Blockchain: A Graph Primer

CUNEYT GURCAN AKCORA, University of Texas at Dallas
YULIA R. GEL, University of Texas at Dallas
MURAT KANTARCIOGLU, University of Texas at Dallas

Bitcoin and its underlying technology Blockchain have become popular in recent years. Designed to facilitate a secure distributed platform without central authorities, Blockchain is heralded as a paradigm that will be as powerful as Big Data, Cloud Computing and Machine learning.

Blockchain incorporates novel ideas from various fields such as public key encryption and distributed systems. As such, a reader often comes across resources that explain the Blockchain technology from a certain perspective only, leaving the reader with more questions than before.

We will offer a holistic view on Blockchain. Starting with its brief history, we will give the building blocks of Blockchain, and explain their interactions. As graph mining has become a major part its analysis, we will elaborate on graph theoretical aspects of the Blockchain technology. We also devote a section to future of Blockchain and explain how extensions like Smart Contracts and De-centralized Autonomous Organizations will function.

Without assuming any reader expertise, our aim is to provide a concise but complete description of the Blockchain technology.

Additional Key Words and Phrases: Blockchain, Bitcoin

ACM Reference format:
Cuneyt Gurcan Akcora, Yulia R. Gel, and Murat Kantarcioglu. 2017. Blockchain: A Graph Primer. 1, 1, Article 1 (August 2017), 16 pages.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 BLOCKCHAIN

In simple terms, Blockchain is a distributed database that is secure by design. It was proposed by the unknown author Satoshi Nakamoto in 2008 [37]. Blockchain consists of blocks of transactions that can be verified and confirmed without a central authority. The technology has been popularized through its use in the digital currency Bitcoin, where each transaction is financial by nature.

As the origins of Blockchain start with Bitcoin, it is easy to confound the two. Mostly people refer to Bitcoin and Blockchain interchangeably.

It started with Bitcoin, but found usage in many new areas (See Section 3 by Mattila [31]). Companies have created Blockchain applications ranging from monitors that track diamonds, to networks that distribute food products in the globe. Recent years have seen Blockchain based commercial products from established tech companies such as Hyperledger Fabric [24] of IBM. This increasing interest has been fueled further by another popular Blockchain application: Ethereum.

XXXX-XXXX/2017/8-ART1 $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

In some aspects (e.g., asset transactions) new Blockchain applications differ from Bitcoin; we will outline these in Section 4. Nevertheless, from a graph perspective these differences are minor; Bitcoin transactions offer very good examples to explain how Blockchain works. Similarly, companies that track and analyze financial transactions on the Bitcoin network use the term *Blockchain analysis* to refer to their research efforts.

Another benefit of using Bitcoin is due to its long history. Bitcoin has been used worldwide by thousands of users since 2011, and its usage has both shown the utility of Blockchain, and highlighted its shortcomings in real life (e.g., delays in transactions).

As in the case of creating blocks of transactions, the core ideas behind Blockchain directly shape creation and maintenance of nodes and edges in graphs, and we will explain these ideas thoroughly. Blockchain lends itself easily to graph analysis; public addresses are linked with transactions, and transferred amounts correspond to weighted edges. In addition to the behavior imposed by the Blockchain core, some user practices change how Blockchain graphs are updated in time. An example of this is the common practice of moving remaining amounts of assets into new addresses at the end of each transaction.

Our aim is to provide a concise but complete summary of Blockchain for graph mining researchers. This manuscript will cover core aspects and user practices in Blockchain and explain how they affect research efforts. Some other aspects that we do not cover in this manuscript are privacy [14], and usage of digital currencies in Blockchain [47]. When needed, we will provide references for them throughout the manuscript.

We will start with a brief history of Blockchain in Section 2.

2 A BRIEF HISTORY

> Nakamoto may have been the mother of Bitcoin, but it is a child of many fathers: David Chaum's blinded coins and the fateful compromise with DNB, e-gold's anonymous accounts and the post-9/11 realpolitik, the cypherpunks and their libertarian ideals, the banks and their industrial control policies, these were the whole cloth out of which Nakamoto cut the invention.
>
> Ian Grigg [21]

Bitcoin [1] represents the culmination of efforts from many people and organizations to create an online digital currency. Notable currencies from early times are *ecash* from Chaum and *b-money* from Dai. Up to the seminal Bitcoin paper by Nakamoto [37], multiple times digital currencies seemed to have finally succeeded in creating a viable payment method [21]. Hindered by laws and regulations but mostly due to technical shortcomings, non of these currencies took root. Each failure, however, allowed digital enthusiasts to learn from the experience, and propose a new building block towards a viable currency.

From the beginning, digital currencies ran into several fundamental problems. A major hurdle was the need for a central authority to keep track of digital payments among users. Companies that invented digital currencies proposed to be the central authorities themselves. In a sense, rather than eliminating financial institutions, currencies tried to replace them. This approach never gained traction.

An alternative to the central authority approach was to use a distributed, public ledger to track user balances; every user stores account balances of all users. Although it sounds nice in theory, this solution is unfeasible because information about transactions cannot be digitally transmitted in fast and reliable ways. The networks are faulty and malicious users try to benefit from lying about balances. In fact, the problems with this approach, known as the Byzantine General's Problem, has been a well studied topic in distributed systems [28].

While the central authority problem was still an issue, in the unrelated spam detection domain a solution, called *HashCash*, was developed for a very different purpose [4]. Email providers had the problem of receiving

[1] In community practice, bitcoin is used for currency units, whereas Bitcoin refers to the software, protocol and community.

too many spam emails. Even though emails can be analyzed and marked as spam, this wasted system resources. Researchers were searching for ways to check emails for spam without using too much resource.

