Universitá di Torino

Scuola di Scienze della Natura
Dipartimento di Fisica
Corso di Laurea Magistrale in Fisica dei Sistemi Complessi

Emerging cryptocurrency trust in an agent–based model

Relatore:
Prof. Pietro Terna

Controrelatore:
Prof. Marco Maggiora

Presentata da:
Luigi Battistoni

Anno Accademico 2016/2017

Abstract

This thesis aims to understand under which conditions the diffusion of a cryptocurrency in a network using fiat is possible. A detailed explanation of the functioning of the blockchain and the Bitcoin protocol is provided. What is more real problems regarding the digital money diffusion are shown. Model dynamic and parameter space are investigated using genetic algorithms and the results are analyzed focusing on clustering properties. The resulting data represent a set of critical points near which the system dynamic may drastically change. Simulations are done both in a single layer model and in a multi layer one. Interpretation of the results finally shows the parameters value in order to observe the emergence of the trust in the cryptocurrency.

"Not all those who wander are lost."
J.R.R. Tolkien

Contents

Introduction

0.1 Bitcoin

Bitcoin is a digital cryptocurrency invented by *Satoshi Nakamoto* in 2008, based on *blockchain* and *Proof-of-Work* technologies. Its main features, like the decentralization of the protocol, the anonymity of users and the public log of transactions, granted Bitcoin a quite controversial spot. On one hand the functioning of Bitcoin protocol is difficult to understand especially for a non expert in blockchain technologies or cryptography. What is more Bitcoin first years are associated to a safe way to conclude illicit transactions. On the other hand the features of the digital money themselves imply both the possibility of financial speculations and the ability to use a safer and less expensive money system.

0.2 Bitcoin value history

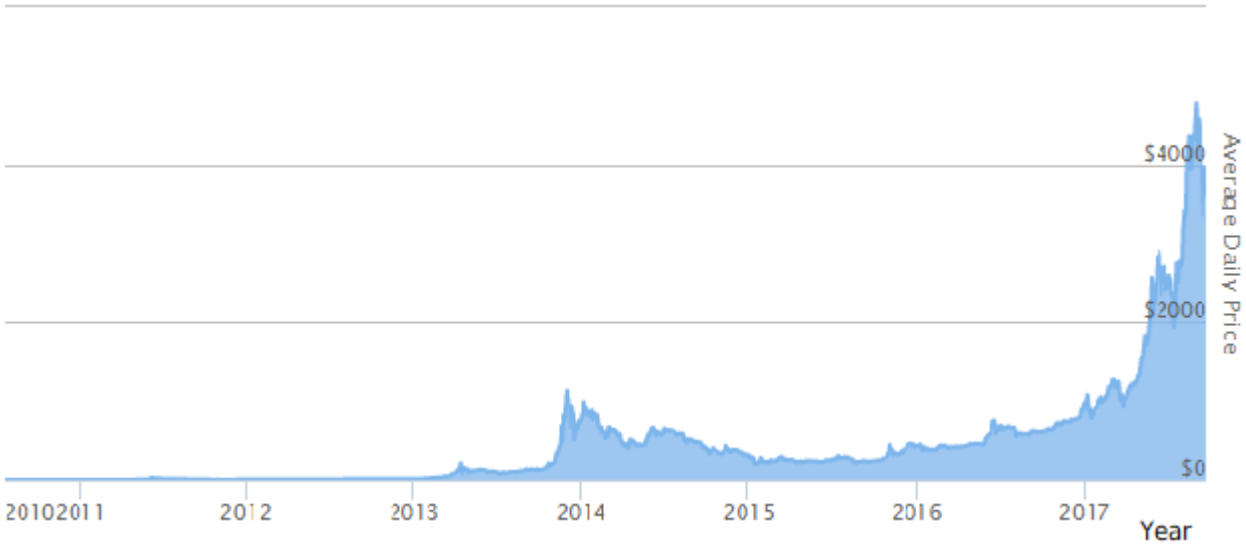

Figure 1: Bitcoin price chart. Source: *99bitcoins.com* [2] .

Date	Event	Value ($)

January 3, 2009	The first Bitcoin transaction block is created. The initial coin offering is not done yet. Value is $0.	0
October 5, 2009	New Liberty Standard allows to exchange Bitcoin. The exchange rate is fixed to 1,309.03 BTC to one US dollar. This is the electricity cost to mine the cryptocurrency.	0
October 12, 2009	New Liberty Standard concludes the first digital to fiat transaction.	0
July 18, 2010	The full-time Bitcoin exchange {Mt.Gox} opens.	0.07
August 15, 2010	An hard fork due to protocol bug occurs, resulting in a fraudulent transaction. Even if this bug was rapidly fixed the market trust in Bitcoin wasn't untouched.	0.07
February 9, 2011	Bitcoin price reaches $1.00 USD. The news spreads and popularity of Bitcoin starts to grow.	0.96
June 1, 2011	Gawker publishes an article about The Silk Road. This makes the interest in Bitcoin grow and people start to understand the potential of this currency.	9.21
June 19, 2011	Mt.Gox got hacked and fraudulent transactions are made. What is more users of MyBitcoin (a wallet storage) got their Bitcoin stolen, resulting in a theft of over 4000 BTC. This results in a bad period for Bitcoin, that results in a fast loss of value.	17.77
March 1, 2012	The hacking of Linode results in the loss of 46000 BTC by the wallet of, in particular, a large pool of miners and a wallet storage service. Losses of both the wallets are beared on behalf of their costumers and users.	4.92
November 15, 2012	Wordpress starts to accept Bitcoin, simplifying the way of payment for Haiti, Etiopia and more countries where PayPal access is blocked. This results in a spread of the Bitcoin and in a grow of its value.	11.04

November 28, 2012	The first Halving Day. The number of Bitcoin mined from a single transaction goes from 50 to 25. This occurred when the 210000th block was solved and until the block 420000 the reward will be fixed at 25 BTC. At the moment before the halving 10,500,000 BTC had been mined.	12.25
March 25, 2013	Cyprus is funded by Eurogroup, the European Commission, the European Central Bank and the International Monetary Fund with 10 billion Euros. As a condition for the bailout, most bank accounts would be forced to pay a sizable levy. In order to preserve their holdings (as Cyprus favorable polices had made it a tax heaven) customers buy a large amount of Bitcoin. This results in a growth in its value.	74.02
April 10, 2013	Customers solution to Cyprus bailout results in an overload in Mt.Gox server traffic, and in a consequent fail of transactions execution. This results in a panic sell that saturates the market and lets the Bitcoin price fall.	181.66
October 1, 2013	FBI identifies the person behind the dark web marketplace (The Silk Road). Ross Ulbricht is arrested and charged with narcotics traffiking, computer hacking and money laundering. In this operation about 170,000 BTC are seized by the authorities.	133.03
November 18, 2013	US Senators state that Bitcoin is promising. The general consensus continues to rise, followed by money value. A month has passed since Ulbricht arrest and Bitcoin passed from 133.03 USD to 685.75 USD.	685.75
November 20, 2013	People Bank of China announces that people are free to participate Bitcoin market. This makes the already active Chinese market grow.	641.23

November 29, 2013	The Chinese announcement makes Bitcoin value peak, expecially because it provides a valid alternative to Chinese customers to their inflated currency. The Chinese interest in Bitcoin is credited to Jet Li's One Foundation, that accepted Bitcoin as donations for the April 20th, 2013 Lushan earthquake and collected 230 BTC. The news was covered by the national media and Bitcoin fame grew.	1242
December 5, 2013	People's Bank of China declares that Bitcoin is not a currency. This results in a ban for financial institutions from using Bitcoin. Prices rapidly fall down.	1022.37
February 24, 2014	Mt.Gox closes after an hacker attack, no additional information are provided.	547.09
July 18, 2014	Micheal Dell announces that his company will now accept Bitcoin as a form of payment. However some form of regulation is imposed, for example only US costumers can use it and transactions must be handled by Coinbase.	624.1
December 11, 2014	Following Dell, Microsoft announces that it will accept Bitcoin payments for US costumers, using Bitpay for payment processing.	352.56
September 18, 2015	Bitcoin is declared as a commodity by the US regulator.	234.65
October 22, 2015	The European Court of Justice ruled that digital currency transactions are not subject to VAT in the EU. This classifies Bitcoin as a currency, and not as a good or a property.	273.82
October 31, 2015	The Economist publishes an article focused on the utility of the blockchain. Bitcoin featured are on the front page of the publication.	323.35
April 27, 2016	Steam announces that it will accept Bitcoin as a form of payment, using Bitpay.	461.08

July 9, 2016	Second Halving Day. The creation of block 420000 results in the halving of the mining reward, from 25 BTC to 12.5 BTC.	625.14
January 3, 2017	For the first time in 3 years Bitcoin is worth more than 1000 USD. Media coverage of the news brings new users.	1020.47
April 1, 2017	Japan recognises Bitcoin as a legal tender, supported by usual controls and regulations.	1085.03
August 1, 2017	Bitcoin divides into Bitcoin and Bitcoin Cash. Bitcoin protocol will optimize in order to support small transactions, while Bitcoin Cash will move towards bigger sized blocks.	2787.85
September 2, 2017	Bitcoin value continues to grow. A new high peak is registered at the exchange rate of 4780.15 USD for one BTC.	4780.15

0.3 Trust

Why do we talk about trust? Looking at the history of Bitcoin value it is clear that value increases after *good news*, like the recognition of BTC as a currency or the possibility to pay online with it, and decreases after *bad news*, like the hack of Mt.Gox.

The more the people trust Bitcoin the more they use it, the more they use it the more other companies implement it as a method of payment and consequently its value increases. On the other hand, and this is why a *DisUtility* function will be implemented in the model, Bitcoin still has a significant speculative side. The more it gains value the more media will talk about it and consequently more people will get in touch with it raising its value.

Still a question remains unsolved. Is it realistic to expect a future digital currency based economy? Will any digital money, say now Bitcoin, be used by common people or will it be used just by speculators or technicians? The *Ermakova* [46] tries to answer this question using an online survey directed to experts in Computer Science or Finance.

The pool of the 137 participants was formed as follows. 65.03% of experts were males and 30.77% females. The majority of experts were aged 30-39 (the 36.36%) and 40-49 (27.27%). A smaller proportion was 20-29 (20.98%)

and over 50 (12.59%). The most of the experts were US American (38.46%) and German (39.86%) and professors in their respective fields.

Questions asked were about the future of Bitcoin and its best features. For example participants were asked if they thought that Bitcoin would be as popular as PayPal in ten years, or the biggest challenges to Bitcoin adoption, or important reasons why one should *not* adopt Bitcoin now.

Participants show to appreciate the absence of a central organ controlling Bitcoin and the consequent absence of taxation. These are quite obvious as they are cryptocurrencies main features. What is more being Bitcoin not country based, 80% of experts agreed that a worldwide usage would be realistic and possible.

Again the 80% of experts recognized as moderately or greatly valued the pseudo-anonymity of Bitcoin. It is known that big data analysis can break the Bitcoin protocol anonymity, but in a first approximation we can attribute some kind of pseudo-anonymity to it.

The missing conditions for participate in Bitcoin community is considered by 70% of experts greatly or mildly important for BTC adoption. What is more the absence of a central point of trust may influence Bitcoin adoption. This means that right now Bitcoin protocol suffers from the *attack of the 51%*, that means that a large enough pool could take decisions on transaction blocks acceptance and on double spending. On the other hand the introduction of a central organ would completely twist cryptocurrency nature.

Theft or loss of Bitcoin is not reversible and every user should deal with this fact. This is probably the main problem to overcome in order to have a Bitcoin acceptance. What is more being cryptocurrency new to both economy and law system, there is the lack of clear regulations regarding them. This results in the possibility of exchange Bitcoin in illicit transactions, with consequent loss in trust and gain in volatility. Again the fact that Bitcoin is so volatile, basing its value only on demand and offer, makes it unpredictable and unstable. As a consequence this negatively influences potential new adopters.

To sum up the main obstacles to overcome in order to have a widespread acceptance of Bitcoin are the vulnerability of wallets, the loss of coins due

to the loss of private key, the risk of deflation due to the fixed amount of possible existing coins, the volatility of exchange rates, unclear taxation and regulation system, lack of general trust in Bitcoin system, irreversibility of transactions and the low adoption of Bitcoin by shops and sellers in general.

Last it is important to notice that these barriers are caused by lack of trust. However two levels of trust can be recognized. One is at the consumer level, as one does not want to lose money due to a theft or is expecting to be able to spend money to buy any kind of stuff he or she needs. The second is at speculation level, since one is afraid of volatility or taxation of big amount of money.

0.4 Expectations and results

The digital currency spread is theoretically supported by the so called *Gresham Law*, that states that *bad money drives out good one*. This thesis aims to find using *genetic algorithms* a set of critical conditions for the emergence of trust in a cryptocurrency. Resulting data will be studied using clustering properties. Critical sets analysis will show that the gain in value of cryptocurrency respect to fiat is crucial for the emergence of trust. What is more agents are more likely to adopt digital money if their neighbors do. These two results are theoretically predicted and hence we can say that are in accordance with the hypothesis.

Moreover an unexpected result was found: trust can emerge even if agents are not interested in what their neighbors think about the cryptocurrency. In fact in this particular behavior agents are only interested in the speculative part of digital currency and diffusion is linked only to the gain in value of the cryptocoins.

Multilayer analysis is able to identify one cluster. Interpretation of the results shows that the diffusion of the cryptocurrency is possible under a certain set of parameters and happens following the Gresham Law.

1 Introduction to cryptocurrency and Bitcoin

The emergence of cryptocoins has its roots in the technology innovations. As the world changes and becomes progressively more and more connected the need for a money specifically designed for web transactions increases. Digital money tries to cover this need, providing a decentralized and anonymous

form of payment. Despite the cryptocurrencies potential, they are judged by the average consumer to be some sort of scam or money laundering system. It is quite common when talking about Bitcoin to be asked the question "Aren't Bitcoin illegal?". This is probably caused first by the fact that Bitcoin actually *is* one of the best way to conclude illicit transactions: think about *Silk Road*. What is more in Spring 2017 sensible information in outdated PCs were stolen and the ransom was asked to be paid in Bitcoin. This surely does not help Bitcoin raising its reputation among people. Moreover doubts over cryptocurrency can raise from a general non comprehension of complex mechanisms like the functioning of the blockchain and the mining process. In order to explore the vast world of cryptocurrencies let us begin with a brief summary of digital money and Bitcoin history [27].

The first work introducing the idea of cryptocurrencies was published in 1982 by *David Chaum* [28]. In 1990 *Chaum* founded *Digicash*, a company that coupled fiat money and cryptography in order to grant the anonymity of transactions. *Digicash* was shut down in 1999.

The actual first digital and decentralized money was thought in 1998 by *Wei Dai*. He explained his idea in a crypto-anarchist mailing list. Even if his idea was quite rudimentary it already introduced the ideas of creating money solving a computational problem and a public log of transactions. What is more it contained a complex proto-idea of smart contracts. He called this first digital money *B-money*.

In December 2005 *Nick Szabo* posted in his blog the protocol of *Bit-Gold*, an implementation of *Wei Dai* ideas: the *proof of work*, that is currently how blockchain controls the validity of a new block of transactions.

These two papers led in October 31st 2008 to the conceptualization of the first cryptocoin, the *Bitcoin*, by *Satoshi Nakamoto*. Here is reported the abstract of his paper. A full version of the paper can be found at *http://www.bitcoin.org/bitcoin.pdf* . Details regarding the functioning of technology under Bitcoin will be given later.

Bitcoin: A Peer-to-Peer Electronic Cash System

Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming

> a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

These three papers constitute the base of digital money philosophy: the web is used in order to create a *decentralized* system that autonomously regulates and that does not need the presence of a *trusted third part*. Moreover complete anonymity is granted and all transactions are public. Theoretically *trust is not necessary* in this system, since the system itself corrects errors and avoids frauds by design. However this is not the kind of trust we will talk about in this thesis, in fact in order to adopt a new kind of money one has to be basically able to be confident in the fact that the money value is "real" (in the sense that no bubbles are overestimating it), that it will not be subject to high fluctuations and that people he interacts with accept it.

Starting from 2008 the spreading of cryptocurrencies has been astonishing. This was maybe boosted by the 2007 financial crisis that reduced the trust in fiat currencies.

January 3rd 2009 the first block (*Zero Block*) was added to the blockchain. January 12th 2009 the first Bitcoin transaction was made. The first Bitcoin valuation was made in October 2009 by the *New Liberty Standard* based on the cost of energy necessary to mine a Bitcoin (1 USD = 1,309.03 BTC). February 6th 2010 the first Bitcoin trading platform opened. It was called *Bitcoin market* and allowed to exchange Bitcoin and Dollars. In July 2010 another trading platform opened in Japan (*Mt.Gox*).

In October 2010 the *GAFI-FAFT* delivered the first warning to Bitcoin, meaning that even if at that time its capitalization was less than one million dollar it already had some international attentions. In 2011 *Silk Road*, a site where transactions could only be fulfilled with Bitcoin, was opened. All this sites mentioned are nowadays closed.

The history of Bitcoin from 2010 to 2017 is made of creations of pools of miners, hard forks, security violations and new releases of the application and

of the protocols. This only underlines the flexibility and the polymorphism of cryptocurrencies.

1.1 BlockChain

A blockchain is a public ledger of all transactions that have occurred in a network and it is shared among the participants of the network themselves [30]. Its two main properties are that it is immune to counterfeit and that it is shared among all the nodes of the network. This means that there is no need of a central authority that can guarantee the security of the informations held by the nodes. Let us begin with explaining in details how a blockchain works [22] [42] [25] [41]. As Bitcoin transactions are blockchain-based let assume that in our network the payments are made with Bitcoins.

For example let assume that node A wants to buy a good from node B, at the cost of x Bitcoin. As this currency is not physical, the only way for B to determine whether A has that amount of money or not, is to "ask" it to the blockchain. As we said, being it a *public* ledger of all transactions, B can easily determine if A really has that amount of Bitcoins simply observing all the transactions that resulted in him having those money. To be more clear: let us assume for simplicity that A has exactly x Bitcoins in his wallet. If there are no pending transactions reporting that he is trying to transfer his money to someone different from B, then the exchange with B can be made (as he is not double spending). Now B has to check that A really has that amount of money. In order to do that he can inspect all the transactions that A made. If he received at some time a quantity$x$ from C and he did not spend it, then he really has that amount available in his wallet. But this is not sufficient, in fact B has to determine if C really had that money, so he can check all the exchanges involving C that resulted in him having those x Bitcoins that he traded to A. This reverse process can be done until the source emitting that x Bitcoins is reached. That source is called *miner* and is a member of the network. I will talk more about miners. At the end of this the transaction is made and the whole network is informed.

