would lead to instability at the end of the chain, while longer ones would cause confirmation times to go up. The Bitcoin alternative Litecoin targets block times of 2.5 minutes.

Other nodes receiving a block suggestion can quickly check the validity of its proof

of work. If it turns out to be ok, the node will stop its work and start building on the new, longer chain. To do so, it has to adjust its suggestion for a new block to contain the reference to the new last block in the chain. It also has to check that the block does not contain any transaction that was already included in the received block.

The hashing function causes the solution finding process to spread out in time, so a found block is broadcasted and accepted throughout the whole network before a second, different, but also valid block is found by somebody else. Occasionally however this can happen. In that case the block chain will split into several branches. Each node would build on the first valid block it received. Since every node immediately switches to the longest branch it receives, the network would come to an agreement on how the block chain looks like when the next block is solved. While it is rare for two blocks to be solved at the same time, it is virtually impossible for this to happen several times in a row. The transactions in the shorter branch, if not present in the longer, will get tossed back into the pool of unconfirmed transactions.

4.3 Double Spend Attacks

This tossing back of transactions from shorter branches potentially opens the door for fraud. Using our example of Alice sending money to Bob again, Bob would wait for the transaction to be taken into the block chain and then ship the package to Alice. If she could then come up with a longer chain in which she replaced the transaction to Bob by one to herself this would be accepted by the network, since all nodes always switch to the longest chain available. The transaction to Bob would get tossed back into the pool of unconfirmed transactions and by the time it is included in a new block, it is considered worthless, since the funds it is trying to utilize have been spent already.

This is impeded by the fact, that Alice would have to outperform the whole network in finding a longer branch. When Bob ships the product she wants to present a chain that is longer than the one the rest of the network was able to find in the meantime. Considering the earlier mentioned 50 years it takes a personal computer to solve a block, this is extremely unlikely. It is also impossible to just precompute a branch and use it at the right moment to replace a shorter one just as it is impossible to replace a block in the middle of the existing chain. This comes as a consequence of the fact, that every block contains a reference to the hash value of its predecessor. At any point Alice can only start building a chain using the current state of the chain. That way she would always be in a race with the network. Changing a transaction in an existing block would result in a completely different hash value for that block. The following blocks reference

would therefore not fit any longer. To successfully replace a block Alice would have to find an r that in combination with the new block would result in the exact same digest of the replaced block. This would be called a hash collision and has never been done for SHA256, the hash function Bitcoin uses.

5 Bitcoin Mining and Mining Pools

5.1 Bitcoin Mining

The motivation for people behind providing computing power to calculate new blocks is a so called block reward. For every block contributed to the chain, the creator of that block gets a reward of 50.0 BTC. It is included in the blocks outputs in the form of a "generate transaction" which in contrast to regular transactions does not need any inputs. If there is no more than one generate transaction with the right amount of Bitcoins being generated, the block is accepted by the rest of the network. This reward is cut in half every four years until it ceases approximately 2140, leaving the Bitcoin network with a total of 21 mio Bitcoins. Once this point is reached, Bitcoin will become a deflationary currency due to the earlier mentioned irreversible losing of money. The process of trying to find the next block is called Bitcoin mining. To keep Bitcoin mining attractive even past the point of block rewards, miners also get any transaction fees, optionally included by the sender of the money. At the moment even transactions with no reward are included in the block chain since mining is done mostly for the block reward but in the future this will most likely change. Transactions with high rewards will be included right away while those without will be ignored. The hope is that these fees will still be lower than credit card or intercontinental wire transfer fees.

5.2 High Performance Mining Examples

The computing cluster of the Forschungszentrum Jülich which in November 2013 ranks 8 in the top500 list of fastest computers in the world reaches around 5000 TFlop/s, consuming 2300kW (cooling included).[8] Assuming the current average 30 days exchange rate of 120 $/BTC, energy costs of 0.34 cent/kWh (Germany) and the current network hashrate of 40793 PFlop/s, this would result in a yearly balance of:

$$(\frac{5000*10^{12}}{40793*10^{15}}*52560\frac{blocks}{year}*50\frac{BTC}{block}*120\frac{\$}{BTC})-(2300kW*8760\frac{hours}{year}*0.34\frac{\$}{kWh}) = \textbf{-6811666}\frac{\$}{year}$$

This can be explained by the fact, that CPUs are very inefficient for hashing. While consumer CPUs only average 10 MH/s (recalling the 500 MH/s for consumer GPUs), there are special hashing rigs ranging from 6$ to 30000$, reaching 6 MH/s to 1500 GH/s. Assuming the same exchange rate, energy price and the current network hashrate of 3212 TH/s, the Black Arrow Prospero X-3 ASIC mining rig (1344 GH/s @ 0.75kW, 3999$) would yield a yearly profit of:

$$(\frac{1344*10^9}{3212*10^{12}} * 52560 \frac{blocks}{year} * 50 \frac{BTC}{block} * 120 \frac{\$}{BTC}) - (0.75kW * 8760 \frac{hours}{year} * 0.34 \frac{\$}{kWh}) = \mathbf{218342} \frac{\$}{year}$$

This difference is explainable by the fact, that SHA256 uses a huge amount of relatively simple calculations with little memory usage, clearly favouring GPU, FPGA and ASIC miners over CPU miners. Bitcoin alternatives like Litecoin are therefore using a different hash function intended to level the playing field but this assumption turned out to be wrong as GPU miners are once again faster.