HashCash proposed an agreement between email senders and email providers. Email senders were tasked to do a computation for each email and append a proof of the computation to the email. The computation itself is designed to be time consuming, whereas checking the proof is easy. The email provider accepts incoming emails with valid proofs, and may discard all others without analyzing them. This scheme is called a **Proof-of-Work**. Note that an email sender can still create a spam email, compute its proof and send it. However, Proof-of-Work makes it too difficult to send too many such emails.

Nakamoto used Proof-of-Work to hinder block creation for two reasons. First, it makes lying about blocks more difficult; a miner needs to spend considerable power to mine a block. Second, the time it takes to create a new block allows the network to reach a consensus about transactions. Furthermore, each block contains information about the previous block, collating all in an immutable chain. This chain of blocks is called a **blockchain**.

Bitcoin's popularity brought a flood of alternative digital currencies. The most famous being LiteCoin, these **alt-coins** propose modifications to Bitcoin in aspects like block creation frequency and Proof-of-Work. More fundamental changes come from products not related to digital currencies. Ethereum, specifically, was designed to be the "World Computer" to de-centralize and democratize the Internet; a network of thousands of linked computers that are run by volunteers. We will discuss these improvements in Section 4.

3 BUILDING BLOCKS OF BLOCKCHAIN

3.1 Addresses

In its essence, a Blockchain address is a unique string of 26-35 characters created from public keys. Bitcoin uses two kinds of addresses:

- *Pay to PubkeyHash* address that starts with 1 as in 1Pudc88gyFynBVZccRJeYyEV7ZnjfXnfKn.
- *Pay to ScriptHash* address that starts with 3 as in 3J4kn4QoYDj95S3fqajUzonFhLyjjfKjP3.

The most common (we may even say the standard) type is the *Pay to PubkeyHash*, where a single private key is used to spend bitcoins received from a transaction. *Pay to ScriptHash* functionality was added later to support m-of-n multi signature transactions; at the receiving address bitcoins can be spend only when at least m out of n users use their private keys. In theory, all of n private keys can belong to the same user but stored on different computers, thereby reducing the risk of theft. In practice, keys belong to different users and *Pay to ScriptHash* is used in creative scenarios such as in wallet sharing, or buying/selling things with increased security.

From a graph perspective, the address type does not change anything; both can be used in transactions as input/output addresses. However *Pay to ScriptHash* makes initiation of a transaction very interesting, because it allows users to participate in spending decisions. This user behavior can be mined for various applications such as fraud prevention, and marketing.

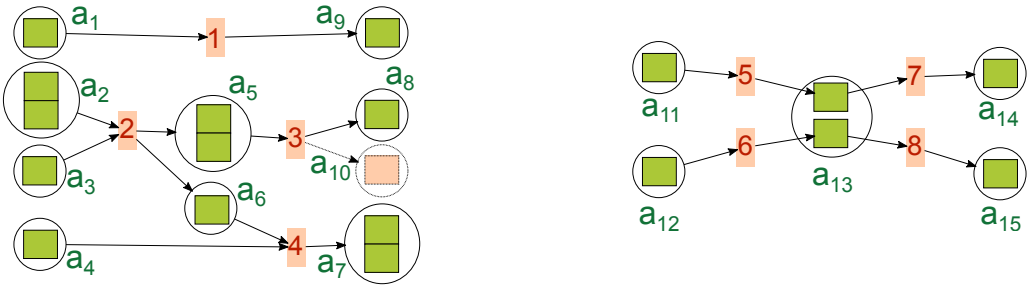
Each user publishes its address on web forums, mailing lists or other mediums. Holder of the address can manage the assets (in Bitcoin the asset is an amount of the bitcoin currency) acquired by transactions, because only she knows the private key associated with the address. A transaction cannot be initiated without a receiver address, and an error checking code appended to the address prevents typo errors in the address.

A key point to note is this: public keys are easily hashed to create addresses but it is a one way street. It is very difficult to compute the public key of a given address.

3.2 Transaction

A transaction is a transfer of assets between addresses. Bitcoin allows transferring funds from multiple addresses to multiple addresses. In practice, all input addresses belong to the same user, but output addresses might belong to different users. For each input, three data pieces are required: i) Id of the previous transaction that brought

Fig. 1. Transactions between addresses.



(a) Four basic type transactions among 10 addresses. Each green rectangle can be thought of as one Satoshi. Transaction 3 sends funds to a_{10} for transaction fees. Other transactions do not pay fees.

(b) Spending bitcoins received from two transactions in two new transactions. An address can receive inputs from multiple transactions, but when it spends, it has to spend each input completely.

the input asset, ii) index number of the output in the previous transaction, ii) amount to be transferred. When a transaction has more than one input, and each input is signed by the associated private key separately. This signature prevents the transaction to be altered and proves that the user has authorized the transfer.

When a transaction sends assets to an address, public key of the receiver address is unknown to the network. Only when the received assets are being spent in the future, the receiver shows its public key, and any user in the network can hash the key to verify that the hash equals the address. This is an additional proof that the assets belong to the receiver.

Figure 2a shows four types of transactions. Each green rectangle can be thought of as one Satoshi. A Satoshi is the minimum fraction of a bitcoin, 1 bitcoin = 10^8 Satoshi. Transaction 1 shows a 1/1 input output transactions. Transactions 2-4 have 2/2, 1/2, 2/1 input/output addresses respectively. When there is more than one input, input source cannot be distinguished. For example, in transaction 4, we cannot know whether the amount in a_6 comes from a_2 or a_3. As we will show in Section 5, these multiple input/output transactions create interesting graphs.

We specifically want to emphasize two aspects of transactions.