Even if this example is simplifying too much, it is useful in order to understand how a transaction is made. The major issue of adding the new transaction to the whole blockchain will be treated later. Let us take a first step: how are nodes and their *wallets* (their amount of money) identified?

As we know, in our economy there is the need for a *trusted third part* in order to do transactions. For example, when we use credit card to buy

some goods, the trusted third part is the bank that assures that A has the quantity of money necessary to pay B and that it will be transferred to B deposit. Bitcoin uses *cryptographic* proof instead of a third part. Each node has a *public key* that is the string associated to that node wallet visible to all network (it is like a username) and a *private key* that it is known only by the owner (like a password). I want to underline the fact that each node has a public key, that is nothing but a fictitious name. No *real identity* proof is required in order to join the Bitcoin network. I will refer to this property as *anonymity*. In our example A has to prove the property of the wallet associated at his public key using his private key. If they match the transaction is made and is broadcast to all the nodes. Now we are ready to talk about the *double spending* issue and the *consensus* problem. These can be stated as "How is a block added to the chain and how can be proved the validity of it?"

By double spending issue I mean that someone has to assure that A is not trying to pay B and C with the same Bitcoins at the same time. This is done by the blockchain itself, that guarantees that the transaction using that specific amount of money is the only one pending. The consensus problem is more complex, as it is asking how the blocks are recognised as valid by all the nodes in the network. A simple democratic method is not feasible: a fraudulent node can have the possibility to create many other wallets in order to control a large part of the network and so influence the validation process. This can maybe be avoided in a non-anonymous system, but this is not the case. Bitcoin protocol uses the so called *Proof of Work* (PoW) to solve this main issue: when a node wants to add a block of transactions to the chain it is required to solve a complex *mathematical puzzle*. This kind of puzzles can be solved only using a large computational power and the solution itself is the proof that a certain amount of work has been done. Let us take a step back: assume that the new block is added to the entire chain. Now a cryptographic key will be created and will link the new block to the previous existing one. That previous one contains a key that links it to the previous one and so on. What's more every block contains the PoW, so every block contains the solution of the problem generated for that particular block. So every block contains a large amount of computational work. This means that if someone wants to counterfeit the chain he has to modify all the blocks. The amount of computational power to do this is clearly too big. Moreover the PoW system avoids *forks*. When a transaction is broadcast to the whole network it clearly reaches different nodes at different times. This can lead to different

miners working on different blocks at the same time. If two problems are solved simultaneously then *two* blocks will refer to *one* previous block. In this eventuality the blockchain will have a *fork*. As a result two versions of the chain can exist at the same time. This issue is solved by the protocol deleting the chain containing the less amount of computational work. It is clear how this can prevent fraudulent actions.

To sum up in order to add a block of transactions to the chain some nodes will decide to use their computational power to generate a PoW. The nodes doing so are called *miners* (because they have to spend energy in order to create money, like if they were mining nuggets) and they all compete in order to solve the puzzle. The first capable to find the solution will send it to the whole network and is rewarded with a certain sum of Bitcoin (currently 25 Bitcoins). When the block is created it is *timestamped* and a string is added to it and to it predecessor, in order to identify their order. What it is interesting is that in this block can be added some *metadata* that are never going to be loss or counterfeit by blockchain properties. I will talk more about this.

1.2 Cryptographic hash function

A cryptographic hash function is an algorithm that converts a string of arbitrary dimension into a string of fixed dimension, called *hash* [5]. Such algorithm must have the following properties:

- applying the algorithm to two identical strings must lead to the same result, or in other words the function is *deterministic*;
- finding the *pre-image* of the function must be very difficult. This means that, given an hash, it should be impossible to invert the algorithm in order to find the string that leads to such hash;
- given an input string s it must be difficult to find a string s' such that the hash of s is equal to the hash of s';
- it must be difficult to find two different input strings s and s' that lead to the same hash.

Cryptographic hash functions are related to the addition of blocks to the blockchain. This will be explained in details in the section *proof of work*. The Secure Hash Algorithm used by Bitcoin is *SHA-256* [16].

1.3 Proof of Work

We saw that in order to add a block to the blockchain the miner has to solve a *mathematical puzzle*. The purpose of this puzzle is to guarantee that a certain amount of work has been done by the miner (or more precisely by the miner's calculators). As a result a *Proof of Work* is produced. What does this all mean? Let us start by understanding better what is the form of the *mathematical puzzle*: the chain is asking the miners to solve a certain type of problem, that is stated in the form "Find a string that when hashed gives as a result a string with a certain amount of leading zeros".

The solution of this puzzle is called *Proof of Work* (PoW) [39]. By definition of hash function, this string of data is difficult to produce, but it is easy to verify that it solves the problem. This *PoW* in particular is composed by the block of transactions plus a *nonce*, that in cryptography is a number, usually random, that can be used only once. Here is an example taken from Bitcoin wiki [15]:

we want to hash the string *"Hello, world!"* with *SHA-256* in order that the resulting string begins with "000" (three leading zeros). The solution is computed by varying the string adding an integer (the *nonce*) to the end, and incrementing it each time.

```
"Hello, world!0" => 1312af178c253f84028d480a6adc1e25e81caa44c749ec81976192e2ec934c64
"Hello, world!1" => e9afc424b79e4f6ab42d99c81156d3a17228d6e1eef4139be78e948a9332a7d8
"Hello, world!2" => ae37343a357a8297591625e7134cbea22f5928be8ca2a32aa475cf05fd4266b7
...
"Hello, world!4248" => 6e110d98b388e77e9c6f042ac6b497cec46660deef75a55ebc7cfdf65cc0b965
"Hello, world!4249" => c004190b822f1669cac8dc37e761cb73652e7832fb814565702245cf26ebb9e6
"Hello, world!4250" => 0000c3af42fc31103f1fdc0151fa747ff87349a4714df7cc52ea464e12dcd4e9
```

as it can be seen it takes 4251 tries in order to find the solution, that is actually not a big deal for most modern computers.

Bitcoin PoW does not work exactly like this, but in a very similar way: instead of asking to find a number with some leading zeros it asks to find a number that is lower or equal than a certain amount, known as *target*. So the *target* is a 256 bit number shared by all Bitcoin users, that has to be bigger or equal than the PoW hashed. This means that the difficulty of the puzzle can be set by choosing a sufficiently small target (as the target becomes smaller it is more difficult to find a number smaller than it). Bitcoin protocol chooses

the target [17] in order to set the average time of finding the PoW at 10 minutes. This is called the *difficulty* and it is proportional to the number of miners in the network, as it represents a blockchain answer (in order to keep the average computing time around 10 minutes) to the increasing number of nodes trying to solve the problem. As a remark the difficulty is not always increasing. Let us see why: if the number of miners increases then the difficulty increases too. This means that the smaller miners (those with the smallest computing capability among all the miners) will extremely get damaged by the more complexity, indeed they will not be competitive anymore and will be forced to get out of mining business. This leads to a decrease in number of nodes mining and consequently to a decrease of the difficulty too. In this way new (minor) miners can join the business, resulting in an increase in difficulty.

One thing to notice is that in the block of transactions the miner will add a preset amount of new currency as a reward for the job done. If he is able to find the PoW before anyone else he communicates it to the entire network and his reward is added to his wallet. Transactions can come with a certain amount of "fees" that are included by the transaction owners as a reward for miners. These fees are added to the solver of the problem wallet as well.

2 Why do we talk about Bitcoin trust?

As an introduction let us understand how Bitcoin market works [31]. Being Bitcoin a digital currency its market is completely electronic and transactions can be made through trading platforms. What is more being it decentralized there is not a central bank, in fact all the cash flows are regulated by the blockchain.

Bitcoin market is active every day of the week all day long. What is more it is completely transparent about the state of an investor order book. Being the blockchain public the transaction record is tracked and visible to investor, granting the absence of hidden volume of currency.

There are two costs associated to Bitcoin trading. The first one is a fee charged by the platform in order to cover its costs. This fee is charged both to the *maker* (the one who sends the money) and to the *taker* (the one who receives it) of the transaction. The larger the volume traded the smaller the fee. As an example in platform *Kraken* fees range from 0.16% for maker and 0.26% for taker for a volume smaller than 50,000 USD to

0.00% for maker and 0.10% for taker for a volume greater than 10,000,000 USD (source: www.Kraken.com/en-us/help/fees). Fees can be paid both in Bitcoin and in fiat currency.

The second cost is due to adverse selection, in fact there is a counterparty risk due to anonymity when a trader has private information.

A natural third cost of classical transactions is the inventory holding cost, covered by the bid-ask spread. This is not present in Bitcoin case due to the fact that platforms are not *market makers*, but only intermediaries.

Bitcoin transactions are taxed by countries differently. In the US the *Internal Revenue Service* treats digital currencies as *property*. So taxes associated to Bitcoin transactions are determined the same way of properties taxes. Europe on the other hand taxes Bitcoin like a currency. The *Court of Justice of the European Union* in October 2015 ruled that transactions made with Bitcoin are treated like a service and so not subject to VAT.

In order to investigate the market microstructure of Bitcoin let us discuss the importance of *liquidity*. Liquidity grants that trading is possible at any given time at fair price. As liquidity is difficult to capture, it should be studied using different measures. As an example *trading volume* can serve to this purpose, as it is an increasing function of liquidity [37]. Trading volume behaves differently in stock markets and in Bitcoin markets due to the fact that the second one never closes. In fact in stock markets the volume is *u-shaped* during the day, meaning that volume (and hence liquidity) have peaks at the beginning and at the end of the trading period. In general for Bitcoin markets this characteristic is not present.

Another measure for liquidity is the *liquidity ratio* [44]. This measure accounts for the price impact of trades and transactions costs, both negatively correlated to liquidity and both not considered by trading volume measure. The liquidity ratio measures the volume necessary to induce a 1% of price change:

$$LR = \frac{\sum_j \sum_t p_{j,t} V_{j,t}}{\sum_j \sum_t |\Delta p_{j,t}| \times 100}$$

where $p_{j,t}$ is the transaction price of day j at time t, $V_{j,t}$ the corresponding Bitcoin volume and $\Delta p_{j,t} = p_{j,t} - p_{j,t-1}$. The sum over j is made over a given number of days and the sum over t is made over the frequency of intraday observations.

A third measure to investigate liquidity is the *Roll* measure relating the

percentage effective spread [43]:

$$S_R = 200\sqrt{-Cov\left(\frac{\Delta p_j}{p_{j-1}}, \frac{\Delta p_{j-1}}{p_{j-2}}\right)}.$$

Notice that in this measure only j is present, since the measures compared are made day by day and not intraday.

Lastly the *spread measure* of *Corwin and Schultz* [29] can be used, which uses daily high and low prices:

$$S_{CS} = \frac{2(e^{\alpha}-1)}{1+e^{\alpha}} \times 100$$

$$\alpha = \frac{\sqrt{2\beta}-\sqrt{\beta}}{3-2\sqrt{2}} - \frac{\gamma}{3-2\sqrt{2}}$$

$$\beta = E\left[\sum_{k=0}^{1}\left[ln\left(\frac{H_{j+k}}{L_{j+k}}\right)^2\right]\right]$$

$$\gamma = \left(ln\frac{H_{j,j+1}}{L_{j,j+1}}\right)^2$$

where H_j and L_j denote respectively the high and the low prices observed over the period of two days j and $j+1$. The following table summarizes the liquidity measures discussed. All tables reported are derived by *Dimpfl* analysis. LR is denoted in local currencies, while S_R and S_{CS} are reported in percent. Dataset is computed over the period from November 2016 to January 2017.

Market	LR	S_R	S_{CS}
Bitstamp	3698.15	1.10	5.01
BTC-e	2924.27	-	4.74
HitBTC	8.19	1.11	3.97
itBit	1382.23	1.47	5.16
bitcoin.de	247.16	-	15.43
Kraken	3547.03	1.31	3.53

BTC China	697745.11	2.62	8.30
OKCoin	612732.25	2.70	8.35

Table shows that using liquidity ratio the most liquid market for trading in US dollars is Bitstamp, the most liquid for trading in Euro is Kraken and finally BTC China is the most liquid one when trading with Renminbi. Using an average exchange ratio between USD and Euro of 0.95 (during the sample period) one can verify that the liquidity of Bitstamp and Kraken are similar. In particular, assuming an average exchange rate of 6.92 CNY/USD (being the numbers in the table reported in local currency values) it can be seen that both BTC China and OKCoin are more liquid than US and European markets. So using this measure we find that *currently the most liquid Bitcoin markets are in China.*

The third column, representing the spread, is expressed in percentage, so values can be compared without any further conversion. Values for BTC-e and bitcoin.de can not be computed as they show a positive serial covariance of relative price changes. The fourth column still represents spread values expressed in percentage. We expect that the less the spread the more a market is liquid. The results that emerge from this analysis are completely in contradiction with the previous ones, in fact not only Bitstamp and Kraken are now more liquid than Chinese markets, but what is more *BTC China and OKCoin are now the least liquid markets.*

This contradiction provides an unclear result and it can not be clearly stated which market is the most liquid.

In order to reach a deeper understanding of the spread let us decompose it into adverse selection and transaction costs. Let X_t be an *unobservable* value of the asset following a random walk [34]:

$$X_t = X_{t-1} + \theta Q_{t-1} + \epsilon_t$$

where θQ_{t-1} represents the amount of the price change given by private information, Q_t indicates the trade direction: 1 represents a buyer and -1 represents a seller initiated trade. In general there is a quote midpoint M_t that deviates from X_t by the cost of inventory. In Bitcoin case the deviation is zero, since as we discussed early platforms are not market makers.

Here *Dimpfl* tries to suggest a parameter of Bitcoin valuation. He states that it *draws its value from traders level of confidence*. This depends on the *potential usage of Bitcoin*. Being the supply of Bitcoin fixed, the higher demand caused by these private information leads to a rising in the prices. In the model introduced earlier we assumed that X_t evolves according to private information too. In particular the this evolution in captured by the parameter $\theta = \alpha(S/2)$, where S is the spread and α is a parameter.

We now want to write an evolution equation for the price. In order to do that we model it as a random walk:

$$p_t = M_t + \frac{S}{2}Q_t + \eta_t.$$

Let us now, as stated earlier, substitute $M_t = X_t$ and take the difference $p_t - p_{t-1}$:

$$\Delta p_t = \alpha\frac{S}{2}Q_{t-1} + \frac{S}{2}(Q_t - Q_{t-1}) + e_t.$$

This equation is in the same form of the initial one, but here the *adverse selection cost is explicit*.

The parameter θ can assume other forms. Let us now allow that adverse selection and inventory holding component depend on volume [33]. Now θ can be written as $z_0 + z_1 V_t$. Doing the same procedure as in the Huang and Stoll model we obtain:

$$\Delta p_t = z_0 Q_t + z_1 Q_t V_t + e_t.$$

A last model is considered by *Dimpfl*: *Madhavan et al.* [20] assume that only the surprise component of the transaction volume affects X_t. This leads to the formula:

$$X_t = X_{t-1} + \theta(Q_t - E[Q_t|Q_{t-1}]) + \epsilon_t$$

with the conditional expectation of Q_t equal to ρQ_{t-1}, with ρ first order autocorrelator of Q_t. As usual, setting $M_t = X_t$ leads to:

$$\Delta p_t = \theta Q_{t-1} - \rho\theta Q_{t-1} + e_t$$

Let us now analyze the results obtained by applying these three models to the common Bitcoin trading platforms. Let us begin with the model of Huang and Stoll. Results are reported in the following table:

Market	S_{HS}	$s.e.(S_{HS})$	α	$s.e.(\alpha)$
Bitstamp	0.7827	0.0050	0.4634	0.0028
BTC-e	0.7762	0.0031	0.5382	0.0020
HitBTC	4.1506	0.0947	0.6431	0.0174
itBit	0.6973	0.0131	0.5220	0.0090
Kraken	0.8081	0.0038	0.5071	0.0024
bitcoin.de	13.5983	0.1409	0.6475	0.0065
BTC China	1.7381	0.0068	0.6376	0.0021
OKCoin	0.5532	0.0018	1.0511	0.0022

S_{HS} is the estimated spread in local currency and α represents the proportion attributable to adverse selection. Standard deviations are reported in the *s.e.* columns.

In all but one markets the 50% circa of the traded spread is due to the adverse selection. The only market that exceeds this is OKCoin, with a proportion greater that 100%, that is not meaningful from a mathematical point of view, but implies the fact that in this market there is a greater adverse selection effect. It appears in general that the traded spread is lower in markets where volume is higher, as already discovered earlier.

Results for the analysis made with the Harris and Glosten model are reported in the following table:

Market	S_{GH}	$s.e.(S_{GH})$
Bitstamp	0.6651	0.0727
BTC-e	0.7780	0.1233
HitBTC	4.2027	0.4228
itBit	0.5447	0.0733
Kraken	0.7800	0.0977

bitcoin.de	15.2394	0.0107
BTC China	1.7283	0.0683
OKCoin	0.5587	0.1513

As in the latter case S_{GH} is the model implied spread expressed in local currency and $s.e.(S_{GH})$ is its standard deviation. S_{GH} is computed as the sample average of the implied spread: $S_{GH} = 2(z_0 + z_1 V_t$. It can be seen that the results obtained now are comparable with the previous ones, in fact the values obtained for both S_{HS} and S_{GH} have respectively the same order of magnitude for the mean and means are consistent using standard deviation. What is different here is that the spread is completely given by adverse selection in this model.

Lastly let us see the results obtained applying the Madhavan model.

Market	θ	ρ	a	S_{MRR}
Bitstamp	0.3215	0.3653	-0.0058	0.6430
	(0.0021)	(0.0020)	(0.0007)	
BTC-e	0.2721	0.0745	-0.0004	0.5442
	(0.0012)	(0.0013)	(0.0004)	
HitBTC	1.8790	0.1094	0.0516	3.7580
	(0.0457)	(0.0100)	(0.0293)	
itBit	0.3471	0.4638	-0.0069	0.6943
	(0.0053)	(0.0031)	(0.0022)	
Kraken	0.3162	0.2241	-0.0011	0.6324
	(0.0016)	(0.0015)	(0.0005)	
bitcoin.de	4.9653	-0.0065	0.0418	9.9305
	(0.0598)	(0.0042)	(0.0175)	
BTC China	0.5574	0.0314	0.0000	1.1148
	(0.0016)	(0.0006)	(0.0005)	
OKCoin	0.3204	0.1908	0.0025	0.6408
	(0.0009)	(0.0004)	(0.0003)	

θ represents the price impact of adverse selection, ρ the first order autocorrelation of the trade direction and a is the drift. The model spread S_{MMR} (as usual in local currency) is computed as $S_{MMR} = 2\theta$. The terms in parenthesis represent the standard deviation measures. Results are compatible in order of magnitude to those obtained with the last two models. Here again the spread is fully attributed to adverse selection.