5.3 Mining Pools

To receive a steadier income instead of waiting several years till getting lucky and being the first one to solve a block, people form groups to work together on finding the next block (similar to lottery pools). These groups are called mining pools and their income is split up between the members depending on how much computing power they contributed.

Some of these pools are quite big with two of them making up more than 20% of the the whole network each. The biggest one, BTC Guild, was on several occasions able to produce 6 blocks in a row and therefore voluntarily limited its members to not cause mistrust within the rest of the community.

Since a double spending attack gets harder the deeper a transaction is in the block chain, it is recommended to wait till the transaction is in the second to last block of the chain (for big transactions the sixth block to the end).

Given these numbers we can consider Bitcoin to be mathematically save against copying or forging and therefore comply with the last point on the list.

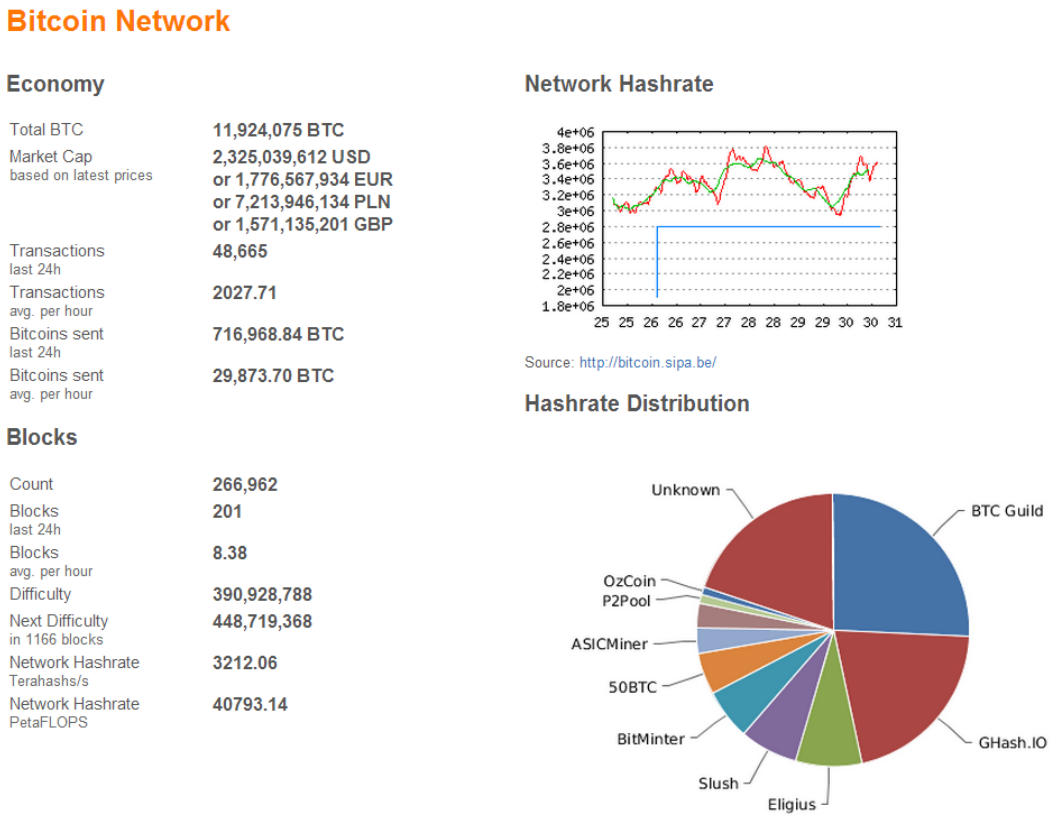

Figure 3: Hashrate Distribution and Other Numbers of the Bitcoin Network [9]

6 Conclusion and Outlook

All in all it can be said that Bitcoin is a currency combining more positive features in it than any other form of money available right now. However, partly due to these features, Bitcoin is facing substantial problems. Coming as a consequence of the anonymity, Bitcoin is highly connected to illegal activities reaching all the way from online black markets like Silk Road, sometimes referred to as "eBay for drugs", to assassination markets where anybody can contribute to bounties on government officials heads. It furthermore enables money laundering as well as tax evasion. Another big problem coming as a consequence of low to none existing transaction fees is the big number of speculators using it. It is assumed that as of 2013, around 70% of Bitcoin trading volume is in fact due to speculation. Lacking any kind of intrinsic value, Bitcoin has been described by Professor John Quiggin of the University of Queensland as "perhaps the finest example of a pure bubble".[10] Erratic exchange rate fluctuations as they have

been happening in spring and autumn 2013, causing jumps between 10$ and 900$ with daily jumps of as much as 300%, give rise to doubts whether Bitcoin can be a functioning medium of exchange. Its design, resulting in deflation, also encourages hoarding which furthermore hinders its usability.