First, in transactions, senders do not specifically indicate a transaction fee; the difference between total inputs and total outputs is considered to be the transaction fee. The fee is sent to the address of the miner who confirms the transaction. Bitcoin transactions can be without fees (i.e., output amount is equal to the input amount), but as we will show in Section 3.3 these fees increase the chance of a transaction to be confirmed in the Bitcoin blockchain. Fees also prevent flooding the network with spam transactions. In Figure 2a only transaction 3 pays a fee. This is shown with an edge to the address a_{10}.

Second, an address can receive transfers from multiple transactions, and the outputs of these transactions can be spent separately. However, the community practice is to spend all inputs in a single transaction. For example, in Figure 2b, a_{13} receives transfers from transactions 5 and 6. a_{13} then spends two Satoshis in transactions 7 and 8. This is possible because each Satoshi was received in a different transaction. Now consider a_5 in Figure 2a. It also receives two Satoshis from transaction 4, but it has to spend both of them at transaction 3. If it only sends 1 Satoshi, the other Satoshi will be taken as the transaction fee, and any unspent amount will be lost. In practice, when a user has to spend a fraction of the received amount, it sends the remaining balance back to its address, or better, creates a new address and sends it there.

Manually creating transactions can be too tedious for ordinary users (See righto.com[2] for details.). Companies have created web sites, called **online wallets**, that allow users to send, receive and exchange bitcoins without dealing with transaction details.

3.3 Verification and Confirmation

Having covered addresses and transactions, we now turn our attention to how Blockchain, and in particular Bitcoin, makes use of them.

A digital currency, such as Bitcoin, has two main problems to address: payment verification and payment confirmation.

Payment verification means creating a mechanism to verify that 1) the spender has the necessary balance to make a payment, 2) the spender wants to pay the indicated amount. This is similar to verifying that someone wants to pay 50$ with authentic dollar bills. A user presents its public key (to show that the address belongs to her) and signs a signature with its private key (to confirm the amount). The transaction also lists as input a past transaction that brought the amount. For example, in Figure 2b, transaction 8 lists that its input is from the output of transaction 6.

These measures verify that a user has the balance, and wants to make a payment. In the ideal case, in a peer to peer system as soon as an amount is spent, everybody would see the transaction and record the new balances. Otherwise, the spender can use the same bitcoins to make multiple payments. This issue is known as the **Double Spending** problem.

In reality, the network is faulty and the news of a transaction may never reach some users. Furthermore, users could be malicious and lie about balances. Due to these problems, users may never reach a consensus about who owns how many bitcoins. Any payment would be a fraud risk. This issue is known as the Byzantine General's problem [28]. As we mentioned earlier in Section 2, early currencies used central authorities to solve this issue, but their fruitless efforts could only replace banks with companies.

Nakamoto proposed a unique solution to the Double Spending problem. Bitcoin uses a distributed public ledger that contains time-stamped and ID'ed blocks, which in turn contain a number of transactions. Each block carries a piece of information (i.e., block hash) about the previous block, so that users can follow how the blockchain grows in time. Blocks have some constraints: as transferring big blocks among peers would clog the network, Bitcoin limits the block size to 1 MB. As a future improvement, the **Segregated Witness** concept has been proposed to exclude transaction signatures from blocks so that more transactions can be fit into a block. When a new user joins the network, the whole chain is downloaded from peers and starting from the first block, all blocks and transactions are verified by the user. Because of this, it takes considerable time to be up-to-date with the blockchain.

Block creation is limited to one per ten minutes (this is more a wish than a rule) through a mechanism known as **Proof-of-Work**. Each block is very difficult to create but easy to confirm once created. Proof-of-Work requires finding a 32 bit number known as the *nonce* through trial and error. In a sense, it is similar to buying a ticket to win a lottery. Using more computing power increases the probability of finding the number, but does not guarantee it. Currently difficulty is such that it takes around 10^{20} attempts to find a valid nonce value. Users who work on finding confirmed blocks are called **miners**.

Proof-of-Work entails these steps: Each miner receives a list of transactions from its peers that are waiting to be confirmed. The list may be different for each miner. The miner picks a number of them to include in a block. While doing this, it may prefer transactions that pay a higher fee. First, a number of block specific data pieces, such as time of the block, hash of the previous block and a special hash value of contained transactions are concatenated. Then in each trial a different nonce is appended to the end of the string. The SHA-256 hash

[2]http://www.righto.com/2014/02/bitcoins-hard-way-using-raw-bitcoin.html

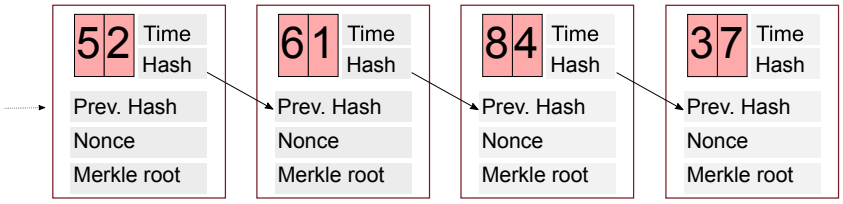


Fig. 3. A toy blockchain for the transactions given in Figure 1. Each block is limited to have two transactions and the coinbase transaction (not shown here).

value of this string is compared against the **difficulty**. If the hash value is smaller than the difficulty, the user is said to have mined a block.

The *difficulty* is a globally determined value that is periodically adjusted depending on how long it took to find the last 2016 blocks. In theory, Bitcoin requires that a block must take around 10 minutes, so 2016 blocks must take two weeks. However, depending on how many miners there are, it may take more or less than the predicted 2 weeks. [3] The new difficulty value is increased or decreased accordingly to approximate 10 minutes per block. The value has actually decreased multiple times so far.