To sum up Bitcoin trading is risky due to its high volatility. Bitcoin market structure is transparent, without hidden volumes and without market makers. Trading platforms charge only a commission fee. The analysis suggests that the markets considered are fairly liquid. What is more the higher the trading volume the more liquid a market is, regardless to the type of currency the trades are made with. Liquidity does not vary in a regular way intraday. The analysis regarding the decomposition of the spread show that a significant part of it is due to private information. This private information is based on a personal and individual valuation of Bitcoin and it depends on whether the trader uses it as an *investment* or as a *form of payment*. As in Chinese markets a low spread is observed , it can be stated that private information is less important when the trading volume is high. This means that strong private information, that for example can be obtained by large pools of miners, have little pricing effects on liquid markets, as observed in Chinese Bitcoin platforms.

Quoting the author, "the value of Bitcoin largely relies on the general acceptance of Bitcoin, irrespective of its classification as investment or currency". The current academic consensus is towards the idea that Bitcoin can not be considered a currency, but only an investment. This is mostly due to its high volatility. On the other hand many companies and services are starting to accept digital money as a form of payment, for example Dell or Microsoft. Another leap for global Bitcoin acceptance and usage is the recent recognition of Bitcoin as a currency in Japan [32]. Starting 1 April 2017 Japan recognizes Bitcoin as a currency, alongside with classic fiat. This means that paying in Japan with this kind of digital money is now legal. Of course this law comes with an additional regulatory scrutiny for Bitcoin exchange platforms. What is more being now this digital money a legal currency, it means that regulations of banks and financial institutions can be also applied to

Bitcoin exchange platforms (like anti money laundering). This is made not only to minimize the illicit activities associated to digital money but also to protect the consumer.

As stated by *Dimpfl* this will result in a gain in Bitcoin price, since this new classification will raise general trust in cryptocurrency and more people will adopt it. As a result this will raise demand and then prices. Even if this adoption process might take a while, mainly due to the lack of understanding of the Bitcoin functioning and to its volatility, the usage of Bitcoin in Japan is expected to increase to $9 billion from 2017 to 2020 (in 2015 the cryptocurrency in circulation was worth $1.7 billion).

To stress more the concept that the value of Bitcoin is strongly linked to its acceptance, *The Economist* [19] in May 26th 2017, a month after the Japanese law on Bitcoin, reported a peak in Bitcoin price ($2.800 on May 25th 2017). What is astonishing is that a Bitcoin is now worth more than two ounces of gold. This increase in price is thought to be linked to Japan law, since in 2016 the 80% of Bitcoin transactions were made in Chinese yuan and now about a half are in Japanese yen. What is more this increase in price may not only be recognized in a money acceptance, but also in the rising of investment opportunities using Bitcoin. This is because of the *initial coin offerings* in which startups mint new kind of cryptocoins and sell them for already existing ones. This raises the demand for existing cryptocoins and speculative tradings between cryptocurrencies. This particular behavior in which the raise in price brings more buyers, that raise the price and hence the number of buyers could eventually be a bubble.

2.1 Bitcoin risks

Being Bitcoin decentralized, digital, without anything that can guarantee its value and without any proper regulation make it very risky. In particular risks can be divided in two types: risk derived by the structure of the protocol and users risk.

Let us analyze the first type of risks [27], keeping in mind that every mathematical protocol is potentially vulnerable to advances in technology. This means that protocols that will be here analyzed might be a solution only at the current state of technology. An ideal protocol has to satisfy the following three points:

- *rules*: participants must find and accept a mutual validation of trans-

action procedure;

- *state*: participants must have a common transaction log and must have a common pending transaction log;

- *value*: participants must agree on the digital currency value.

The inability to satisfy one of these rules leads to consensus risk and eventually to the closure of the protocol.

Moreover protocol is vulnerable to the following types of risks:

- *attack of the 51% risk*: protocol has to avoid the possibility that a large miner (with more than 50% of consensus) can take crucial decisions and hence represent the majority;

- *Goldfinger risk*: this type of risk takes his name from the 007 movie *Goldfinger*, where a multimillionaire wants to destroy the US gold reserve in order to modify Dollar value and in order to let his own gold reserve value grow. This traduces in the risk that a majority, a State or some other community would destroy the protocol in the name of any cause;

- *privacy risk*: technological advances and the increasing number of blocks of transactions could lead to a performing technique of profile analysis. In this case the anonymity of the wallets would be compromised;

- *price fall risk*: a price fall is followed by a reduction in mining activities due to less remuneration. As a consequence the whole protocol would be unused and would collapse;

- *Denial-of-service attacks risk*: a denial-of-service attack (DOS) consists in overcharging the target system requesting the use of an amount of resources that it can not sustain. This leads to the malfunctioning and the shut down of the system.

The analysis of these risks were made by the *European Banking Authority* in a document of July 2014 [23]. This document consists in a complete analysis of the risks that follow the use and the rise of virtual currencies. Note that cryptocurrencies are a subset of virtual currencies. Risks are classified into five classes: users risks, market participants risks, financial integrity risks,

payment system risks and regulation risks. Let us now see a list of these risks.

Users risks:

- user loses his money when the trading platform does some fraudulent actions. Being the "exchanger" (the platform) not based in any country it is not subject to any regulation either. Risk level is high;
- user loses his money when the exchanger refuses (or is not able) to convert digital money in fiat. Risk rises by the fact that anyone can anonymously create a cryptocurrency and create an exchanger of that currency without regulations. Risk level is high;
- digital money value falls unexpectedly. The change in a cryptocurrency value can be altered and controlled by large pool of miners. Moreover DOS attacks can deny the elaboration of transactions, changing prices. The lack of a central authority capable of stabilize the situation makes the risk level high;
- digital money is not currently taxable. If suddenly it would become taxable user should pay an unexpected high quantity of money. Risk level is medium;
- user can join *mining pools* and share the computational power of his computer in order to mine currency. Pools are completely unregulated, so owners can divide the mined currency unequally or can be not transparent on the effective use of the computational power. Risk level is low;
- being the members of a pool connected to internet it is possible that some sort of hack or malware could infect their computers and "steal" their computational power, giving it to a third part. Risk level is low;
- digital money protocol can be changed anytime if the majority of miners agree. This can lead the user to losses due to errors in the new protocol or due to some fraudulent action. Risk level is high;
- objective information about a specific type of digital money is not always obtainable. Moreover it is possible that someone could have private information. These facts linked with the fact that creating a new

protocol, and hence a new money, is easy and anonymous can lead the user to unexpected losses due to the fall of digital money value. Risk level is low;

- user could violate laws or regulations. Jurisprudence is still unclear in digital money field. Authorities could change norms and law (or their interpretation) fast and unexpectedly. Risk level is medium;

- users wallet information can be stolen from their computers and hence their digital coins too. This is not possible in the blockchain because of its design, but the private and public keys of wallets are stored as strings in files in computers. The theft of these files leads to the theft of currency too. What is more stolen digital coins are identical to non stolen ones. Risk level is high;

- user can suffer a loss when the exchanger is violated. Exchangers offer storage services but the lack of a regulation does not guarantees users refunds or any sort of cover. Moreover being the transactions registered on the blockchain irreversible, stolen coins can not be refunded. Risk level is high;

- theft of personal information is still a menace even in an anonymous system. This is because some protocols permit to access the personal wallet using an ID document, a scan of the iris or something similar. The usage of these sensible information is not subject to any regulation and hence user can suffer of identity theft. Risk is high;

- some contracts may become illegal due to the application of new laws. Risk level is medium;

- market operators can suffer from late accreditation of units of digital money or from a freeze of their positions. This is due to the possibility that (anonymous) counterparts have insufficient funds or to a temporary market illiquidity. Risk level is high.

- sellers have to accept fiat money. On the other hand they are free to refuse to accept digital money. This means that digital coins can suddenly become unusable and hence useless. Risk level is high;

- the loss of the private key of a wallet traduces in the loss of the coins inside the wallet too. There currently is not a service able to recover lost wallet passwords. Risk level is high;
- digital money conversion in fiat is not granted and is not necessary at a fair price, due to the lack of regulations. Risk level is high;
- money held by an exchanger can not be withdrawn without the exchanger permission. Moreover being all the deposit stored in an unique wallet held by the exchanger the risks of thefts or lack of funds are high;
- when an exchanger closes it is impossible to recover money from its fund. This leads to coin losses. Risk level is high.

Market participants risks:

- exchanger is insolvent and hence can not conclude transactions. This risk is due to the lack of governance mechanisms regulating these kind of third parts. Risk level is high;
- exchanger is not granted to have a total control over its operations. This can lead to insolvency and losses. Risk level is medium;
- *double spending* issue can lead to a loss of coins. The lack of a central authority lets this issue possible in badly designed protocols. Blockchain based protocols however are designed to protect against double spending. Risk level is medium.

Financial integrity risks are all classified as high level. Digital money system offers an efficient money laundering service. What is more the financing of illicit activities like terrorism is extremely easy with cryptocurrencies. Causes are linked to anonymity, to the fact that digital currency is not country-based and to the fact that anyone can create a new type of coin anonymously.

The presence of possible risks linked to the global economy are low but still real. Digital currency could lead to a financial crisis due to irregularities with cryptocoins transactions. What is more the crescent rise in digital money importance backed with the lack of regulations could bring to a crisis linked to protocol malfunctioning, creditors insolvency and late accreditation of transactions.

Last, a risk classified as medium is linked with regulation authorities operations. They are exposed to risks if they do nothing to regulate possible actions, if they deliberately decide not to regulate or if they introduce an unsuccessful regulation. Risks can involve juridic nature and reputation. In this case however the regulation authorities risk is completely in control of regulation authorities themselves.

To sum up the European Banking Authority analysis risks hidden in the use of digital money are several and linked to very different fields and market agents. Probably the riskier fact is that cryptocoins are not backed by any govern or country. This implies a lack of obligatory acceptance and an unclear valuation system. Secondly the lack of regulations lets users exposed to money theft and the introduction of new laws could damage existing contracts, pending transactions or the value of coins itself. Last being cryptomoney technology based, all risks deriving from hacking, malware or general malfunctioning are present.

2.2 Alt-coins: the evolution of cryptocoins.

One of the main characteristics of Bitcoin protocol is that it is *open source*. As a consequence it is extremely easy to use it as a base in order to create another cryptocurrency, possibly with completely different uses than Bitcoin. These currencies are called *alternative coins*, or *alt-coins*. We will see some examples of these alt-coins in order to understand ho Bitcoin protocol can be implemented in order to do various things. Alternatives to the blockchain are called *alt-chains*. These implementations of the blockchain use basic blockchain principles in order to exchange contracts, register sites domains and other various scopes. What is more the concept of proof of work has been used too. In particular the proof of work can be sided with *Proof-of-Stake* (where the new block added is chosen in a deterministic way, depending on the creator wealth) or *Proof-of-Publication* (to authenticate that some information has been published at a certain date).

An interesting implementation of the blockchain is the possibility to add to the transactions some string of information, using *metacoin* platforms. As an example the *Colored-Coin* associates string of information to Bitcoin quantities. In particular the Colored-Coin associates the ownership of a good

to Bitcoin. In this way a Colored-Coin can be used to exchange the property of the good (exchanging the Colored-Coin itself) or can be "decolored" and traded just as Bitcoin. The implementation of metadata in Bitcoin has a high potential since for blockchain design as long as that particular Bitcoin is not lost (only theft) the metadata will be included in it and will not mutate. Usually metadata are in the form of strings hashed in transaction blocks. This will provide them with protection to falsification and with a timestamp provided by the blockchain.

Alt-Coins are implemented using Bitcoin design, but on a different blockchain and on a different network. First alt-coins were created in August 2011 from Bitcoin source code. The first alt-coin were *IXCoin* [12] which main feature was the increased speed of mining and hence the reward. In September 2011 *Tenebrix* was created. Its importance is due to the fact that it was implemented with an alternative control algorithm (*scrypt* instead of the proof of work). Even if Tenebrix didn't spread it inspired *Litecoin* [13]. The latter uses scrypt and set the average time to create a block at 2.5 minutes (instead of 10 minutes for Bitcoin). As a consequence Litecoin is considered to be a lighter version of Bitcoin: "silver to Bitcoin's gold". The speed of the transaction confirmation (sided to a maximum of 84 million units of coin) traduces in a better ability to satisfy average consumer demand: it is not realistic that the average person would wait 10 minutes to have his coffee paid, while 2.5 minutes could be more acceptable.

In 2014 alt-coins protocols were more than 500 due to the easy implementation (for a complete list of alt-coins visit the site http://mapofcoins.com). For example *Dogecoin* [7] born in December 2013 on the basis of Litecoin. Dogecoin lowered the transaction block generation time to 60 seconds and uncapped its maximum quantity of coins.

Frecoin [9] born in July 2012 feature is its negative interest rate, in order to increase the transaction volume and lower the storage. Block generation time is set at 10 minutes.

Even if Bitcoin aimed to grant users anonymity, bigdata analysis is able to track nodes behaviors and connect them to real identities. Many alt-coins modified Bitcoin protocol in order to obtain the maximum possible anonymity and in order to be immune to bigdata analysis. The first implementation was *Zerocoin* [18] introduced in 2013 by the *Johns Hopkins University*.

CryptoNote [6] introduced in October 2013 constitutes the basis for anonymous digital cash. *Bytecoin* [4] born in March 2014 introduced an anonymous

currency based on an implementation of CryptoNote.

Metadata can be embodied in blockchain implementations too. These implementations are called alt-chains and their main function is not monetary, but they serve as a storage of pieces of information. Coins formally exist only as tokens containing data. The first type of alt-chain ever created was *Namecoin* [14]. Namecoin system it is mainly used to register site domains. It includes the cryptocurrency Namecoin, and it is used in order to storage data into the blockchain and to pay transaction fees.

Bitmessage [50] implements a safe and decentralized messaging system. Messages associated to transactions last two days and are canceled if they do not reach the node of destination. Senders and receivers are pseudonymous with a bitmessage address and their identity are heavily identified by the cryptography system. Bitmessages are crypt at the receiver and hence a third part reader should compromise receiver computer in order to read the message.

Ethereum [8] platform is used to manage contracts. It also has a cryptocurrency called *Ether*, used to pay the contract execution.

3 Smart contracts

At this point we saw how blockchain is used in order to perform Bitcoin transactions. Then the inclusion of metadata was shown as a demonstration of blockchain power. Last let us see what *smart contracts* are.

A smart contract [47] is a software able to negotiate an agreement between two parts, like a contract, using blockchain technology. As smart contracts are automated there is no need of a (trusted) third part that can guarantee the effective application of contract terms. In fact contract terms are implemented as coded instructions and automatically executed. As clearly emerges the power of smart contracts comes from the ability to conclude a contract *between two anonymous people that do not know each other without the regulation of a third part.*

Traditional contracts are written in legal language and rely on the presence of a third part. As a consequence the regulation of some possible fights over the contract terms is necessarily done by a judge, since contract terms might be ambiguous and in any case written in a technical language. On

the other hand smart contracts are completely written in computer language (like C++ or Python) and are digitally stored in strings. Differently from traditional contracts, smart contracts are executed by a distributed ledger like a blockchain. Smart contract is made of two parts: a *smart contract code* which is the code stored and executed on the ledger, and a *smart legal contract* that contains legal information.

The functioning of a smart contract can be sum up to three steps:

- first step is the coding. The importance of this step relies on the fact that the smart contract will do exactly what it is written in the code. This drives out both ambiguity and the possibility to contest possible contract scenarios, since all the decisions (and hence all the scenarios) are chosen during the coding of the contract;

- the second step is how the smart contract is distributed. The code is first encrypted and then sent to the blockchain. In Bitcoin case the sending of a smart contract is identical to the inclusion of metadata in a transaction block;

- last step is the contract execution. When the contract code is added to the ledger all the network participants agree on its application and results. Then at the processing of the next transaction block the result of the smart contract is added to the blockchain and hence timestamped and visible to all nodes.

The idea of smart contracts was inspired by *Szabo* [**?**] in 1994. His idea was followed by an example. Let us think about a vending machine. The distribution of coffee is managed by both the software and the hardware of the machine. The software verifies if inserted money is sufficient, the quantity of product to emit, etcetera. Hardware distributes the physical product. The actual implementation of the first smart contract was possible only twenty years later, in 2014 with the creation of Ethereum, the platform where smart contracts are mainly managed and distributed. The design of Ethereum combined with the power of the blockchain (the immutability of its log) made possible the creation of such contracts.

4 Disease spread: SI and SIS models

Epidemic models label individuals depending on their actual state of wealth. This action is called *compartmentalization*. The three simplest states of wealth are:

- *Susceptible (S)*: indicates those who don't already have contracted the pathogen;
- *Infectious (I)*: individuals who have contracted the pathogen. Infectious can infect other agents, resulting in the spread of the disease;
- *Recovered (R)*: individuals that have been infected but have recovered. In general a Recovered can't be infected anymore.

At *t=0* individuals belong to a certain compartment. As the disease spreads for $t > 0$ they can change their wealth state: a susceptible agent can get in contact with an infectious one and so get infected too. In general after some time an infectious agent will recover, meaning that he will develop immunity to the pathogen or he will die due to the disease. It is not really important how he recovers – meaning if he is still alive or he dies– because from a disease dynamic point of view he will not be in the model anymore. This is because he can't spread the disease anymore, so every connection passing through him will be unused.

In my model anyhow there only will be susceptible and infectious agents. An agent not trusting in digital money will be susceptible, while an agent trusting in it will be infectious. As it is not rational to develop a "digital money immunity", because if trust is sufficiently high it is useful to use it, only SI and SIS models are going to be analyzed and implemented.

The second hypothesis that permits us to fully describe these two kind of dynamics is the *homogeneous mixing hypothesis*. It assumes that every agent has the same probability of coming in contact with an infected individual.

4.1 SI model

Let the network be made of N agents. Assume that the average degree of the network is $< k >$, meaning that every agent has $< k >$ neighbors in average. Let now introduce β, that is the likelihood that the disease will be transmitted from an infected agent to a susceptible one in a unit time.

Let us now set the initial conditions infecting only one agent of the network: at time *t=0* there will be $S(0) = N - 1$ susceptible and $I(0) = 1$ infected individuals. How will the disease spread for $t > 0$?