Figure 4: Bitcoin Exchange Rate and Trading Volume in 2013 [11]

Bitcoin has furthermore been criticised for wasting real resources on the process of mining. Bloomberg argues, that lots of the advantages Bitcoin enjoys, are due to the fact that governments did not pay too much attention so far, but as that changes gradually, these advantages will nullify. Thailand as the first country in the world banned any kind of trading done with Bitcoin and other countries could soon follow. If not banning it completely, there will for sure be laws instated that monitor and tax Bitcoin flows when they are cashed in for "real" currency. [12][13][14]

Despite these disadvantages it has been observed that Bitcoin has become increasingly popular in countries with problem-plagued national currencies. Examples for this are the Argentine peso that is stymied by inflation and strict capital controls as well as the Iranian rial against which currency sanctions were imposed. There is even a suggested linkage between increased Bitcoin usage in Spain and the Cypriot financial crisis. There is also a growing number of vendors accepting Bitcoin (Mega, Baidu) and shops that price their goods soley in Bitcoin (bitcoinin, BitcoinShop.US). Lastly even ATMs supporting the access to Bitcoin wallets are starting to appear.

A lot of Bitcoins fade will be decided over the coming 10 years. The single most important thing will be how the number of shops accepting Bitcoin as a way of payment will develop. With the ongoing NSA scandal and international internet shopping on the raise there is certainly a growing number of people looking for methods of payment that cannot be monitored by governments and provide low transaction fees, so chances for

Bitcoin are not bad. In the long term it will be equally important how transaction fees develop as the block reward ceases and governments might invoke transaction fees on the conversion of Bitcoin to “real world“ currencies.[15][16]

Taking into account how much happened in the first five years of the existence of Bitcoin, we have an interesting ride ahead of us.

7 References

[1] Satoshi Nakamotos original paper on Bitcoin
http://bitcoin.org/bitcoin.pdf

[2] private blog entry explaining the math behind the ECDSA algorithm
http://kakaroto.homelinux.net/2012/01/how-the-ecdsa-algorithm-works/

[3] english Wikipedia on the birthday attack
http://en.wikipedia.org/wiki/Birthday_attack

[4] paper on the functioning of SHA cryptographic hashes
http://www.iwar.org.uk/comsec/resources/cipher/sha256-384-512.pdf

[5] english wikipedia on cryptographic hash functions
http://en.wikipedia.org/wiki/Cryptographic_hash_function

[6] official Bitcoin website publishing the block chain
http://blockexplorer.com/t/2uR2474Kdt

[7] official Bitcoin wiki displaying the pay-to-pubkey-hash script
https://en.bitcoin.it/wiki/Script

[8] top500 list of fastest computers in the world
http://www.top500.org/lists/2013/06/

[9] official statistics and numbers about the Bitcoin network
http://bitcoincharts.com/bitcoin/

[10] article on Bitcoin by Professor John Quiggin
http://nationalinterest.org/commentary/the-bitcoin-bubble-bad-hypothesis-8353

[11] Bitcoin exchange rate
http://bitcoincharts.com/charts/mtgoxUSD#rg360ztgSzm1g10zm2g25zv

[12] Bloomberg article on the future of Bitcoin
http://www.bloomberg.com/news/2013-08-08/did-the-sec-just-validate-bitcoin-no-.html

[13] Bloomberg article on the future of Bitcoin
http://www.bloomberg.com/news/2013-11-20/bitcoin-is-still-doomed.html

[14] article on Bitcoin being banned in Thailand
http://www.telegraph.co.uk/finance/currency/10210022/Bitcoins-banned-in-Thailand.html

[15] private article on the future of Bitcoin
http://simondlr.com/post/44214403928/speculation-about-bitcoins-future

[16] article on Bitcoin ATMs

http://www.cbc.ca/news/technology/world-s-first-bitcoin-atm-opens-in-vancouver-1.2286877

[17] english Wikipedia on Bitcoin

http://en.wikipedia.org/wiki/Bitcoin

[18] private blog entry explaining the functioning of Bitcoin

http://www.imponderablethings.com/2013/07/how-bitcoin-works-under-hood.html

[19] online acadamy course on Bitcoin

https://www.khanacademy.org/economics-finance-domain/core-finance/money-and-banking/bitcoin/v/bitcoin-what-is-it

[20] online acadamy course on Bitcoin

https://www.udacity.com/course/cs387

[21] official Bitcoin wiki

https://en.bitcoin.it/wiki/Main_Page