Once a miner finds a block b, she sends b to its peers in the network, and hopes that this is included in the chain. As the information about b propagates in the network, users update their chains and set b as the latest block. Once notified about this new block, miners stop their ongoing computations. If changed, they update the difficulty, and remove transactions that are already included in b. Furthermore, miners change the *hash of previous block* with hash of b, and resume their mining efforts.

The first block of the chain, known as the *Genesis* block, was mined by Nakamoto. Miner or not, a user in the system considers the longest chain as the valid Blockchain.

In some cases, news about a block's discovery may reach some users late. Meanwhile another miner may come up with a new block. In this case, users receive two alternative blocks that **fork** from a previous block. In the main block chain these **forks** get into a race. As miners choose one block to build on, the fork that is shown to require "the most effort" becomes the main chain. If the difficulty is the same at both forks, the longest chain has the most effort. Although the last block may change, longer forks are not very common [29] . [4]

Figure 3 shows how blocks can be created from the transactions that we gave in Figure 1. Initially assume that the latest block is block 5-2 (i.e., Chain ..., 5-2). Assume that after the block 5-2 is appended to the blockchain , block 6-1 and 8-4 are mined at the same time. At this point, users and miners can choose any fork, and continue their efforts. While miner a builds on Chain ..., 5-2, 6-1, miner b searches for a new block on Chain ..., 5-2, 8-4. After all, both chains are the same length and choosing one over another is a kind of gambit. Next, miner a also mines block 8-4, so the fork becomes Chain ..., 5-2, 6-1, 8-4. Note from Figure 1 that blocks 6-1 and 8-4 contain different transactions, so they can be mined one after another. At this point, other miners in the system see that there are two forks and Chain ..., 5-2, 6-1, 8-4 is the longest. They choose to build on this longer chain, otherwise they risk their resources to be wasted on the shorter chain. Once the network miners start building on Chain ..., 5-2, 6-1, 8-4, the chances of miner b mining a longer fork gets even slimmer. There is no rule that stops miner b from continuing to work with its fork. Blockchain assumes that this effort will be futile because rest of the network will create a chain that is much longer than that of miner b.

A transaction is tentatively considered *confirmed* when it appears in a block. In practice a transaction is considered secure after six confirmations, i.e., six blocks.

[3]For example, In July 2017 it took 7.63 minutes to find a block in average. See https://blockchain.info/stats for the latest values.
[4]The longest fork was created due to a version mistake of the Bitcoin protocol, but was solved after 24 blocks.

With Proof-of-Work, Bitcoin developers ensure that deliberately creating a fork to eventually replace the main block chain becomes very expensive; a malicious user must mine new blocks faster than all other miners in the network combined. This means that the malicious miner must hold at least 51% of all mining resources, which is not probable. In **eclipse attacks**, a single user may be scammed by taking over all of its peers in the network. The isolated user can be fed a different fork for scam purposes.

Although transactions are secured by the mechanisms we mentioned so far, malicious users have found multiple ways to scam wallets and users (See an excellent survey on all scams in [14]). A very simple case involves **transaction malleability** to scam Bitcoin exchange websites. The scam works as follows: although transactions cannot be modified, transaction ids are not protected i.e., they are malleable. A user opens a wallet on an bitcoin exchange, and buys bitcoins. The user orders the wallet to send bitcoins to an address. As soon as the exchange publishes the transaction to send the users' bitcoin, the user herself captures the transaction, modifies the id and re-sends it to the network. Now, the network contains two copies of the same transaction. If a miner includes the fake transaction in a block, the real transaction will be rejected by all miners, because these bitcoins are already spent.

The exchange sees that the transaction was never included in a block, and may refund the malicious user. By repeating this attack many times, attackers caused denial of service attacks on the blockchain network in 2014. Malleability attacks are a nuisance more than a grave threat; they force users to track individual transaction outputs, which number in billions. This puts stress on the whole network.

In transactions, we briefly mentioned that miners who confirm blocks receive transaction fees. Transactions themselves do not have to pay fees, miners can put non-paying transactions in a block as well. However setting aside a fee increases the chances of being picked up. In fact, as Bitcoin receives more transactions, waiting queues have become longer. For the future, a possible remedy is to increase the block size from 1MB, so that blocks can contain more transactions. Similarly, 10 minute rule can be eased, and the Segregated Witness allows squeezing more transactions in a block.

In addition to transaction fees, the Bitcoin protocol creates a mining reward for each mined block. This reward transaction is known as the coinbase transaction, and is usually the first transaction in a block. In early days when Nakamoto was doing all the mining himself, a coinbase transaction was even the only transaction in a block.

Starting with 50 coins for each block and halving every 210K blocks, the total number of created bitcoins will pass 19M in 2022. This geometric series of rewards converges to a maximum of 21 million bitcoins [5]. Eventually when rewards become very small, it is expected that transaction fees will provide a sufficient incentive for miners.

4 CHANGES AND IMPROVEMENTS IN BLOCKCHAIN

As Bitcoin became popular, its limitations also became more visible. New generation blockchains learned from these limitations and improved over Bitcoin. Many others adopted the underlying Blockchain concepts for various use cases. Results range from minor tweaks to revolutionary ideas. In this section we will go over the most important ones.