The probability that the infected agent comes in touch with a susceptible one is $\frac{S(t)}{N}$, by the homogeneous mixing hypothesis. Being $< k >$ the average degree of connection, an infected person comes in touch with $< k > \frac{S(t)}{N}$ susceptible agents per unit time. Assuming there are *I(t)* infected individuals in the network, every of them transmitting the pathogen at rate β, the average number of new infected in dt is:

$$\frac{dI(t)}{dt} = \beta \langle k \rangle \frac{S(t)I(t)}{N}$$

Let for notation simplicity $s(t) = \frac{S(t)}{N}$ be the fraction of the susceptible population at time t and $i(t) = \frac{I(t)}{N}$ the fraction of the infected population at time t. Dropping the variable t we can write:

$$\frac{di}{dt} = \beta \langle k \rangle si = \beta \langle k \rangle i(1 - i)$$

where $\beta < k >$ is the *transmission rate*.

Separating variables and integrating both sides, we obtain:

$$\ln i - \ln(1 - i) + C = \beta \langle k \rangle t$$

In order to find the constant C we use the initial condition $i_0 = i(t = 0)$: set *t=0* in the equation and solve for C, obtaining $C = \frac{i_0}{1-i_0}$. Substituting we can find the fraction of infected individuals in function of t:

$$i = \frac{i_0 e^{\beta \langle k \rangle t}}{1 - i_0 + i_0 e^{\beta \langle k \rangle t}}$$

Conceptually this equation says that at the beginning of the dynamic there will be few infected individuals with many susceptible neighbors. This will result in an exponential spread of the disease. In particular the *characteristic* time required to infect a fraction of $\frac{1}{e}$ (that is circa 36%) of the susceptible agents is $\tau = \frac{1}{\beta}$. So increasing $< k >$ or β will result in a decreasing of τ, meaning that the more the network is connected or the more the "speed" of the disease is high the less it takes to infect the 36% of the susceptible agents.

Looking again at the form of the equation we notice that as the number of infect grows (and the number of susceptible obviously decreases) the number of new infections decreases, because the pathogen can come in contact with less number of healthy individuals. This means that i grows slower as t gets larger.

After a sufficiently large time t we will have *i=1* and *s=0*. This means that there are no nodes susceptible left and the simulation can end.

4.2 SIS model

This model equips the agents with an immune system (or healthcare). This means that the previous model will be implemented with a *recovery rate* μ. An individual recovered in this way returns susceptible and can become infected again. The dynamic is captured by:

$$\frac{di}{dt} = \beta \langle k \rangle i(1-i) - \mu i$$

where the term μi represents the rate at which the population recovers from the disease. Now the fraction of infected in function of time i is:

$$i = \left(1 - \frac{\mu}{\beta \langle k \rangle}\right) \frac{Ce^{(\beta\langle k\rangle - \mu)t}}{1 + Ce^{(\beta\langle k\rangle - \mu)t}}$$

similarly to the previous case we can find C using $i_0 = i(t=0)$:

$$C = \frac{i_0}{1 - i_0 - \frac{\mu}{\beta < k >}}$$

It is clear that in this model the dynamic can't evolve simply like in SI model: here the number of infected agents can fall down to zero, eradicating the disease, or remain stable in general.

Let us look at the first case, called *Disease-free State*. In this scenario the recovery rate must be sufficiently high in order to let i decrease exponentially. In other words the number of individuals infected per unit time is less than the number of individual cured per unit time. This happens when $\mu > \beta < k >$.

The second case is called *Endemic State*, and happens when $\mu < \beta < k >$, a low recovery rate. Differently from the SI case, the fraction of infected will not be *i=1*, but it will reach a constant value strictly less than one. This

is because the number of recovered individuals will be equal to the number of new infected agents, so the fraction i doesn't change. In particular using this last piece of information we can set $\frac{di(t)}{dt} = 0$ and obtain the number of infected when the dynamic is stationary:

$$i(+\infty) = 1 - \frac{\mu}{\beta \langle k \rangle}$$

What about the *characteristic time*?

Let $R_0 = \frac{\beta \langle k \rangle}{\mu}$ be the *basic reproductive number*, representing the average number of susceptible agents infected by an infected agent in a fully susceptible population. More clearly, R_0 is the number of new infections that each infected individual will cause in ideal settings.

The characteristic time can be now written as $\tau = \frac{1}{\mu(R_0 - 1)}$. This means that if $R_0 < 1$ we will have that $\tau < 0$ and the dynamic evolves according to the Disease-free state, so the disease will die since every infected individual will infect less than one additional agent.

On the other hand if $R_0 > 1$ then $\tau > 0$ and the dynamic will evolve according to the endemic state. In this case every infected individual will infect more than one additional agent and the disease is now spread. The greater R_0 the faster the spreading.

4.3 Diseases on Network

Having given a general description of SI and SIS models we can now go much deeper. In particular two assumptions are going to fall: the homogeneous mixing hypothesis and the fact that every node has a similar degree with everyone else. In fact agents can spread the disease only to their neighbors, so the network structure will be crucial, and we will work with scale-free networks, that are surely not well described by $< k >$.

We will need a formalism that includes the degree of a node as an implicit variable, as the more the node is connected the more he will likely be infected (or spread the disease). In order to do this the *degree block approximation* will be used. It incorporates degree as an implicit variable and assumes that nodes with the same degree are statistically equivalent. So now the fraction of infected nodes will be $i_k = \frac{I_k}{N_k}$ that is the fraction of infected nodes with degree k among all nodes with degree k.

4.4 SI model on a Network

We write the SI model differential equation for each degree k separately:

$$\frac{di_k}{dt} = \beta(1 - i_k)k\Theta_k$$

The form is similar to the one we saw in precedence. The different features are that now the average degree $< k >$ is substituted with the *actual degree* k of nodes. This is possible since we are treating the degree as an implicit variable. What's more the function Θ_k has been implemented. It represents the *density function*, that is the fraction of infected neighbors of a susceptible node with degree k. This is like a substitute for the homogeneous mixing assumption, where the density was the fraction of the infected nodes. Last, this is a system of k_{max} equations, one for each degree.

Let us now look at the disease dynamic. At the beginning of the simulation we expect the epidemic to be small. This means that since ik is small we can neglect it and the equation becomes:

$$\frac{di_k}{dt} \approx \beta k\Theta_k$$

Two important results regarding networks are that for a network without degree correlations the Θ_k function is independent from the degree k. Moreover the Barabási-Albert network (that we are using in the simulation) has not degree correlations. Having told these we can go further.

So, being τ^{SI} the characteristic time, the equation becomes:

$$\frac{di_k}{dt} \approx \beta k i_0 \frac{\langle k\rangle - 1}{\langle k\rangle} e^{t/\tau^{SI}}$$

with $\tau^{SI} = \frac{\langle k\rangle}{\beta(\langle k^2\rangle - \langle k\rangle)}$.

Integrating the equation for i_k we obtain the fraction of infected nodes with degree k:

$$i_k = i_0 \left(1 + \frac{k\left(\langle k\rangle - 1\right)}{\langle k^2\rangle - \langle k\rangle}\left(e^{t/\tau^{SI}} - 1\right)\right)$$

what is more we have that the total fraction of infected nodes is:

$$i = \int_0^{k_{\max}} i_k p_k dk = i_0 \left(1 + \frac{\langle k \rangle^2 - \langle k \rangle}{\langle k^2 \rangle - \langle k \rangle} \left(e^{t/\tau^{SI}} - 1 \right) \right)$$

What is clear is that the higher the node degree the more likely it will become infected.

4.5 SIS model on a Network

Let now extend what we found to the SIS model:

$$\frac{di_k}{dt} = \beta(1 - i_k)k\Theta_k(t) - \mu i_k$$

$$\tau^{SIS} = \frac{\langle k \rangle}{\beta \langle k^2 \rangle - \mu \langle k \rangle}$$

Notice that for a sufficiently large μ the characteristic time is negative, so i_k decays exponentially. What is more the characteristic time now depends from μ, $< k >$ and also $< k^2 >$, that is the *network heterogeneity*.

In order to study better the dynamic let introduce the *spreading rate* $\lambda = \frac{\beta}{\mu}$, that depends only on disease biological characteristics. The higher λ the higher is the spreading rate respect to the recovery rate, so the more likely the pathogen will spread. The crucial fact is that *a disease can spread only if the spreading rate is higher than an epidemic threshold* λ_c.

As we are working with scale-free networks now this case will be treated. For a network with an arbitrary degree distribution, setting the characteristic time greater than zero it can be found that the epidemic threshold is $\lambda_c = \frac{\langle k \rangle}{\langle k^2 \rangle}$. The fact that for $N \to +\infty$ in a scale-free network $< k^2 >$ diverges means that for sufficiently high N the epidemic threshold goes to zero. This implies that every kind of virus, even the ones that find it difficult to pass from individual to individual, can spread.

In a *BA* network it is the case that the exponential of the power-law $\gamma = 3$. In this case we have the following results for a SIS model:

$$\lambda_c = 0$$

$$\Theta(\lambda) \approx \frac{e^{-1/k_{min}\lambda}}{\lambda k_{min}}(1 - e^{-1/k_{min}\lambda})^{-1}$$

$$i(\lambda) \sim 2e^{-1/k_{min}\lambda}$$

5 Model description

The model goal is to understand how and under what conditions the agents *trust* in the digital money, here called for simplicity *Bitcoin*. Moreover sided to the trust function an *utility function* will be implemented.

The agents are nodes of different networks. Every network has its own characteristics and the agents are going to behave differently depending on the type of network they are trading in. For example a node that is trading money in a *legal* network (meaning that he is trying to buy a generic legal good, like a financial asset) is obviously facing different threats than a node that is buying drugs in an *illegal* one: not fulfilling a contract in a legal network can lead to a legal report, doing so in an illegal one can lead to death. This means that the trust in a digital money is going to emerge differently among nodes of different kind of networks. Moreover not only the type but also the *dimension* and the *density* (that is the average degree of a node) of the network is going to influence the spread of the digital currency.

The model used for the simulations allows nodes to belong to different networks. Differences between networks are crucial in order to understand the agents behaviors. As anticipated an illegal network will have different issues and characteristics respect to a legal one. We saw how in a legal network it is way less risky to be in debit with someone, but being in credit is safer too. In fact any legal action against not paying debtors it is obviously impossible when trading illicit goods. What's more from a taxation point of view any transaction made in the black market is clearly off the books, while every exchange done in a licit market can be taxed (if the members are not anonymous, as we will now see).

Side by side with the legal/illegal network there is the *anonymous/not anonymous* one. By not anonymous I mean that every public key associated to a wallet is equipped with an identity document attesting the wallet ownership. In this way it is possible not only to monitor every exchange made by a

specific member of the blockchain (this was already possible as it is public), but that member will also have a *real identity* and not just a fictitious name. The security implications are both amazing and dramatic.

On the other hand there can be a total anonymity granted to the nodes, as it is the case of Bitcoin. Every wallet is associated to a public key that acts as a fictitious name of the owner of the wallet itself. This leads to various consequences. First of all being the real person behind a node not traceable, no one is really discouraged from doing any kind of illegal activity on the chain. For example *Silk Road* was a website (closed in 2014) accessible only using the anonymity software *Tor* where it was possible to purchase any kind of goods – mostly drugs and arms – using Bitcoin, thanks to the complete anonymity that this currency grants [40].

Moreover nobody can be sure about any wallet ownership. This means that one single person can have presence in different locations of the network without being detected or recognized. This can lead to various fraudulent actions as there is no evidence that two different agents that are trading a good for some amount of money are actually different individuals.

The network is made by using the *Barabási-Albert algorithm* [24] : a first number of initial nodes is set and they are fully connected. Then the *number of links per iteration (L)* is fixed. At each iteration of the algorithm a new node is created and is connected to L already existing nodes. This process is repeated until the network reaches the desired dimension. In this way a *scale-free network* is created. As a consequence we will notice that the distribution of degrees of nodes follows a *power law*, instead of a *Poisson distribution* as in a random network.

This means that our network will have a lot of nodes with a low degree and very few nodes with a high degree.

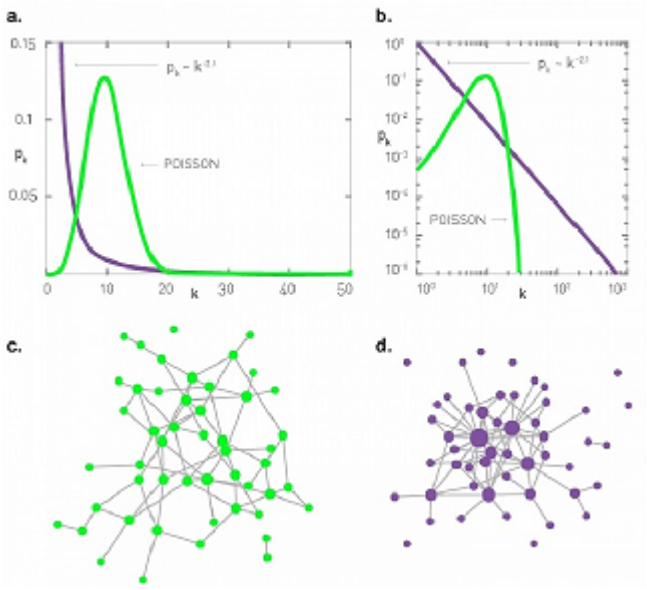

Figure 2: Random network versus Barabási-Albert network distribution of degrees. Source: Albert-László Barabási. Network science, chapter 4.

Figure 1 compares the distribution of degrees of a random network versus a Barabási-Albert network. It can be seen that the power law distribution is greater than the Poisson one when $k \ll <k>$. In the neighborhood of $k = <k>$ the Poisson distribution is greater than the power law one, in fact the maximum value of the Poisson distribution is at the mean value of k. For $k \gg <k>$ both Poisson and power law distributions go to zero, but the first one decreases faster than the second one. This means that in a scale-free network we have a small but significant number of nodes with k greater than the degree of the most connected nodes of a random network. Such kind of high connected nodes are called *hubs* and their state influences the properties of the whole network dynamic.

It is a feature of large networks to be scale-free [21]. This means that for large k we will have a power law distribution of degrees. It is observed that random graphs can't reproduce this property, as their distribution results in a Poisson one.

But why are *real* networks scale-free? Evidences show this property in real networks like the *World Wide Web* or the *references network*. When we will talk about the network dynamic properties we will see how the scale-free feature captures them. Most real networks grow continuously by the

addition of new nodes, for example the WWW grows when new web pages are published. It is crucial to underline that a random network assumes that the probability of linking two nodes is independent of the nodes degree, in fact the attribute *random* means that the probability of creating a link between two nodes is uniformly distributed between 0 and 1.

This characteristic is not visible in most of real networks. In fact they exhibit *preferential attachment* feature, that links the probability of connecting to a node to that node degree. For example a brand new paper is more likely to cite well known and already a lot cited papers.

Growth and *preferential attachment* inspired the *Barabási-Albert algorithm*: we start with a number N of nodes and at every time step a new node is added with L links to already existing nodes. This is how the network grows. The preferential attachment is performed every time the new nodes have to choose the old nodes to link with. The probability that a new node will connect to a node i is given by the following:

$$\pi(k_i) = \frac{k_i}{\sum_j k_j}$$

where the denominator is the sum of all nodes degrees. As it is clear in this algorithm the probability of connection depends on k_i. Numerical simulations show that a network generated with this method evolves into a power law distributed with exponent $\gamma = 3$. As we can see the only two parameters of the model are the initial number of nodes N and the number of links created at each time step L. A remarkable result, that can be proved, is that gamma doesn't depend by N.

Let us write an equation for the time dependence of the degree k_i of a node i, consider for simplicity continuous quantities. At every time step there is a probability $\pi(k_i)$ that a new node will link to the node i. So the dynamical equation becomes:

$$\frac{\partial k_i}{\partial t} = L\pi(k_i)$$

Being t the time step, at time t the value of the sum of the degrees over *all* nodes is *2Lt*. But in this case we can not consider the new node in the sum, because it has not already been added to the network. So, remembering that $\pi(k_i) = \frac{k_i}{\sum_{j=1}^{n-1} k_j}$, where *n-1* are the nodes already in the network, we have

that the sum at the denominator is equal to *2Lt - L*. So the equation becomes

$$\frac{\partial k_i}{\partial t} = L\frac{k_i}{2Lt - L} = \frac{k_i}{2t - 1} \simeq \frac{k_i}{2t}$$

where the last equality is true for large values of t. In order to find a solution we set the initial conditions $k_i(t_i) = L$ when the node i is added to the network at time t_i. The solution of the equation is:

$$k_i(t) = L\left(\frac{t}{t_i}\right)^{\beta}$$

with $\beta = \frac{1}{2}$. This means that the degree of the nodes follows a power law and that all nodes evolve the same way. The probability that a node has degree $k_i(t) < k$ is given by

$$P[k_i(t) < k] = P\left[L\left(\frac{t}{t_i}\right)^{\beta} < k\right] = P\left[\frac{t}{t_i} < \left(\frac{k}{L}\right)^{\frac{1}{\beta}}\right] = P\left[t\left(\frac{L}{k}\right)^{\frac{1}{\beta}} < t_i\right]$$

Let us now find a form for t_i. Assume that the nodes are added to the network at equal time intervals, so the t_i have a constant probability density given by

$$P(t_i) = \frac{1}{N + t}$$

where N is the number of nodes at $t = 0$. This implies that

$$P\left[t_i > t\left(\frac{L}{k}\right)^{\frac{1}{\beta}}\right] = 1 - P(t_i)\frac{L^{\frac{1}{\beta}}t}{k^{\frac{1}{\beta}}} = 1 - \frac{m^{\frac{1}{\beta}}t}{k^{\frac{1}{\beta}}(N + t)} = P\left[k_i(t) < k\right]$$

Now we can find the distribution *P(k)* of the degrees as follows

$$P(k) = \frac{\partial P\left[k_i(t) < k\right]}{\partial k} = \frac{L^{\frac{1}{\beta}}t}{k^{\frac{1}{\beta}+1}\beta(N + t)}$$

taking the limit $t \to +\infty$ we get

$$P(k) \sim \frac{1}{\beta} L^{\frac{1}{\beta}} k^{-\gamma}$$

with $\gamma = \frac{1}{\beta} + 1$. Remembering that $\beta = \frac{1}{2}$ we obtain that $\gamma = 2 + 1 = 3$, as predicted by numerical simulations and independent from L. What's more the coefficient of the power law distribution is proportional to L^2.

Let us return now to the model description. At the beginning of the simulation every agent is equipped with an amount of traditional currency and a trust (in digital currency) function. The number of different networks, the dimension and the density can be set independently. Then a certain amount of Bitcoin is given to only one node (or to a small number of connected nodes). In general the first node with available digital currency is not confident in it, so he will try to get rid of it. As it will be explained in details later, agents can choose whether to use traditional or digital money. The choice depends on two factors: the *buyer utility function* and the *seller trust function*. Intuitively the buyer is going to decide what type of money to use depending on the utility he is expecting to gain from the exchange. Similarly the seller is going to accept the digital money only if he is sufficiently confident about it. What's more as he is now infinitely confident in the traditional currency he will always accept it and there is no need for a "traditional trust function". Theoretical structure of both functions will be discussed later.