4.1 Assets over cryptocurrency

Bitcoin uses transactions to transfer an asset: the bitcoin currency. A transaction consists of addresses, hash of the asset and a few other data pieces such as time. The key point is that the asset is represented by its hash value. This also implies that any hashable digital asset could appear in transactions. Law documents, contracts, agreements, wills and many other types of assets can be stored in the Blockchain. These asset based blockchains, however, would still need to use a currency to facilitate block mining. For example, although the Ethereum blockchain is

[5] 21M is chosen because when each coin is divided by 10^8 Satoshis, there will be as much Satoshis as there are state issued M1 currencies.

not a digital currency, it uses a currency named ether. Beyond digitized assets, some companies also give ids to physical objects such as diamonds and use Blockchain to track their movements in time (See Section 3.4 of [31] for examples). From a graph perspective, proliferation of exchanged assets will create a network that is best described by multilayer networks [9], where the same set of nodes are connected by edges of different natures. An example is the districts of a city being connected by bus, electricity and subway edges. On blockchain, edges will be exchanges of various assets. On a worldwide blockchain, this multilayer network will contain invaluable data about how assets move, change hands and increase/reduce the demand of each other.

4.2 Smart contracts

A new blockchain, Ethereum, has been created to implement not only transactions, but contracts which contain transactions with conditions and rules. Those so called smart contracts are written in proprietary coding languages and put to a network address by everyone to see and analyze. An analogy is the MYSQL snippets stored on a database. A contract clearly defines the functions that can be used, and is guaranteed to work in the way it is specified. A simple example is an exchange service of assets $\mathcal{A}$ and $\mathcal{B}$. The order of transactions is as follows:

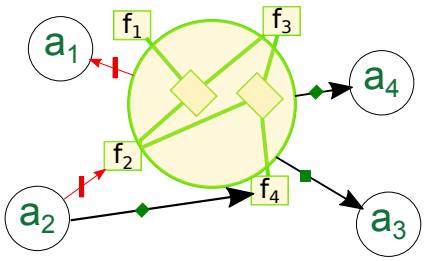


Fig. 4. Four addresses and a contract (it is an address as well). The contract has four functions and two decision boxes. Depending on the function used by a_2, and even the amount of transferred asset, a_1, or a_4 and a_5 can be linked to a_2.

(1) Alice writes a contract with three functions: deposit, draw and exchange, and puts this to an address. Exchange indicates that $1\mathcal{B} = 5\mathcal{A}$ and it rejects asset fractions. Alice signs the contract with her private key.
(2) Alice deposits 100 $\mathcal{A}$s to the contract's address. This transaction is recorded in the Blockchain.
(3) Bob has $\mathcal{B}$s and wants to exchange them for $\mathcal{A}$s. Bob sends 20 $\mathcal{B}$s to the contract's address.
(4) The contract automatically accepts 20 $\mathcal{B}$s and sends 100 $\mathcal{A}$s to Bob's address.
(5) Alice uses the draw function and receives 20 $\mathcal{B}$s from the contract.

Although Alice wrote the contract, the contract does not require any involvement from Alice before sending $\mathcal{B}$s to Bob. Even if Alice is malicious, she cannot intervene, take Bob's $\mathcal{B}$s and disappear. In reality, contracts are more complex, and there have been misuses. Formal verification of contracts to make sure that they work as intended is a developing research field [6].

Smart contracts bring a very interesting notion to graph analysis. A contract can be thought of a template for future edges. Once a node creates an edge with a contract (e.g., Step 2), the contract will create more edges (e.g., Step 4) that are predefined, but constrained by conditions (if Bob sends less than 5 $\mathcal{B}$s, exchange will be refused.). Figure 4 shows such a contract with its created edges. Edge templates such as this brings cascading behavior into mind. How can the graph be manipulated by planning to facilitate or restrict cascading behaviors?

Other than Ethereum, smart contracts are expected to be play a big role in the Internet of Things [13]. Once used by billions connected devices, smart contracts wll create huge amounts of graphs data.

4.3 Decentralized Autonomous Organizations

> When this happens, machines (hardware and software) become peers in the economy rather than mere tools. Imagine a world where drones that own themselves make deliveries, where autonomous software applications engage in virtual business such as buying and selling server time, or even buying and selling stocks and bonds. One day, you may just be hired by a machine, as a one-time gig, or perhaps even for full-time employment.
>
> Adam Hayes [23]

Ethereum's Smart Contracts made it possible to create a system where actors get into contracts with each other while matters, decisions and results of decisions can be recorded on the blockchain for everyone to see. This inspired the community to advocate for a future where Ethereum can be used to create an online democracy.

This utopic future is discussed under the term **Decentralized Autonomous Organizations (DAO)**.

The first attempt to create such a future was *the DAO project*, which aimed to create an online investor fund. Joining members could vote on decisions to invest, and earn money through their investments. Due to a hacking incident, the DAO became a traumatic experience that lost $50M and led Ethereum to split into two. We will mention this story in Section 4.4.

The DAO project left a bitter taste in the community and hindered development of new DAOs. Without a very successful example, there is not yet a consensus about what a DAO is and how it should make use of humans, robots, contracts and incentives. Some argue that other than creating contracts and putting them online, and occasionally providing services to DAO (and getting paid), humans should not be involved. Others use a few humans as *curators* that will manually filter incoming proposals to the DAO before a vote. We may add that even the term DAO is used in different meanings. However, considering that the blockchain itself does not require consensus, a consensus on DAO terms may be asking for too much.

Instead, a few influential developers have opined on what should be the common points of all DAOs. For example, the creator of Ethereum mentions three points [12].

(1) DAO is an entity that lives on the internet and exists autonomously.
(2) DAO heavily relies on hiring individuals to perform certain tasks that the automaton itself cannot do.
(3) DAO contains some kind of internal property that is valuable in some way, and it has the ability to use that property as a mechanism for rewarding certain activities

Opinions differ on how DAOs should function, but not on how influential they will be. Consider this example from Hayes [23]. A DAO acquires a car, and puts its contract on Ethereum. The DAO investors can vote (without management) to send the car to work at the Dallas Fort Worth Airport. Riders pay the car on the blockchain, and the car pays its investors dividends from what is left after gas, tax, maintenance and insurance fees. This DAO can buy new cars, decide on their working sites and even connect them to plan routes together. All of this is already possible through Ethereum.

4.4 Fork Issues

A short reading quickly reveals that most Blockchain technologies are forks of each other (See mapofcoins.com for a map). Blockchain uses **soft forks** to continue on the same main chain while changing a few aspects of the underlying technology. These are considered improvements or extensions. A soft fork is backward compatible, and reflects community consensus on how the network should evolve. **Hard forks**, on the other hand, creates a split in the main chain: two versions of the main chain are maintained by different groups of people. In a sense, it creates a new currency, technology or community. The most famous hard fork happened in the Ethereum project in 2016 due to the DAO hacking.

A hacker stole $50M from *the DAO project* which had raised $150M from the community for a proof-of-concept investor fund. The DAO was hacked because of coding mistakes that allowed multiple refunds for the same investment. Hackers drained DAO while the community was watching the theft in real-time, helplessly. Many developers wanted to roll back the transactions to forfeit the stolen amount. Disagreeing users rejected the roll back, and sticked to the existing blockchain. This created two versions: Ethereum and Ethereum Classic. This fork started a discussion on how Ethereum should be governed. The **Code is law** proponents claim that any behavior that conforms to the Blockchain protocols should be accepted; a theft, once happened, cannot be punished by rolling back transactions. For legal aspects, see the article by Abegg [1].

In August 2017, Bitcoin faced a hard fork of its own due to the Segregated Witness extension. Bitcoin split into two chains: Bitcoin and **Bitcoin cash**.

4.5 Tokens

Some blockchains, such as, Ethereum, encourages developers to create applications that use it to offer services. These applications are allowed to develop their own named assets, called tokens. Tokens are offered by the company and bought by investors using the blockchain currency. These tokens are similar to arcade tokens we used to buy at high school and insert into that helicopter game where you could shoot at ships directly or bounce bombs of the ground to do so. [6] So why tokens, instead of just using the system currency? Mainly because tokens can be used in creative ways. For example, a new service can sell its tokens to raise seed money. These **initial coin offerings (ICO)** have been very popular on the Ethereum blockchain. For token types and legal issues see [11].

4.6 Scalability Issues on Blockchain

Bitcoin expects a block to be mined every 10 minutes. It also imposes a block size of 1 MB, thereby limiting how many transactions can be processed in 24 hours. As a result, Bitcoin processes 7 transactions per second only. The payment method VISA, on the other hand, processes 2000 transactions per second, in average (See the bitcoin wiki for more details [8].).

Size and speed limitations have been eased in some other coins; LiteCoin, for example, mines blocks every 2.5 minutes. The Bitcoin community has been discussing ways to increase the number of transactions. A solution was adopted in 2017 with the Segregated Witness (SegWit) extension. Segregated Witness removes signatures from the block, which reduces block size by 60%. Further improvements would require increasing the block size itself, which is still debated in the community. See the bitcoin wiki for details [7].

4.7 Hyperledger Fabric

By definition any user can join a blockchain, and all transactions are immutable and public. For corporate settings, this transparency means that rivals can learn company finances and buy/sale relationships. The **Hyperledger Project** was created to use blockchain in industrial settings. Supported by many organizations, the Hyperledger offers membership services to choose blockchain participants and uses permissioned and even private blockchains. The project is hosted by the Linux Foundation and focuses on the storage, capacity and availability aspects of the blockchain. Hyperledger users are allowed to choose their own consensus and mining approaches. For more on Hyperledgers, see [48].

[6] Regretfully, the author could never remember the game's name.

4.8 Proof-of-X

A major criticism of Bitcoin is that while miners are in a race to find the next block, no thought goes into how much power and resource is wasted. In 2014 it was estimated that one bitcoin cost 15.9 gallons of gasoline. [7] It will only get worse as more miners join the race. As Swanson notes, Bitcoin is in reality a "peer-to-peer heat engine" [46].

Proof-of-X is an umbrella term that covers Proof-of-Work alternatives in block mining. Each alternative scheme expects miners to show a proof that they have done enough work or spent enough wealth before creating the block. In Proof-of-Work the work is the mining computations and proof is the hash value (see Section 3).

Instead of doing some work, some wealth can be destroyed (as a kind of sacrifice) to mine a block. **Proof-of-Stake** considers coin age as wealth; 10 coins of 2 years old means a wealth of $10 \times 2 = 20$. The block whose miner has sacrificed the highest wealth becomes the next block in the chain. Once coins are used as a sacrifice, their age becomes zero. Coins will have to accumulate their wealth again. **Proof-of-Burn** goes a step further, and indeed sacrifices coins. In the scheme a miner first creates a transaction and sends some coins to a "verifiably unspendable" address. These coins are called burned, but other than the sender, no one in the network is yet aware that the send address is an invalid/unusable one. Some time later, the miner creates a block and shows the proof of burnt coins. The proof is a script that shows how the address was created through erroneous computations. Miner who has burnt the highest number of coins can mine the next block. In this scheme burning coins to collect transaction fees is only viable if transaction fees are high enough.

If miners decide to game the system by supporting every competing fork, Proof-of-Work becomes too expensive due to required computations. Other schemes are not as effective against miner malice. Consider the case with two competing forks in the blockchain. In Stake or Burn based proofs miners can create blocks for each fork by using the same (burned or aged) coins. Eventually one of the forks will be the longest and become the main chain. Regardless of which one is chosen, the miner's block will have taken its place in the chain, and the miner will reap fees. For an extensive survey on Proof-of-X schemes, see Section 6 of [47].