It is important to remark that I am not trying to redefine the neoclassical meaning of utility function or enter anyhow in an utility debate. What is here meant by utility is a specific metric created *ad hoc*, to order agents preferences. As a result a number will be obtained and will be compared to a specific threshold parameter.

In order to let the agents choose between the two currencies they will be equipped with a *value memory* too, that is the value that had the currencies when they received them. These are numbers, one for each type of money in the network. To be more clear: let us assume that agent A received 1 Bitcoin at time *t-1* and now at time t he is thinking about buying something with that Bitcoin. The utility associated to the exchange with Bitcoin has to take in account the difference between the value of Bitcoin at *t-1* and at t. If the transaction succeeds then the buyer and the seller will both update the memory value of Bitcoin with the value at time t.

To reflect the volatility of Bitcoin value – compared to traditional currency – it will follow a sinusoidal function, precisely the value of one unit of good is *2 Euros* (traditional currency) or *2 + sin(t) Bitcoins* (digital currency), where t is the discrete time. As it appears clear the value of Bitcoin goes from 1 to 3.

5.1 Gresham Law

Gresham Law is sufficiently well described on Wikipedia [11] and many other Internet resources. For our purposes it is sufficient to understand the form in which it is usually stated: *"bad money drives out good"*. For example it states that if there are two commodity money in circulation with the same face value, the more valuable one will then disappear from circulation.

What about Gresham Law and Bitcoin? The law implies that the "bad" money is *overvalued* and so you want to spend it, while the "good" one is *undervalued* and you want to keep it. Let us take an example [26]: consider a $100 bill and a 1oz Gold Double Eagle $20 coin. The $100 bill has a higher *nominal value* (that is $100 > 20$) than the coin, however the gold coin commodity value is worth more than $1000 (while producing a $100 bill costs 12.5 cents [45]) so it is senseless to spend it as a $20 coin. This is why in the USA this kind of coin is not traded as money anymore, while the $100 bill obviously is. This is what Gresham Law means when referring to bad and good money: the bad one has nominal value higher than the commodity value, on the contrary the good one has a higher commodity value. Being the commodity value the worth of the material that the money is made of, the Bitcoin commodity value is the *electrical energy that is used to mine it.*

While in the past the nominal value of a currency was linked to something, like gold, nowadays we use *fiat money* without intrinsic value. This means that the value of Dollars, Euros and all the other fiat money is decided by the market alone and can free-float, just like the Bitcoin. The reason why we use Euros in Italy is because we know that everybody in Italy will accept this currency as a payment and, more important, *we can pay taxes with it.* This is what I mean by equipping agents with a trust function. An Italian agent is fully confident in Euros, as we said because he can buy any kind of good with it and because there is a major debtor – the state – that will accept it in form of taxes. So why should an agent with such a currency switch to a "trust less" digital money and in which conditions?

Let us focus on Gresham Law applied to Bitcoin in order to determine

if it is overvalued or undervalued. Some indicators could be the transaction volume, the cost of bitcoin mining, the number of market participants and the bitcoin day's destroyed metric, that is a transaction measure that gives more weight to bitcoins that haven't been spent in a while. The transaction volume can act as an indicator: if Bitcoin is overvalued then it will be spent more than fiat money and vice versa if it is undervalued people would rather spend fiat. Remembering about the major debtor issue we notice that miners can't pay taxes using Bitcoin. This means that they need to sell Bitcoin and buy fiat in order to pay taxes. As a consequence of this selling the price of Bitcoin could fall, implying that more quantity of digital money would be required in order to buy the same amount of fiat, obliging miners to sell more and accelerating so the drop in price.

What can be concluded is that digital currencies are undervalued now: there is a limit in the quantity of bitcoin that can be mined, they are based on the quantity of energy used in order to mine them and they live in the Internet world, which is the biggest and fastest growing economy in the world. This means that in my model it can actually be possible that all the agents decide to switch to digital money, basing on the Gresham Law. The facts to understand and analyze now are the conditions – the parameters critical values – under which this happens.

5.2 Trust function

The way how agents will decide to use traditional or digital money is inspired by Luther 2015 [36]. In his paper he models a network with M nodes that can trade a traditional currency. The *utility function* of agents using the money at time T is

$$u(T) = (a + bn) \int_T^{+\infty} e^{-r(t-T)} dt = \frac{a + bn}{r}$$

with a, b fixed parameters, r *discount rate* and $n = ln(N)$, where N is the number of agents using the same currency. The number n contains the *network effects* and bn captures the *network-related benefits (if $b > 0$)* that an agent receives from using the same money as the other $N - 1$ agents in the network. On the other hand the factor a is independent from network size.

Suppose now that a new kind of currency (a digital one in our case) becomes available at $T = T^*$. In this model agent can use *only one* currency and he has to figure out if using the new digital money is better than continue to use the old money. In particular if an agent chooses to switch to digital money he has to face a *one time fixed cost of switching s*. Let us now imagine that N agents switch to new money. Their utility will be

$$v(T) = [(c + dn) \int_T^{+\infty} e^{-r(t-T)} dt] - s = \frac{c + dn}{r} - s, \quad T \geq T^*$$

Switching at T = T* increases aggregate welfare if and only if

$$Nu(T)_N < Nv(T)_A$$

where $u(T)_N$ is the utility function of an agent switching when no other does, while $v(T)_A$ is the utility function of an agent switching when everyone does. Substituting we obtain

$$N \frac{a + bn}{r} < N [\frac{c + dn}{r} - s]$$

So when is it optimal to switch? It is *socially* optimal when

$$s < \frac{c - a + (d + b)n}{r}$$

that is when the cost of switching is less than the net gain in utility from the switch. On the other hand it is *individually* optimal to switch *independently to other agents choices* if

$$u(T)_N < v(T)_N$$

(notice that we are now using $u(T)_N$ since it is the utility of the agent when he switches and no one else does).

Since there are no network benefits $v(T)_N = \frac{c}{r} - s$. So substituting, if $s < \frac{c-a-bn}{r}$ an agent will switch.

Conversely an agent will continue to use old money (even if everyone else is switching) if

$$u(T)_A > v(T)_A$$

where now we used the utility function $u(T)_A$ because every one apart from the agent is switching.

Since there are no network benefits $u(T)_A = \frac{a}{r}$ and the agent will continue to use the old money if $s > \frac{c-a+dn}{r}$.

To sum up when $s < \frac{c-a+(d-b)n}{r}$ it is *socially* optimal to switch, while for $s > \frac{c-a+(d-b)n}{r}$ it is *socially* optimal for no agents to switch.

What happens when $\frac{c-a-bn}{r} \geq s \geq \frac{c-a+dn}{r}$? There can be two *suboptimal* cases in which agents continue to use old money when it is socially optimal to switch (*excess inertia*) and vice versa they change money when it is socially optimal not to (*excess momentum*).

My model differs from this in three distinct aspects:

- the network effects implemented by Luther as $n = ln(N)$ have the same logarithmic form, but N is now not the total amount of nodes that use that kind of currency, but the amount of *neighbors* (connected nodes) that use that currency. In any case the component given by the total amount of agents using a currency will be still considered;
- agents will have in their wallets any kind of currency and they can decide whether to use traditional or digital money. This is important as the *cost of switching* will be slightly different from Luther model;
- a trust function will juxtapose with this kind of utility choice.

What about the form of the trust function? As said before, every node is equipped with a trust value. The form of the function computing this value has to depend on:

- the number of transactions in Bitcoin made by neighbors;

- the number of total transactions (Bitcoin plus traditional money) made by neighbors;
- a certain parameter identifying the network type;
- the number of neighbors using Bitcoin;
- the total number of Bitcoin users in the network.

6 Program description

The language used to code the program is *Netlogo* [48]. In order to work with networks the extension *Nw* is included. This extension adds some interest tools useful for network analysis and node dynamic. The *Barábasi-Albert model* construction is inspired by an already existing model in the Netlogo Library: *Preferential Attachment* [49]. The model starts with two connected nodes. At each time interval (named *tick* in Netlogo) a new node is created and connected with a fixed number of pre-existing nodes. The node to connect with is chosen using preferential attachment, that is an algorithm that gives to high connected nodes a higher probability of connection.

```
to create-network
  let u 0
  while [u < (n - 3)][
  ask edges [ set color gray ]
  make-node find-partner
  set u u + 1]
end

to make-node [old-node]
  create-nodes 1
  [
    set color red
    if old-node != nobody
      [ create-edge-with old-node [ set color green ]
        move-to old-node
        fd 8
```

```
      ]
  ] set numm numm + 1
end

to-report find-partner
  report [one-of both-ends] of one-of edges
end
```

These three *procedures* build the network following preferential attachment algorithm. The procedure *make-node* has an input parameter *old-node*. What this function does is creating a new node (agent) and, if the already existing node given as input (old-node) is a valid node – meaning that it really exists – the new node will connect to it and will move next to it. It is interesting to cluster the connected nodes putting all of them close to each other. In this way a more clear graphical vision of hubs can be obtained.

The *find-partner* procedure assigns to every node a ticket, like in a lottery. In this way the probabilities of connection are set and the new node can decide the old node to connect with.

Finally the *create-network* procedure will build the network: a *for cycle* will add *n-2* nodes (using the previous two procedures) to the initial network – made of 2 nodes. This will result in a *BA* network with n nodes and n-1 links.

In order to understand the dynamic of an anonymous network a certain number of *"real identities"* are added. To be more clear: the nodes in the network I have just created with the *BA algorithm* represent *wallets*. I will now create another network that links *wallets* to *owners*, that are the real identities. Being this an anonymous network, the owners don't have the possibility to directly interact. They can only exchange money and goods using their wallets (that are *fake identities*) indeed. So owners are effectively *invisible* and the edges linking them to wallets (that are visible) are invisible too. This results in a visible network made by wallets (that are identified by public keys) and an invisible network made by owners and their wallets.

The most important implication is that a single owner can have presence in different parts of a network without having his wallets directly connected in that network. This is because he can decide to *reallocate* his resources by

taking money from one of his wallets and putting them into another one. For our purposes this means that potentially an agent with a very low trust in the digital money may "suddenly" develop a belief in it without an apparent reason. By a node in the visible network point of view this is a no reason (irrational) change of mind, for the owner of that wallet this is completely rational instead: he learned to trust the digital money using one of his wallets and all of his other wallets are influenced by this.

The procedure implementing the invisible network is here reported:

```
to connect_nodes_and_invisible
  let o count invisible_nodes
  let till o + numm
  let g numm
  let r 0

  ask nodes [create-invisible_edge-with one-of invisible_nodes with
 [invisible_edge-with myself = nobody]]
  ask invisible_edges [set color grey]
  ask invisible_nodes with [count invisible_edge-neighbors = 0] [die]
end
```

A number n' of invisible agents are created and they are connected randomly to each wallet. In this way every wallet will be connected to *one and only one* owner. Every invisible agent without a wallet is then destroyed. The distribution of connections is uniform for the invisible network, in order to let every owner have similar network presence. More particular distributions can be implemented to let emerge different aspects.

Let us now talk about the digital currency, the Bitcoin (or *Btc*). Data evidences show that the value of this cryptocurrency respect to the value of a traditional one – say the Euro – is really volatile [1]. The fact that the Bitcoin value differences can be significant from one day to another is relevant when talking about trust. Agents will be more reluctant towards a currency with such a volatility, especially when the holdings are high. As discussed previously this feature can though be positive for the spread of the

digital currency, in fact for the Gresham Law if holdings are risky then agents will try to spend the money as fast as possible, transforming it in a *"good money"*. This fast change in value is going to affect how an agent chooses what kind of currency to use (the utility function). In fact if one believes that the Bitcoin value will grow he will tend to keep it, while if one expects that it will fall down he will try to spend it. So the agents are equipped with some kind of prediction tool, here represented by a *moving average*. Both the present value of the Bitcoin and the moving average value will participate in the agents expectations: if the Bitcoin value is currently lower than the moving average nodes will expect it to fall more, resulting in the sell of the currency, on the other hand the growth in value of the Bitcoin will make agents think that it will continue to grow, resulting in them saving it.

Various functions can be implemented in order to let the Bitcoin change value respect to the traditional money. In this model Bitcoin value changes according to a sine function and to a random walk

The Bitcoin value is computed following:

$$value_{btc} = value_{traditional} + sin(t) + r$$

where r is a random number belonging to the interval $[-\frac{1}{2}, \frac{1}{2}]$.

```
to compute-value-bitcoin
  let re ((random-float 1) / 2)
  let h random-float 1
  ifelse (h > 0.5) [set re (re * (-1))][set re re]
  set bitcoin_value (2 + (sin ticks) + re)
end
```

The change digital-traditional money is computed too using the trivial formula

$$change_{\frac{traditional}{digital}} = \frac{value_{traditional}}{value_{digital}}$$

```
to compute-value-bitcoin
  set bitcoin_value (traditional_value + sin ticks)
end
```

```
to-report compute-change-btc-eur
  let change-btc-eur (traditional_value / bitcoin_value)
  report change-btc-eur
end
```

Memory of agents regarding Bitcoin value is implemented as follows. Note that agents are equipped with an initial memory:

```
to-report compute-initial-memory [ttt]
  let re ((random-float 1) / 2)
  let h random-float 1
  ifelse (h > 0.5) [set re (re * (-1))][set re re]
  report (2 + (sin ttt) + re)
end

to create-value-list
  set mylist []
  let tt (-20)
  while [tt < 0][
    set mylist lput (compute-initial-memory tt) mylist
    set tt (tt + 1)
  ]
end
```

The moving average is coded as the following procedure:

```
to-report compute-mean
  let btc_mean ( sum mylist / 20)
  report btc_mean
end
```

Every agent has a limited memory that can contain a fixed number of past bitcoin values (20 in this case). As the memory doesn't go that far in the past there is no need to weight the values, so a simple moving average is used. As in the real world, Bitcoin quotations are public, so every node knows the preset value of the digital money.

The following procedures are used in order to study the evolution of the diffusion process. In particular the state of each agent (Bitcoin user or non-user) is saved in an array. When all agents have turned into Bitcoin users the procedure time will give as output the present number of model cycles.

```
to create-bitcoin-list
  set bitcoin_user_list []
  let ko 0
  while [ko < 100][
   set bitcoin_user_list lput 0 bitcoin_user_list
   set ko (ko + 1)
  ]
end

to create-transaction-list
  set trans_t 0
  set trans_t-1 0
end

to time
  ifelse (min bitcoin_user_list = 10) [set time_ten ticks][set time_ten 0]
end
```

This procedure counts the number of Bitcoin users neighbors of a node. It is useful in order to compute the first factor of the Trust function.

```
to-report bitcoin-user-neigh [thisnode]
  let pp 0
  ask thisnode [
    ask edge-neighbors [
      if (bitcoin_user > disease_threshold) [set pp (pp + 1)]
     ]
  ]
  set pp (pp + 1)
  ifelse (pp > 0) [report ln pp][report 0]
end
```

Now Trust and Disutility functions are computed. The procedures used are the following:

```
to trust-function
  ask nodes[
    let network-contribute (bitcoin-user-neigh self)
    let trans-contribute ((sum[bitcoin_transactions] of edge-neighbors)
                   / (sqrt (sum[total_transactions] of edge-neighbors + 1)))
    let tot-trans-contribute ((sum[bitcoin_transactions] of other nodes)
                         /(sqrt (sum[total_transactions] of other nodes + 1)))
    let value-contribute ((bitcoin_value - compute-mean) / 3 )
    set trust ((omega + (alpha * network-contribute)
                                    + (beta * trans-contribute)
                            + (gamma * tot-trans-contribute)
                                + (csi * value-contribute)) / 4)
    set trust max list 0 trust
  ]
end

to-report inverse-trust
  ifelse (trust = 0) [report 1][report (1 / trust)]
end

to disutility
  ask nodes[
  let trust-contribute (da * inverse-trust)
  let btc-value-contribute ((bitcoin_value - compute-mean) / 3)

  set disty ((trust-contribute + (((-1) * (dc))* btc-value-contribute)) / 2)
  set disty max list 0 disty
  ]
end
```

Last, the *buy* procedure is shown. This procedure determines the system dynamics, as it allows agents to exchange goods and money. At every time each agent tries to buy goods from one of his neighbors. Each transaction is

done simultaneously. If DisUtility of buyer is sufficiently high he will propose seller to conclude the transaction using Bitcoin. If the Trust of the seller is sufficiently high he will accept it and the transaction is registered.

```
to buy
  compute-value-bitcoin
  set trans_t-1 trans_t
  set this_trans 0
  let ty 0
  trust-function
  disutility

  while [ty < number_nodes] [
    ask node ty
     [
      let tt total_transactions
      set b bitcoin
      set m money
      set diss disty
      set bit_t bitcoin_transactions
      set bit_us bitcoin_user

      ask one-of edge-neighbors[
       ifelse ( diss >= threshold-disutility)
       [
         ifelse (trust >= threshold-trust)
         [
           ifelse (b >= ((compute-change-btc-eur) * 2 ))
           [
           set b (b - ((compute-change-btc-eur) * 2))
           set bitcoin (bitcoin + ((compute-change-btc-eur) * 2))
           set bit_t (bit_t + 1)
           set bitcoin_transactions (bitcoin_transactions + 1)
           set number_bitcoin_transactions (number_bitcoin_transactions + 1)
           set number_total_transactions (number_total_transactions + 1)
           set bit_us (bit_us + 1)
           set bitcoin_user (bitcoin_user + 1)
           set tt (tt + 1)
```

```
            set total_transactions (total_transactions + 1)
            set this_trans (this_trans + 1)
            ]           [
            ifelse (m >= 2)
            [
            set m (m - 2)
            set money (money + 2)
            set number_total_transactions (number_total_transactions + 1)
            set tt (tt + 1)
            set total_transactions (total_transactions + 1)
            ]           [           ]           ]           ]
            [
            ifelse (m >= 2)
            [
            set m (m - 2)
            set money (money + 2)
            set number_total_transactions (number_total_transactions + 1)
            set tt (tt + 1)
            set total_transactions (total_transactions + 1)
            ]               [               ]           ]         ]
            [
            ifelse (m >= 2)
            [
            set m (m - 2)
            set money (money + 2)
            set number_total_transactions (number_total_transactions + 1)
            set tt (tt + 1)
            set total_transactions (total_transactions + 1)
             ]            [           ]         ]         ]
        set bitcoin b
        set money m
        set bitcoin_transactions bit_t
        set bitcoin_user bit_us
        set total_transactions tt
        ]
        set ty (ty + 1)
    ]
    ask nodes with [bitcoin_user > disease_threshold][set color lime]
```

```
  set mylist lput bitcoin_value mylist
  set mylist but-first mylist
  ifelse (number_total_transactions = 0) [set trans_t 0]
  [set trans_t ((number_bitcoin_transactions) /
(number_total_transactions))]

  let kok 0
  while [kok < 100][
   ask node kok [
      set bitcoin_user_list lput bitcoin_user bitcoin_user_list
      set bitcoin_user_list but-first bitcoin_user_list
      set kok (kok + 1)
    ]
  ]
  time
  tick
end
```

6.1 Trust function and DisUtility function

Now the two main functions controlling network dynamic will be presented: the *trust function* and the *disutility function*. While the concept of trust function has been deeply explained, an elucidation of *"disutility"* will now be given. First of all let us see the form of the two functions:

$$DisUtility(x) = da * \frac{1}{Trust(x)} + dc * Value(BTC)$$

$$Trust(x) = \omega_x + \alpha\, A_N(x) + \beta\, B_{BTC}(x) + \gamma\, C_{total} + \xi\, Value(BTC)$$

where

- $A_N(x)$ is the *network contribute*, in particular $A_N(x) = lnN(x)$, where *N(x)* is the number of bitcoin user neighbors of the node x;
- $B_{BTC}(x)$ is the number of transactions completed using Bitcoin by the neighbor of the node x;
- C_{total} is the total number of Bitcoin transactions divided by the number of total transactions (traditional + digital). This contribute is the same for all the nodes;

- $Value(BTC)$ is the difference between the present value of Bitcoin and the moving average of past values. This contribute is used by agents in order to predict the future behavior of the digital currency: if it is positive it means that they expect Bitcoin to gain value, if it is negative they expect it to fall;

- $da, dc, \omega_x, \alpha, \beta, \gamma$ and ξ are the control parameters and are constant. The whole dynamic depends on how these numbers are set. We will study further them in details.