5 BLOCKCHAIN ANALYSIS

Blockchain technologies have been used worldwide in various scenarios. As Bitcoin became popular, research works analyzed the network for coin price predictions. Various features, such as mean account balance, number of new edges and clustering coefficients have been used in research works [20, 44]. other than features, network flows [49] and temporal behavior of the network [25] have also been used in predictions.

Studies in network features show that since 2010 the Bitcoin network can be considered a scale-free network [29]. In and out degree distributions of the transactions graphs are highly heterogeneous and show disassortative behavior [26]. Active entities on the network change frequently, but there are consistently active entities [38]. The most central nodes in the network are coin exchange sites [5].

On Blockchain, each coin is minted by a miner, and it changes hands through transactions. In theory, it is possible to track a coin in time, and see where it originates from.

Taint Analysis [15] is used to find out whether a unit of currency is tainted because it is acquired through a crime such as theft or ransom. If a unit of good can be exchanged by the same unit of good in another place, the good is deemed **fungible**. For example, gold is fungible because a gram of gold in the US is equal to a gram of gold in Russia. Bitcoins, however, may not be fungible because selling or buying tainted coins is difficult due to legal issues.

Blockchain analysis started with taint analysis, but soon expanded its scope. From bitcoin value predictions to scalability studies, blockchain analysis now covers many topics.

[7] http://www.coindesk.com/carbon-footprint-bitcoin/

Each Blockchain application has two layers: application and network. In the application layer, content graphs reflect transfers of assets on the Bitcoin protocol. In the network layer, communication graphs reflect the underlying P2P communication network among Bitcoin addresses.

Content graphs are mainly used to track and locate stolen assets. They can also be used for marketing purposes. Communication graphs have diverse use cases. For example, they have been analyzed to improve the scalability of Bitcoin. Furthermore, content and communication graphs can be integrated for various purposes, such as locating real identities of Bitcoin addresses. In this section we will look at these practices.

5.1 Content Graphs

Content graphs are created in three forms: transaction, address and entity/user graphs. Figure 5 shows the graph for transactions given in Figure 1. By nature, the **transaction graph** [19] is acyclic; a transaction can have incoming edges once only. Each edge represents a transferred asset, and can be denoted with a weight. Starting with the genesis block, each incoming edge is either a transfer, or a coinbase transaction. For example, in Figure 5, incoming edge to transaction 1 might be a bitcoin gained from coinbase transaction. The edge between transaction 4 and 5 is a transfer edge. Transaction graphs are the largest graphs in Blockchain analysis. Storing them and working with them can be very costly. Using dominant sets [3] or filtering transactions by weight [30] before analysis are possible approaches.

The **address graph** uses addresses as nodes, and can be cyclic. Figure 7a shows the address graph of our recurring example from Figure 1. As in the transaction graph, each edge carries a weight (i.e., bitcoin amount), but there can be multiple edges between addresses. Furthermore, if change is sent back to the input address, the graph contains self loops. In practice, however, an address is used to send assets only once. As a community rule, **address reuse** is generally avoided to prevent cryptography attacks (See the appendix for address attacks). Due to this, loops and multi edges are rare. A major aspect of address graphs is this: as shown in edges from a_2 and a_3 to nodes a_5 and a_6, all input addresses contained within a transaction are deemed to have edges to each output address. This complicates tracking how assets are transferred from input to output addresses.

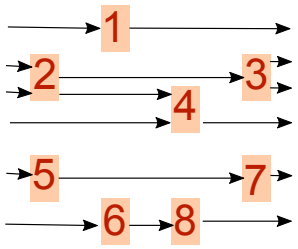


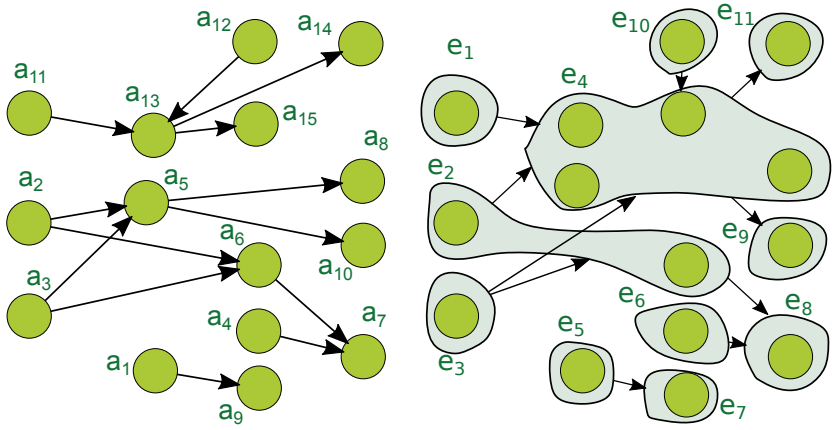
Fig. 5. A graph of the 8 transactions from Figure 1. Transactions are ordered on the horizontal axis by their appearance in blocks (see Figure 3).

Differing from others, the **entity graph** tries to find which addresses are owned by the same entity. These efforts are also known as user/entity clustering. Some entities, such as Wikileaks, publicly advertise its address. Some other addresses can be found on mailing lists, forums etc. publicizing an address as yours user might provide benefits; if the user is known (i.e., trusted) transactions from her address can be accepted by vendors without waiting 6 blocks. These known and trusted addresses are called **Green/Marked addresses**.

In general, there is not a clear cut method to link addresses together. Some works have proposed graph algorithms to cluster users [42]. Despite having false positives, heuristics have been widely used by community and in research works. These heuristics are Peeling Chain [34], Change Closure [2], Idioms of Use, Transitive Closure, Ip Clustering [39], and Temporal Clustering [39].