Now that the form of the functions is clear let us go deeper. It is clear that the *DisUtility* depends on Trust, as stated in the Gresham Law. A remarkable aspect is that the more the Bitcoin gains value the more agents will accept it, so the ξ parameter has to be positive. On the other hand the DisUtility function is used by agents to understand what kind of money to use in every single transaction, in particular it tells them if they have to get rid of Bitcoin or if it's better to keep it. If agent has a *low* trust in the digital currency and if it is *losing* value then he is more stimulated to exchange it. So the more the DisUtility is high the more he wants to get rid of Bitcoin, because he doesn't trust it and in particular – and this is the reason why the Value term has an individual weight – he is afraid of the possibility that it will quickly lose value.

This means that the parameter da is positive and the term $\frac{1}{Trust(x)}$ is higher the more the trust function is low, so a low believing in Bitcoin results in a high DisUtility.

Moreover the parameter dc is negative, in fact if the Value contribute is positive it means that the Bitcoin has gained value and the agent wants to keep it. On the other hand if Value is negative the Bitcoin has lost value and the agent wants to get rid of it, resulting in a higher DisUtility.

What about the other parameters?

- α is positive, since the more neighbors use the digital currency the more a node believes in it;

- β and γ are positive, since an increasing number of transactions with Bitcoin results in a gain in trust in them, by Gresham Law;

- ξ is now positive instead, in fact the more the Bitcoin gains value the more an agent trusts it;
- ω_x finally is a real number. It is the initial value of the beliefs of node x. It is in general different for every node.

7 Genetic algorithms and BehaviorSearch

In order to explore the vast parameter space linked to our model we will use *genetic algorithms* methodology. In particular we will work with *BehaviorSearch*, a useful tool supported by NetLogo. We will present both and introduction to genetic algorithms and the idea behind the design of BehaviorSearch discussed in the *Miller* [38].

Genetic algorithm (GA) [10] is an adaptive heuristic search algorithm based on natural selection and genetics. Given a search space, such as the range of variation of different parameters, the algorithm uses historical information to direct the search towards the region of the search space associated to a better performance. In other words the GA solves an optimization problem exploring the parameter space.

GAs are designed in particular to solve evolutionary problems using the Darwinian principle of *the survival of the fittest*. This is done by simulating the behavior of individuals belonging to different consecutive generations. In our particular case each individual is actually a *model*. This means that we are working with a *population of models*. The evolution through generations affect some character strings of the individuals that represents chromosomes in DNA. Each character is a point in the parameter space and so it can be a possible solution to the maximization problem. The process of evolution is the following:

- the individuals of a population compete for mates and resources;
- *winners* reproduce more than *losers*;
- the genes of an offspring are the result of the combination of the genes of its parents. This means that genes of winners propagate through generations and sometimes result in an offspring with better genes than its parents;

- in this way each generation will fit better to the environment than the former.

The population is chosen inside the *search space*. Each individual represents a possible solution to the optimization problem and is represented as a finite length vector. The finite length vector, that is the solution, is called *chromosome* and each of its components it is called *gene*. Every component represents a variable regulating the individual behavior.

Each solution is then associated with a *fitness score*, that represents the ability of the individual to compete in the environment. Individuals with high fitness (winners) will now mate and produce an offspring with a chromosome composed by the genes of its parents combined. This hence results in the creation of a better performing individual. The population is always composed of n individuals, so every time a newborn joins the population another old individual will die. As a result after n mating the old generation will be replaced by a new one, with better fitness scores and hence with better genes.

The GA iterates this process until the new generation doesn't differ in a significant way with respect to the previous one. When this happens the algorithm converges and a set of solutions to the optimization problem has been found.

Let us now talk about some *implementation details*, that will come useful when setting the parameters of BehaviorSearch.

The *selection operator* regulates the way winners are chosen, in particular how the fitness is computed. Fitness can be given by an *objective* function or by a *subjective* judgment.

The *crossover operator* is the key point of the whole GA. It represents the way two individuals mate. Two individuals are chosen from the population and a random *crossover site* is chosen. This means that the vector representing the solution of the two individuals will be cut in a random point and combined in order to obtain two new strings.

For example, using binary, let the string A be 111111 and let the string B be 000000. Choose a random crossover site, for example 4. This means that the two strings will be cut at the forth character and combined together. This results in two new strings $A' = 111100$ and $B' = 000011$. These newborns are

individuals of the next generation of the population. Combining characters of winners it is likely to obtain a better performing offspring.

The *mutation operator* introduces random modifications to the offspring chromosomes, for example changing the order of some genes. In this way the diversity in a population is granted.

Each of these operators is necessary to have a good GA, in fact using only the selection operator alone will result in a population made of copies of the best individual from the former generation, using only selection and crossover operators will result in a convergence on a sub-optimal solution (still made of winner individuals), using mutation operator alone results in a random walk through the search space and finally using only selection and mutation operators creates a *parallel, noise - tolerant, hill climbing algorithm.*

7.1 Active Nonlinear Tests

BehaviorSearch was inspired by the *Miller* [38], where the author shows how a complex, large-scale computational model can be studied via a *simple, automatic, nonlinear search algorithm* called *Active Nonlinear Test* (ANT). These kind of algorithms exploit a nonlinear optimization in order to maximize the deviation between the predictions of the original model and the results obtained by perturbing it. The maximization here is done across a set of reasonable model perturbations.

First of all a maximization algorithm has to be chosen. The ANT that we want to optimize has to be in the following form (using Miller notation):

- let $M_h(p)$ be the model implications for hypothesis h and given assumptions p. Denote as $\hat{p}$ the original model assumptions;
- let Δp be the set of allowable perturbations of the model assumptions;
- define $\Pi(M_h(p), M_h(\hat{p}))$ the *reasonable* objective function, chosen in order to illustrate the model behavior under hypothesis h;
- choose an optimization algorithm in order to maximize $\Pi(M_h(p), M_h(\hat{p}))$ over $p \in \Delta p$.

The algorithm used to optimize ANT should be able to search over nonlinear objective functions while confronting noise, discontinuities and large search spaces. Two algorithms are considered: a *hill-climbing* algorithm and a genetic algorithm.

The first one works in the following way: it first chooses a random solution belonging to the state space and marks it as a temporal solution. Then, at each iteration of the algorithm, a new solution in the neighborhood of the temporal one is compared with it. If the new solution gives a higher value of the objective function than the older one it becomes the temporal solution. After a fixed number of iterations the final temporal solution is given as the optimization problem solution.

The GA has the same form shown previously.

Now that the forms of the ANT and of the optimization algorithms have been stated let us talk about the *objective function*. It has to be defined in order to give useful insights about the model behavior. It is evident that there isn't in general a fixed form for the objective function, so in order to illustrate how to implement it an example is provided. In the paper the case of *World3* developed by Meadows et al. [35] is provided. This model represents a well done example of large-scale simulation model. It can simulate things like pollution, agriculture and industrial output. It is used as an example because of its large search parameter space. Here the function to be maximized is the predicted population in 2100 and in order to do this the two algorithms are used. What is more a third algorithm (*random-search*) is implemented. It randomly generates a fixed number of solutions and uses them to compute the objective function. The best of these solutions is taken as the final one.

As a result of the optimization process Miller concludes that hill-climbing and GA significantly outperform random search. In particular *in this example* the hill-climbing algorithm performs better than GA. This is not in general true, but depends on the form of the problem. Last it is observed that increasing iterations increases performances *but at a decreasing rate*, that is something we should expect from such an optimization problem.

The design of BehaviorSearch was inspired by this paper, in fact the model exploration follows these four steps:

- choose the search space;
- define an objective function;
- select an optimization algorithm;
- examine the results.

The selection of the search space has to be done as a first thing. BehaviorSearch allows to select the parameters, set the range and the step of variation. The set of parameters with their range constitutes the search space. In our model for example every parameter regulating the Trust and the DisUtility value are included in the search space. Moreover their thresholds are included in this space, in order to find their maximum value that allows Bitcoin spread.

Now the crucial decision has to be taken. This is the choice of the objective function, that represents some sort of *behavioral measure*. This is nothing but a quantitative measure that mimics some type of behavior of the system. In our model case we want to find the critical point that separates the scenario in which the digital currency is accepted from the one in which it is not. This means that we want to find either the maximum fitness (combination of parameters) under which the digital currency is not trusted or the minimum value that assures the diffusion of the digital money. This finally leads to a solution vector which components are critical values for the parameter set, meaning that a slight change in some parameter value can drastically change the dynamic behavior.

Now that the objective function has been discussed and implemented the search algorithm has to be chosen. In BehaviorSearch, like in Miller paper, three different algorithms can be chosen: the random search, the hill-climbing and the genetic algorithm. As discussed in the paper the random search is outperformed by the other two. In our model case a GA will be used.

Last the results are saved in an output file that will be analyzed using R. This will give useful insights not only on the critical numerical value of the parameter set, but also on the weight of each parameter related to dynamic behavior.

7.2 Holland's Schema Theorem

In this section we will discuss the power of GA, or the effectiveness of the GA search process. In particular we will prove the *Holland's Schema Theorem*, that states that *every genetic algorithm converges*.

Let us begin with some definitions. First of all a *schema H* is a template that identifies a subset of strings with some patterns or similarities at certain string positions. For example, using binary code for simplicity, the schema H = [* 1 *] identifies the chromosome set

0 1 0

1 1 0

0 1 1

1 1 1

The *schema order o(H)* is the number of genes in the schema H not denoted with *. For example, in the previous example the order of H = [* 1 *] is $o(H) = 1$, in fact we have only one gene not labeled with *.

Lastly the *schema defining length* $\delta(H)$ is the distance between the first and the last gene in schema H not denoted with *. The example of H = [* 1 *] is quite trivial, since the first and the last "not *" genes coincide, so $\delta(H) = 1 - 1 = 0$.

Now we can start discussing the problem. Let us start with the selection operator with fitness proportional selection. What we are trying to model is the probability that an individual h samples the schema H, or mathematically $P(h \in H)$.

Let *m(H,t)* be the number of individuals matching the schema H at generation t, $f(H,t)$ be the mean fitness of individuals matching H, M the population size and $\overline{f}(t)$ the mean fitness of individuals in the entire population. So we can express the probability as

$$P(h \in H) = \frac{m(H,t)\, f(H,t)}{M\, \overline{f}(t)}$$

Under fitness proportional selection the expected number of instances of H at $t+1$ is

$$E[m(H,t+1)] = M \times P(h \in H) = \frac{m(H,t)\, f(H,t)}{\overline{f}(t)}$$

This means that schema with fitness greater than the average have more chances to proportionally account for more of the population at $t+1$.

As we know the second operator to take in account is the crossover operator. In particular we analyze the problem using a single point crossover operator, as discussed previously. Let us consider this example: take two schema

$$H_1 = [\ 1\ 1\ *\ *\ *\ 1\ ]$$
$$H_2 = [\ 1\ 1\ *\ *\ *\ *\ ]$$

and take the crossover point at the second position of each schema. This results in H_1 being broken by this choice of crossover point, while H_2 is not compromised and remains unaffected. So we can state that *schema with longer defining length are more likely to be broken by single point crossover than schema with shorter defining length.* Formally we get that the probability that a schema H will not survive under a single point crossover at generation t is:

$$p_c \frac{\delta(H)}{l-1} P_{diff}(H,t)$$

where l is the length of the string, p_c is the *a priori* chosen threshold of applying crossover and $P_{diff}(H,t)$ is the probability that the "second parent" is not able to repair the broken genes of schema H, that are the ones excluded by single point crossover. In the worst case, that we label as *lower bound*, $P_{diff}(H,t) = 1$, that means that the second parent is never able to repair H.

The third and final step is the mutation. In this case it will be applied the *gene wise mutation*, where mutation is applied gene by gene. Let p_m be the probability that a gene mutates, naturally $p_s = 1 - p_m$ will be the probability that a gene doesn't mutate. We are interested in such a probability because a schema H survives when all his "non *" genes remain unchanged. This can be written as

$$p_s^{o(H)}$$

Since typically $p_m << 1$ we can write

$$p_s^{o(H)} = (1 - p_m)^{o(H)} \approx 1 - o(H)\, p_m$$

This represents the lower bound probability that a schema H of order $o(H)$ will survive at t under gene wise mutation.

We can now state the *Schema Theorem*: the expected number of schema at $t+1$ when using a Genetic Algorithm with proportional selection, single point crossover and gene wise mutation is:

$$E\left[m(H,t+1)\right] \;\geq\; \frac{m(H,t)\, f(H.t)}{\overline{f}(t)} \left[1 - p_c \frac{\delta(H)}{1-l}\, p_{diff}(H,t) - o(H)\, p_m\right]$$

A needed specification is that this holds for a population with infinite members. In the finite case results may be different. What is more this formula is strictly linked to the type of GA we selected. In general let $\alpha(H,t)$ be the selection coefficient and $\beta(H,t)$ the transcription error. Then the Schema Theorem assumes the form

$$E\left[m(H,t+1)\right] \;\geq\; m(H,t)\, \alpha(H,t) \left[1 - \beta(H,t)\right]$$

This allows us to write formally when a schema H survives:

$$\alpha(H,t) \;\geq\; 1 - \beta(H,t)$$

or

$$\frac{f(H,t)}{\overline{f}(t)} \;\geq\; \left[1 - p_c \frac{\delta(H)}{1-l}\, p_{diff}(H,t) - o(H)\, p_m\right]$$

As a result short defining length and low order schema of above average population fitness will be favored by "canonical" GAs, that are in the form that we discussed so far.

7.3 Configuration of Behaviorsearch

In order to find critical sets of points we have to investigate borderline system dynamics, where the small modification of a crucial parameter can influence the diffusion of digital money. We fix a certain number of maximum model cycles, say 2000 in our example. What is more we fix the rate $\frac{<k>}{<k^2>}$ in order to always work with the same type of network. We then focus on the number of Bitcoin users. When the number of Bitcoin users is equal to the number of agents the cryptocurrency has diffused and the dynamic can stop. The genetic algorithm has to maximize the time required to transform all agents in Bitcoin users. In this way all the trivial cases in which diffusion is extremely

fast or extremely slow are cut out. In particular the first cases represent local maximum points and the second cases represent local minimum points. We are interested in critical points that stand between these two types of objects. Concluding the sets of critical points and the variation of each parameter are empirically studied on the model. Points found by genetic algorithms behave like critical points and a small variation can lead to a different system dynamic.

An example of a genetic algorithms search is shown in the following picture.

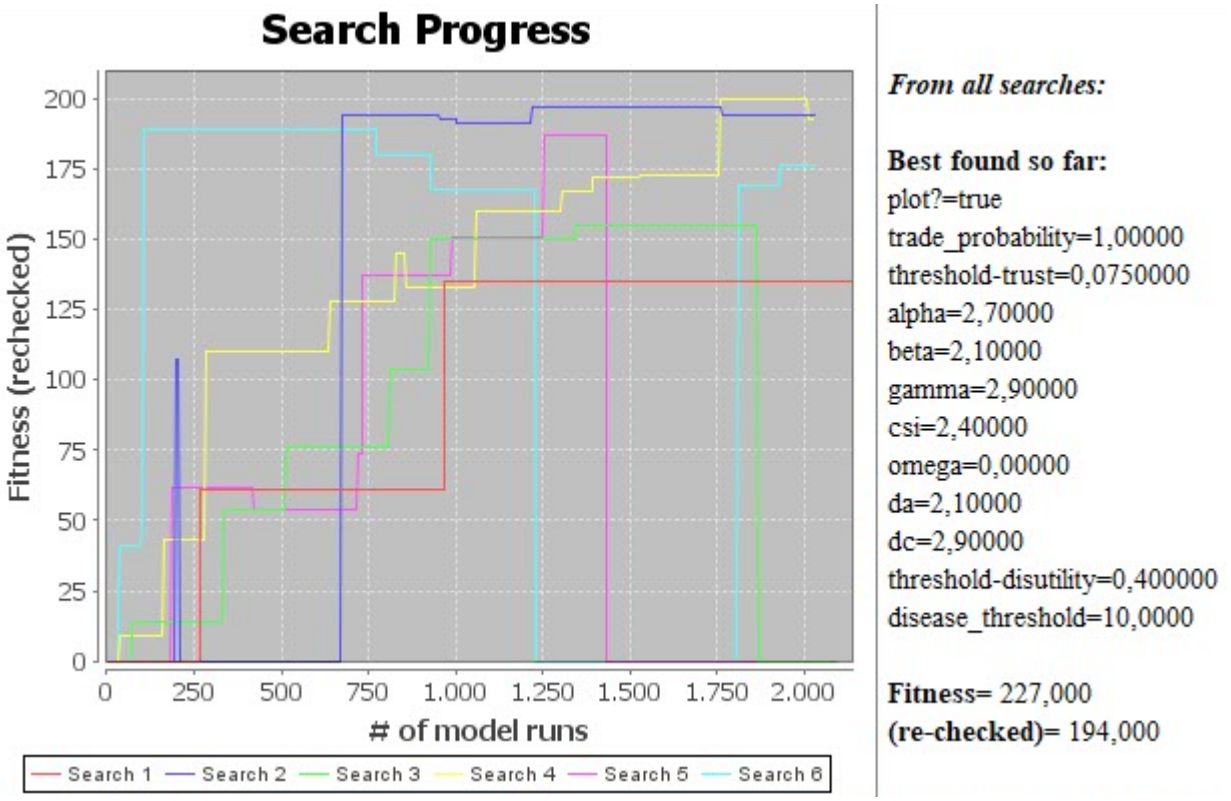

Figure 3: An example of a successful genetic algorithm search.

Here we used a Standard Genetic Algorithm with:

$$mutation - rate = 0.01,$$

$$population - size = 50,$$

$$tournament - size = 3,$$

$$population - model = generational,$$

$$crossover - rate = 0.7,$$

$$ChromosomeRepresentationType = GrayBinaryChromosome,$$

$$BestCheckingNumberOfReplicates = 10.$$

8 Single layer Analysis

Genetic algorithms produced a set of critical points for the system dynamic. These points represent a threshold for the diffusion of cryptocurrency among agents: modifying one of the parameters value results in a crucial alteration of the dynamic of the system. Mathematically, the second derivative of the 8-dimensional hyper-surface (the parameter space) is zero in the critical points found by genetic algorithms. This property is useful in order to find both the number of maximums of the hyper-surface and where these maximums are.

In order to investigate the hyper-surface 28 graphs were produced. In z axis we always mapped the fitness attached to the set of parameters, in x and y axis we put every possible couple of different parameters. In this way studying resulting graphs is it possible to detect some clusters.

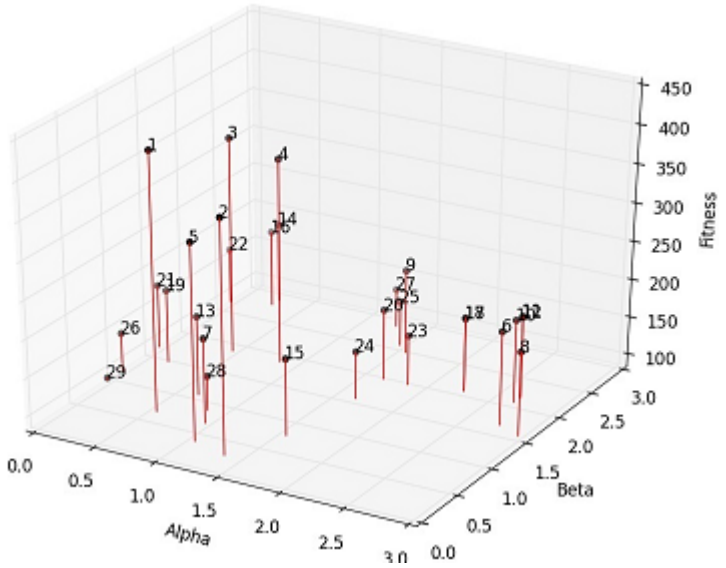

Figure 4: alpha - beta graph.

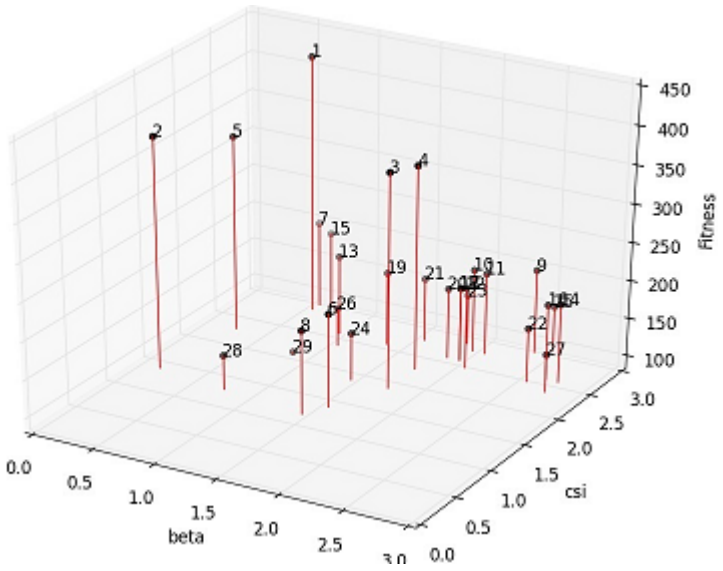

Figure 5: beta - csi graph.

The points of each cluster of each graph are now studied numerically. In order to allow two points to be part of the same cluster at least four of their

parameters have to be *compatible*. Two parameter values are compatible if their values differ at most for 10% of the allowed range of values for that parameter. In our case every parameter range is $[0, 3]$, so the maximum difference can be 0.3 .

Let us make an example in order to clarify: assume we have two four-dimensional points A and B. Each dimension represents a parameter. Points so are in the form $X = \{x, y, z, t\}$. Let

$$A = \{0.4, 1.8, 2.5, 2.1\} \qquad B = \{0.5, 3.0, 2.2, 0.1\}$$

In order to see if x_A and x_B are compatible we do the following:

$$|x_A - x_B| = |0.4 - 0.5| = 0.1 \leq 0.3$$

So in this case the two parameters are compatible.
Let us do this now with y_A and y_B:

$$|y_A - y_B| = |1.8 - 3.0| = 1.2 \geq 0.3$$

In this case the parameters are not compatible.

In general this is not the only clustering condition for this kind of problem, in fact a cluster can have *sub-clusters*. This will be explained using a graphical representation. In the following picture the two flex points are part of the same maximum (cluster), but have in general different x and y values. This kind of problem will be studied using the first derivative of the hyper-surface relative to those points. If the variation of non compatible parameters will result in a gain in fitness then the two points are part of the same cluster. This analysis will be done empirically on the model.

8.1 Orange Cluster

In the *Orange Cluster* the parameters ξ, dc and $threshold - disutility$ are compatible:

$$\xi_{orange} \in [2.4, 2.7], \qquad dc_{orange} \in [2.9, 3],$$
$$threshold - disutility_{orange} \in [0.4, 0.6]$$

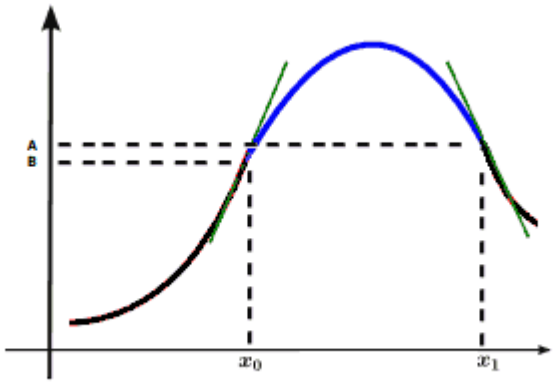

Figure 6: Flex points x_0 and x_1 have in general different values for y.

Studying the first derivative it can be seen that the parameters α and da are compatible too. This means that in order to find a point in the Orange Cluster we have to fix the parameters in the following ranges:

$$\xi_{orange} \in [2.4, 2.7],$$
$$dc_{orange} \in [2.9, 3],$$
$$threshold - disutility_{orange} \in [0.4, 0.6],$$
$$\alpha_{orange} = 2.7 \ \ or \in [0, 0.7],$$
$$da_{orange} \in [2, 2.1] \ \ or \ \ [0.5, 1.1]$$

We can then study the first derivative of the remaining parameters: β, γ and $threshold - trust$. This is done empirically on the model, modifying the parameters and looking at fitness. If an increase of parameter value results in a gain in fitness then the derivative is positive, on the contrary if this results in a loss in fitness then the derivative is negative.

β_{orange} derivative is positive, in fact high betas are followed by high values of fitness. On the contrary, γ_{orange} derivative is negative. $Threshold - trust$ derivative is positive as expected, in fact the higher the threshold the less

fast the diffusion of cryptocurrency will be and the greater the fitness will be.

fitness	alpha	beta	gamma	csi	da	dc	threshold-trust	threshold-disutility
198.5	2.7	2	2.9	2.7	2.1	2.9	0.125	0.4
197	2.7	2.1	2.9	2.7	2.1	2.9	0.075	0.4
196	2.7	2.1	2.9	2.4	2.1	2.9	0.075	0.4
193.5	0.7	1	0.6	2.5	0.5	2.9	0.125	0.4
193.5	0.3	2.8	1.3	2.5	1.1	3	0.075	0.4
189	0.3	2.7	1.2	2.5	1.1	3	0.075	0.4
185.5	0.2	1.4	1.6	2.5	1.1	3	0.025	0.4
172.5	0	1.6	0.7	2.7	2	3	0.025	0.6
mean	1.20	1.96	1.76	2.56	1.51	2.95	0.08	0.43
std dev	1.26	0.62	0.99	0.12	0.63	0.05	0.04	0.07

Table 3: Orange Cluster points.

8.2 Violet Cluster

In the *Violet Cluster* the following parameters are compatible in the following ranges:

$$\beta_{violet} \in [1.9, 2.0],$$
$$\xi_{violet} = 2.5,$$
$$dc_{violet} \in [1.1, 1.4],$$
$$threshold - disutility = 0.2,$$

$Threshold-trust$ derivative is positive as expected. Values of fitness are high for α near both to 3 and to 0. This means that first derivative respect to α is positive around the borders of its range and negative in the interval. This results in a maximum less smooth than the one in the previous case. γ shows a regular behavior, as its derivative is positive. Last, da derivative is negative.

fitness	alpha	beta	gamma	csi	da	dc	threshold-trust	threshold-disutility
187	2.3	2.2	2.3	2.5	1.5	1.1	0.025	0.2
186	2.3	2	2.3	2.5	1.5	1.1	0.025	0.2
182	1.7	1.9	1	2.5	0.6	1.4	0.075	0.2
mean	2.10	1.97	1.87	2.50	1.20	1.20	0.04	0.20
std dev	0.35	0.06	0.75	0.00	0.52	0.17	0.03	0.00

Table 5: Violet Cluster points.

8.3 Other Points

In general we expect the 8-dimensional hyper-surface we are working with not to be particularly smooth. Data produced by genetic algorithms highlighted two local maximums in the form of clusters, but many more points showing weaker compatibility were found. This analysis concludes that two local maximums can be found in the neighborhoods of *Orange Cluster* and *Violet Cluster*. Six potential smaller local maximums *could* be in the neighborhoods of the six weaker clusters. Data regarding the small six clusters are reported in the following table.

Blue Cluster								
fitness	alpha	beta	gamma	xi	da	dc	threshold-trust	threshold-disutility
425	0.6	0.6	2.4	2.8	2.3	0.5	0.1	0.2
155	1.9	1.9	2.2	2.8	2	0.8	0.125	0.2
mean	1.25	1.25	2.3	2.8	2.15	0.65	0.1125	0.2
std dev	0.92	0.92	0.14	0.00	0.21	0.21	0.02	0

Grey Cluster								
fitness	alpha	beta	gamma	csi	da	dc	threshold-trust	threshold-disutility
392	1.4	0.2	2	1.3	0.5	1	0.075	0.1
212.5	2.8	1.6	2.1	1.3	1	1.2	0.075	0.1
200	3	1.5	2.6	1.1	0.1	2.1	0.05	0.2
mean	2.40	1.10	2.23	1.23	0.53	1.43	0.07	0.13
std dev	0.87	0.78	0.32	0.12	0.45	0.59	0.01	0.06

Red Cluster								
fitness	alpha	beta	gamma	csi	da	dc	threshold-trust	threshold-disutility
201	1	0.6	1.1	2.9	3	2.7	0.075	0.7
135	0.9	0.8	1	1.2	2.8	1.9	0.025	0.5
mean	0.95	0.70	1.05	2.05	2.90	2.30	0.05	0.60
std dev	0.07	0.14	0.07	1.20	0.14	0.57	0.04	0.14

Brown Cluster								
fitness	alpha	beta	gamma	csi	da	dc	threshold-trust	threshold-disutility
200	1.6	2.4	1.4	2.9	1.1	2	0.025	0.3
190.5	1.6	0.7	2.3	2.9	0.9	2.4	0.025	0.3
152	1.7	1.5	2.2	1.8	2.8	2.5	0.05	0.4
148.5	1.5	2.5	0.8	3	3	0	0.175	0.2
mean	1.60	1.78	1.68	2.65	1.95	1.73	0.07	0.30
std dev	0.08	0.85	0.71	0.57	1.10	1.17	0.07	0.08

Yellow Cluster								
fitness	alpha	beta	gamma	csi	da	dc	threshold-trust	threshold-disutility
345.5	1.1	0.3	2.3	2.2	1.8	0.6	0.025	0.2
139.5	1.3	2.8	2.6	2.3	2.5	1.1	0.025	0.2
mean	1.20	1.55	2.45	2.25	2.15	0.85	0.025	0.2
std dev	0.14	1.77	0.21	0.07	0.49	0.35	0.00	0.00

Green Cluster								
fitness	alpha	beta	gamma	csi	da	dc	threshold-trust	threshold-disutility
370.5	0.5	1.8	2.8	1.8	1.5	0.3	0.1	0.1
356.5	0.9	1.8	0	2.2	1.3	2.1	0.125	0.5
160.5	0	2.6	2.9	2.4	2.9	0.8	0.125	0.2
139.5	0	1.1	0.2	2.3	2.3	2.1	0.125	0.8
92.5	0	0.9	0.2	2	2.3	2.8	0.075	1
mean	0.28	1.64	1.22	2.14	2.06	1.62	0.11	0.52
std dev	0.41	0.67	1.49	0.24	0.65	1.03	0.02	0.38

Table 12: Other Clusters points.

8.4 Interpretation

Let us look at similarities between the Orange and the Violet Cluster. First of all we can highlight that $\xi_{orange} \in [2.4, 2.7]$ and $\xi_{violet} = 2.5$ are comparable in the chosen ± 3 interval. What is more, fitness for critical points attains its maximum value for $\alpha_{orange} \in [0, 0.7]$ or for $\alpha_{orange} = 2.7$ while fitness for Violet Cluster attains its maximum for α_{violet} near 0 and near 3.

In our model ξ represents the weight that agents give to the difference between the present value of Bitcoin and the moving average of past values,

that represents agents memory. We can then conclude that in order to have a maximum for fitness, and hence in order to have *diffusion of cryptocurrency*, agents have to care about value fluctuations of Bitcoin respect to fiat currency. In the model the more the Bitcoin gains value the more agents trust it. Being the value of weight associated to this factor so high we can say that the gain in value of cryptocurrency is a necessary condition for its spread.

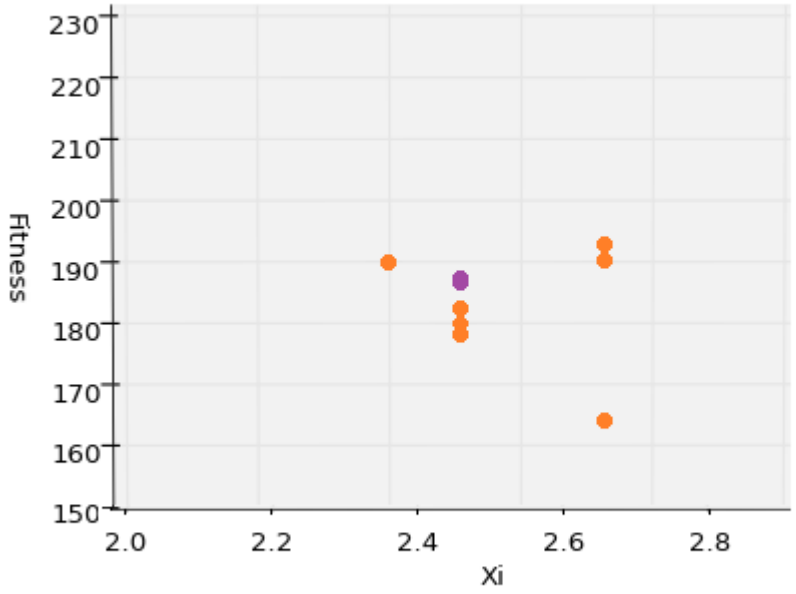

Figure 7: ξ_{Orange} values and ξ_{Violet} values comparison

Comparing theory to *data* [3] we can see that in the last year the number of Bitcoin users grew from 9 millions in October 2016 to more than 17 millions in October 2017. An interesting fact is that in general Bitcoin value grew from circa $600 in October 2016 to circa $3700 in October 2017, but we saw an high peak in the early days of September 2017 where its value reached $4780 .

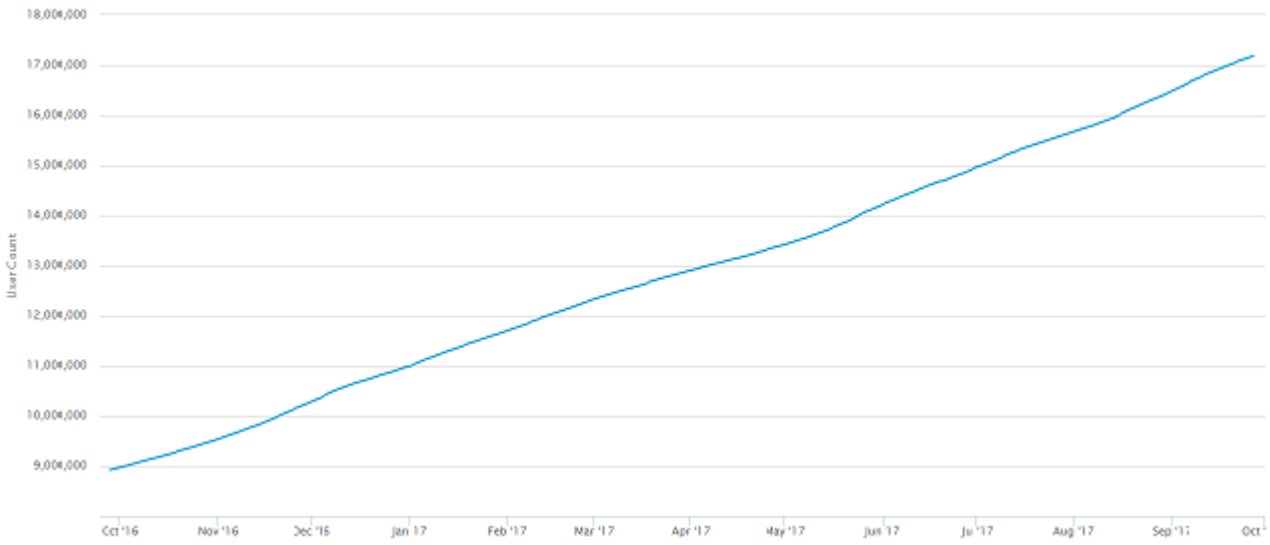

Figure 8: Number of Bitcoin users from October 2016 to October 2017.

Despite the fall in price, Bitcoin users growth remained stable during the whole month of September. We can then conclude that value growth is crucial in order to affect digital money diffusion and that temporary shocks to price values do not affect the process significantly. Note that in the model we simulated the Bitcoin price dynamic as a random walk. In this way we avoided extreme situations where value can significantly grow or significantly fall.

Parameter α represents the weight that agents give to the logarithm of the number of their Bitcoin users neighbors. We should expect that a high number of neighbors using Bitcoin is associated to a higher agent trust. This is because agent is confident that he will be able to spend that kind of money because people interacting with him will accept it. Simulations show that cryptocurrency diffusion is associated to high values of α as expected.

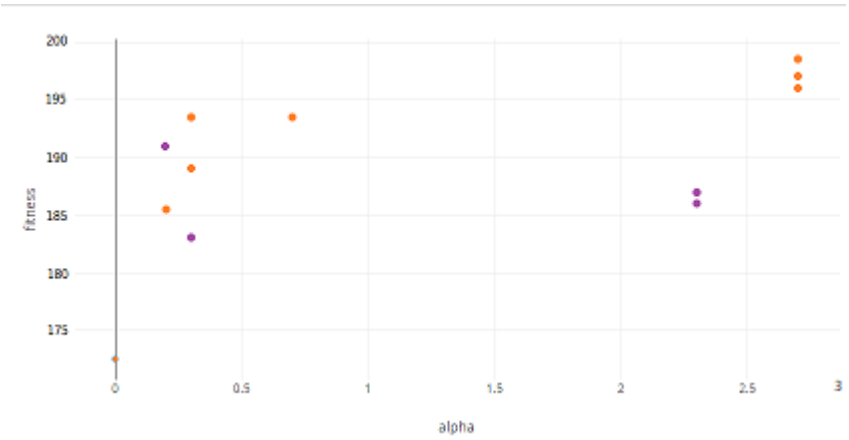

Figure 9: α_{Orange} values and α_{Violet} values comparison.

Surprisingly simulations show that we can have diffusion for low values of α too. Low values of α are followed by values of γ, ξ and da in the interval $[1, 3]$. The process of diffusion associated to this kind of parameter set can be caused by a pure speculative use of the cryptocurrency. In fact agents are not looking at their neighbors and at their possibilities to sped Bitcoin, but at its value growth, at the number of total transactions and at DisUtility. The *Green Cluster* is a perfect example of this behavior.

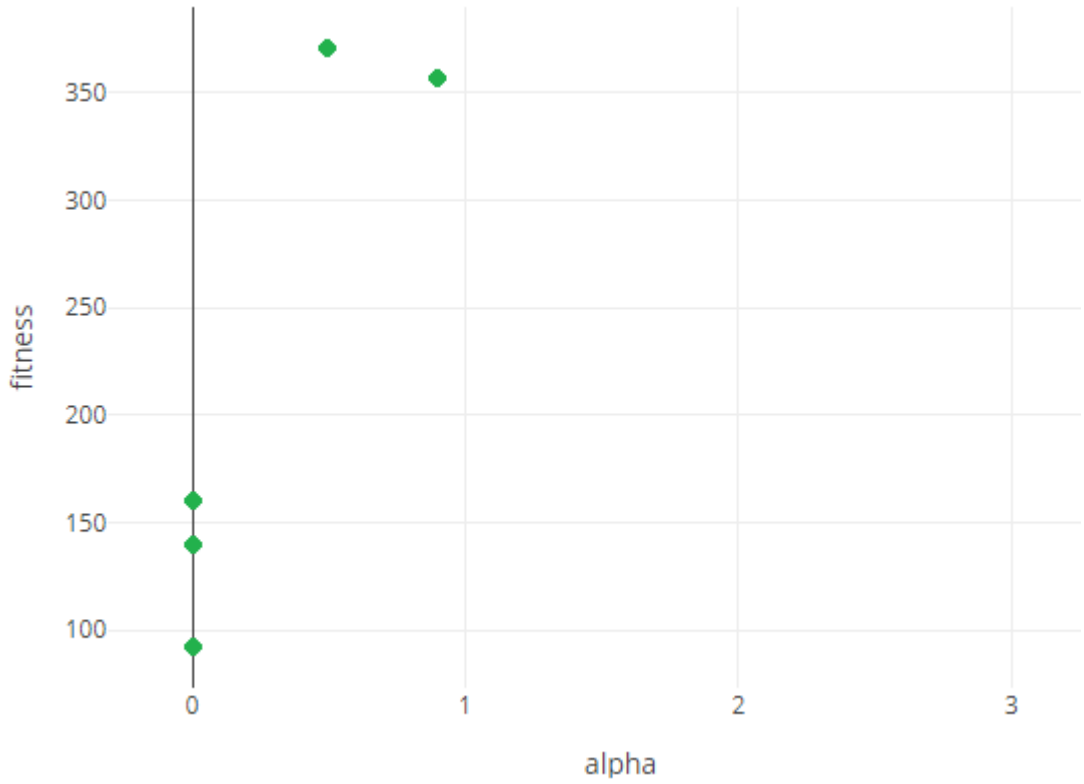

Figure 10: α_{Green} values. It is interesting to notice that cryptocurrency spread is still present for low values of α too.

9 Multilayer Analysis

As a further develop a *multilayer network* is implemented. Nodes represent wallets and a layer of *owners* is added. In our model the number of owners is the 30% of the number of nodes and wallets are distributed to them following a Poisson distribution. The main difference with respect to single layer model is that here trust and disutility are not properties of the wallets, but of the owners. In this way trust can emerge in different parts of network, that is constructed using the preferential attachment algorithm as usual, without any particular reason. This is due to the fact that anonymity is granted, in fact owners are hidden and it is not possible for other individuals to know the ownership of one wallet.

What is more nodes can participate to two different networks. One is a legal one and one is an illegal one. The entire number of nodes participate of the legal layer while a smaller part of them is present on the illegal one. The transaction process is the same as the single layer one, but now nodes can choose if they want to interact with a node in the legal layer or in the illegal one. Only legal layer transactions are public.

Last, sometimes an illegal node can propose to a legal one to conclude an illegal transaction, and hence to participate to the illegal layer. This kind of transaction will be negated and as a consequence the trust value of the owner will decrease (if he has no illegal wallets).

Parameter space is investigated using genetic algorithms as the single layer case.

9.1 Results

Cluster analysis has been done for multilayer case too. It is interesting to notice that only one cluster is identified and hence only one local maximum is recognized. What is more the iteration of the genetic algorithms gave as a result the same type of solutions. As a result we don't have a large variety of critical sets as in the single layer model, but a group of solutions near to the same cluster. Data regarding the maximum are reported in the following table.

Purple Cluster								
fitness	alpha	beta	gamma	csi	da	dc	threshold-trust	threshold-disutility
180	2.5	1.2	2.2	3	2.4	1.1	0.075	0.2
168.5	1.5	1	2.7	2.7	2.6	2.1	0.125	0.3
153	2.6	1.4	2.6	2.4	1.4	1.6	0.075	0.2
144.5	1.3	2.8	1.7	3	2.9	2.1	0.075	0.3
mean	1.98	1.6	2.3	2.78	2.33	1.73	0.09	0.25
std dev	0.67	0.82	0.45	0.29	0.65	0.48	0.03	0.06

Looking at data it is possible to notice that multilayer fitness for critical

points is in general lower than fitness for single layer critical sets, for example the maximum value in the first case is 180 while in the latter 425. This implies that in general the emergence of the trust in cryptocurrency is faster in multilayer networks than in single layer ones. This is motivated by the fact that owners share the trust and the disutility among their wallets. It is then sufficient to observe the diffusion of the digital money in an area of the network to seriously influence other detached areas.

What is more it is possible to notice that critical points belonging to identified local maximum have the following properties:

$$\alpha \in [1.3, 2.6]$$
$$\gamma \in [1.7, 2.7]$$
$$\xi \in [2.4, 3]$$
$$da \in [1.4, 2.9]$$

This means that diffusion of cryptocurrency is heavily influenced by these parameters. A high value of α means that owners require a high number of neighbors using digital money in order to trust it. What is more a high value of γ represents that they are influenced by the number of digital money transactions made in the entire legal network too. This is probably due to the fact that owners have different presences in the network and hence they prefer to look at the entirety of it instead of looking at their neighbors. It is important to stress that they actually do not know who their neighbors are, since they only interact with wallets and not with other real individuals.

Similarly to the single layer case agents trust is sensible to Bitcoin value volatility, in fact ξ assumes large values. Last, agents' disutility is highly affected to the inverse of trust. In this way we notice the *emergence of the trust induced by the Gresham Law*, in fact individuals want to get rid of the digital money because they do not trust it (da is high), leading to a trust in the same digital money given by the fact that many nodes use it (in fact α is high) and consequently many cryptocurrency transactions are concluded (in fact γ is high).

To conclude, the speculative emergence of trust observed in the single layer case is not present here.

10 Conclusion

Two different models were implemented in order to study the diffusion of a cryptocurrency: a single layer and a multilayer one. The space of parameters was investigated with genetic algorithms in order to find a set of critical points for the dynamic of the systems.

In the single layer model two major local maximums were identified using cluster analysis. What is more a variety of other solution sets were identified. Maximums show that diffusion can happen under two possible conditions: in the first agents care much about the presence of cryptocurrency in their neighborhood as well as its value fluctuations. Whereas in the second case agents are only interested in digital money value and hence in the possibility of personal gains. This is then a speculative regime.

Multilayer model on the other hand mimics the Gresham Law, in fact it shows that owners of wallets tend not to trust cryptocurrency and want to get rid of it. What is more, diffusion is possible when many agents neighbors adopt digital money and when there is a high number of total transactions. As a consequence trust in digital money emerges due to the fact that individuals do not trust it and tend to exchange it, increasing the number of digital money transactions and the number of digital money users.

10.1 Further developments

In this model agents trust is initially set at zero. This is not in general true and we could implement some sort of *prior thoughts* on cryptocurrency. This is however coded in the model under the name of ω but is not considered in the analysis.

What is more multilayer model is now static in the sense that wallets can not change ownership and owners can not create new wallets. A wallet dynamic is more realistic and would affect diffusion. Moreover illegal network could be modified in order to get more realistic, for example illegal nodes could be sometimes detected and excluded by the network as well as all the other owner wallets.

References

[1] Bitcoin chart volatility. https://data.bitcoinity.org/markets/volatility/2y/EUR?c=e&f=m10&g=15&st=log&t=l [Online; accessed 5-October-2017].

[2] Bitcoin price chart history. https://99bitcoins.com/price-chart-history/ [Online; accessed 29-September-2017].

[3] Bitcoin users chart history. https://blockchain.info/charts/my-wallet-n-users?timespan=1year [Online; accessed 29-September-2017].

[4] Bytecoin. http://bytecoin.org/ [Online; accessed 22-August-2017].

[5] Cryptographic hash function. https://en.wikipedia.org/wiki/Cryptographic_hash_function [Online; accessed 9-May-2017].

[6] Cryptonote. http://cryptonote.org/ [Online; accessed 22-August-2017].

[7] Dogecoin. http://en.wikipedia.org/wiki/Dogecoin [Online; accessed 22-August-2017].

[8] Ethereum. https://www.ethereum.org/ [Online; accessed 22-August-2017].

[9] Frecoin. http://en.wikipedia.org/wiki/Frecoin [Online; accessed 22-August-2017].

[10] Genetic algorithms. *Surprise 96, Department of Computing, Imperial College of London*. https://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol1/hmw/article1.html#top [Online; accessed 30-July-2017].

[11] Gresham's law. https://en.wikipedia.org/wiki/Gresham%27s_law [Online; accessed 9-May-2017].

[12] Ixcoin. http://www.ixcoin.co/ [Online; accessed 22-August-2017].

[13] Litecoin. http://en.wikipedia.org/wiki/Litecoin [Online; accessed 22-August-2017].

[14] Namecoin. http://en.wikipedia.org/wiki/Namecoin [Online; accessed 22-August-2017].

[15] Proof of work. https://en.bitcoin.it/wiki/Proof_of_work [Online; accessed 9-May-2017].

[16] Sha-256. https://en.bitcoin.it/wiki/SHA-256 [Online; accessed 9-May-2017].

[17] Target. https://en.bitcoin.it/wiki/Target [Online; accessed 9-May-2017].

[18] Zerocoin. http://zerocoin.org/ [Online; accessed 22-August-2017].

[19] Bitcoin's boom: digital currencies. *The Economist*, 26 May 2017. https://espresso.economist.com/4afe044911ed2c247005912512ace23b [Online; accessed 22-August-2017].

[20] Richardson M. A. Madhavan and M. Roomans. Why do security prices change? a transaction-level analysis of nyse stocks. *The Review of Financial Studies, volume 10*, 1997.

[21] Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Reviews of modern physics*, 74, 2002.

[22] Marcella Atzori. Blockchain-based architectures for the internet of things: A survey. 2016.

[23] European Bank Authority. European bank authority opinion on bitcoin. 2014.

[24] Albert-László Barabási. Network science. http://barabasi.com/networksciencebook/chapter/5 [Online; accessed 3-May-2017].

[25] Rainer Bohme, Nicolas Christin, Benjamin Edelman, and Tyler Moore. Bitcoin: Economics, technology, and governance. *Journal of Economic Perspectives*, 2015.

[26] btctheory.com. Greshams law and bitcoin. https://btctheory.com/2014/08/07/greshams-law-and-bitcoin/ [Online; accessed 9-May-2017].

[27] Stefano Capaccioli. Criptovalute e bitcoin: un'analisi giuridica. 2015.

[28] D. Chaum. Blind signatures for untraceable payments. *Advances in Cryptology: Proceedings Of Crypto*, 1982.

[29] S. A. Corwin and P. Schultz. A simple way to estimate bid-ask spread from daily high and low prices. *The Journal of Finance, volume 67*, 2012.

[30] Verma S Crosby M, Nachiappan Pattanayak P and Kalyanaraman V. Blockchain technology: Beyond bitcoin. *Appl Innov Rev 2:619*, 2016.

[31] Thomas Dimpfl. Bitcoin market microstructure. 2017. `https://ssm.com/abstract=2949807` [Online; accessed 21-August-2017].

[32] Gautham. Japan officially recognises bitcoin as currency starting april 2017. *NewsBTC*, 2017. `http://www.newsbtc.com/2017/04/02/japan-officially-recognises-bitcoin-currency-starting-april-2017/` [Online; accessed 22-August-2017].

[33] L. R. Glosten and L. E. Harris. Estimating the components of the bid/ask spread. *Journal of Financial Economics, volume 21*, 1998.

[34] R. D. Huang and H. R. Stoll. The components of the bid-ask spread: A general approach. *The Review of Financial Studies, volume 10*, 1997.

[35] Meadows D. L., Behrens III W. W., Meadows D. H., Naill R. F., Randers J., and Zahn E. K. O. Dynamics of growth in a finite world. *Wright-Allen Press, Cambridge, MA*, 1974.

[36] William J Luther. Cryptocurrencies, network effects, and switching costs. *Contemporary Economic Policy*, 2015.

[37] H. Mendelson. Market behavior in a clearing house. *Econometrica, volume 50*, 1982.

[38] John H. Miller. Active nonlinear tests (ants) of complex simulation models. *Management Science, Vol 44, No. 6*, 1998.

[39] Michael Nielsen. How the bitcoin protocol actually works. `http://www.michaelnielsen.org/ddi/how-the-bitcoin-protocol-actually-works/` [Online; accessed 9-May-2017].

[40] Global Drug Policy Observatory. Silk road and bitcoin. https://www.swansea.ac.uk/media/GDPO%20Situation%20Analysis%20silk%20rd%20and%20bitcoin.pdf [Online; accessed 9-May-2017].

[41] John Prpić. Unpacking blockchains. 2017. Collective Intelligence 2017.

[42] Zulfikar Ramzan. Bitcoin: Transaction block chains. https://www.khanacademy.org/economics-finance-domain/core-finance/money-and-banking/bitcoin/v/bitcoin-transaction-block-chains [Online; accessed 9-May-2017].

[43] R. Roll. A simple implicit measure of the effective bid-ask spread in an efficient market. *Journal of Finance, volume 37*, 1984.

[44] Groth J. C. S. K. Cooper and W.E. Avera. Liquidity, exchange listing, and common stock performance. *Journal of Economics and Business, volume 37*, 1985.

[45] Lisa Scherzer. What you might not know about the $100 bill. https://finance.yahoo.com/blogs/the-exchange/might-not-know-100-bill-235314234.html [Online; accessed 9-May-2017].

[46] Fabian B. T. Ermakova, Baumann A., Izmailov M., and H. Krasnova. Bitcoin: Drivers and impediments. 2017.

[47] Blockchain Technologies. Smart contracts explained. www.blockchaintechnologies.com/blockchain-smart-contracts [Online; accessed 22-August-2017].

[48] Wilensky U. Netlogo. 1999. https://ccl.northwestern.edu/netlogo/ [Online; accessed 5-October-2017].

[49] Wilensky U. Netlogo preferential attachment model. 2005.

[50] J. Warren. A peer-to-peer message authentication and delivery system. 2012. https://bitmessage.org/bitmessage.pdf [Online; accessed 22-August-2017].

Ringraziamenti

Vorrei innanzitutto ringraziare il mio relatore, il Professor Pietro Terna, per avermi seguito durante l'elaborazione di questa tesi. La sua grande competenza e disponibilitá sono state fondamentali per la riuscita di questo lavoro. Vorrei ringraziare inoltre il mio controrelatore, il Professor Marco Maggiora, per gli interessanti spunti proposti.

Coloro a cui devo di piú, e che maggiormente ringrazio, sono i miei genitori: a mio padre, che mi ha trasmesso l'amore per la conoscenza e il cui esempio mi ha condotto verso studi scientifici e a mia madre, eterno esempio e punto di riferimento della mia vita.

Un ringraziamento va anche ai miei compagni di corso, con cui in questi due anni ho potuto confrontarmi, divertirmi e stringere solidi legami.

Vorrei ringraziare i miei amici di sempre, con cui ho vissuto le piú improbabili esperienze, cosapevole del fatto che potremo sempre contare sul reciproco sostegno e senza cui gli anni universitari sarebbero stati decisamente peggiori.

Infine un particolare ringraziamento va alla mia ragazza, Gaia, a cui questa tesi é dedicata, conscio di quanto io sia fortunato ad avere nella mia vita una persona cosí straordinaria.