The **Peeling Chain** is frequently used in criminal activities to divide funds, and hide their origins. In a Peeling Chain, an address contains an initial amount of bitcoins. From the initial amount, a small amount is peeled and the remainder is sent to a change address. This peeling can continue thousands of times until the initial sum is divided and transferred to the peeled addresses. All the peeling addresses are expected to belong to the same entity. **Change Closure** follows the community practice of sending whatever change remains from a transaction

Fig. 6. Blockchain graphs



(a) Address graph with edges in horizontal time order. Note that a_13 receives input from two addresses in different blocks.

(b) Entity graph with owners of addresses shown as super nodes. This graphs helps us to see the change transfers.

to a new address owned by the spender. For this heuristic, the change address should never appear before, and never be re-used to receive payments. In Figure 7a after paying a_10, a_5 sends the change to a_8. Both a_5 and a_8 are deemed to belong to the same entity. **Idioms of Use** posits that all input addresses in a transaction should belong to the same entity because only the owner could have signed the inputs with associated private keys. By this heuristic, a_2 and a_3 from Figure 1 belong to the same user. **Transitive Closure** extends Idioms of Use: if a transaction has inputs from a_x and a_y, whereas another transaction has from a_x and a_z, a_y and a_z belong to the same entity. **Ip Clustering** proposes using the network layer to find entity clusters (We will focus on the network layer in Section 5.2). Bitcoin addresses that use the same IP address are merged. **Temporal Clustering** is a more difficult approach; it posits that similar transfers at similar times imply that addresses are managed by the same entity.

By nature all user clustering heuristics are error prone. Some community practices further complicate the issue. For example, online wallets such as coinbase.com buy/sell coins among its customers without using transactions; ownership of an address is changed by transferring the associated private keys to another user. Although the user associated with the address changes, nothing gets recorded in the blockchain.

Another measure to prevent matching addresses to users is known as **CoinJoin** [32]. The key idea is to use a central server that mixes inputs from multiple users. Only the server knows the mapping between inputs and outputs. Mistakenly, the Idioms of Use heuristic would mark all these input addresses to belong to the same user. Relying on a central server is a major weakness in CoinJoin, but there are attempts to develop viable alternatives [43]. We expect CoinJoin related mixing technologies to be widely used in future. *Flow* of bitcoins among addresses can be quantified [18] and *purity* measures to detect suspicious mixing has been proposed in [15]. For limitations of bitcoin laundering, mixing and shared bitcoin sending, see [36].

The increasing usage of coins in crime has necessitated finding users/entities behind addresses. For example, ransom software that encrypt hard drives use Bitcoin as a medium for ransom payments. Coins have also been stolen (i.e., transferred to addresses of thieves) from online wallets such as Mt.Gox. Companies, such as Elliptic and Numisight develop Blockchain graph analysis tools to track these crime related coins. In research works,

BitIodine [45] and McGinn [33] offer visualization tools. GraphSense [22] is an analytics platform that also provides path search on transaction graphs.

With so many works on identifying entities behind addresses, privacy research on Blockchains have also become an interesting topic. See [2] for a recent study on user privacy on Bitcoin.

5.2 Communication graphs

At its core, Bitcoin maintains a peer-to-peer architecture [16, 17]. Bitcoin peers create persistent TCP channels with each other and relay transactions. Each peer seeks a minimum of 8, a maximum of 125 peers. It can accept connection requests from more peers. Neighbors of a peer can also be asked and retrieved.

Bitcoin is an unstructured network; centralized directories do not exist and network topology is not managed or controlled. Peers join and leave as they want. Without super nodes that connect to many peers, diameter of the network increases in time with new users. Although resilient, this causes the network to not scale well; network expansion may result in communication latency between peers. To make the matters worse, usage of gateways and autonomous systems mean that not all peers are directly connectible [17]. As a result, information on newly confirmed blocks takes longer time to propagate in the network. This may create short term forks on the blockchain. Hierarchical models have been proposed to alleviate these problems [40].

Even when not joining mining efforts, a peer's main role is to relay correctly signed and valid transactions. A peer regularly sends ids (that is, hashes) of transactions to its peers. If a transaction hash has not been seen before, its details (i.e., inputs/outputs of the transaction etc.) are requested from the relaying peer. This mechanism is also used to inject a new transaction in to the system. The origin peer advertises its own transaction by sending its hash to other peers. This, however, puts the origin peer in risk; by observing how a transaction is relayed in the network, the origin peer can be located. See [27] for a detailed analysis of transaction relays.

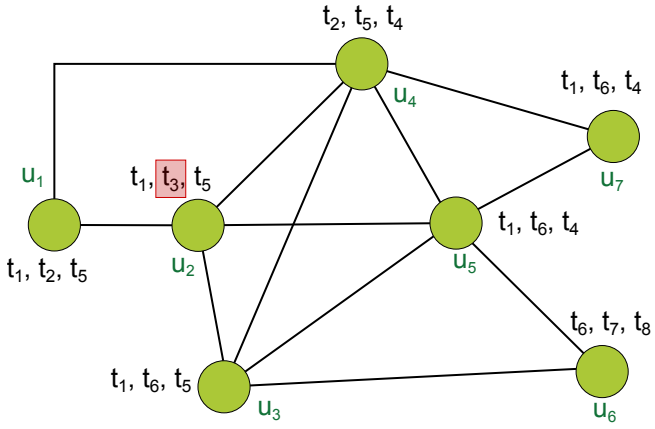


Fig. 8. A graph with seven peers and three trickled transactions at each peer.

Bitcoin uses **trickling** of transactions to improve privacy: peers randomly choose and relay only 1/4 of transactions from their pools. Own transactions, on the other hand, is always trickled.

Figure 8 shows when trickling cannot hide the origin peer. In the figure, transaction t_3 originates from u_2. u_4 is connected to all the peers of u_2. As it tracks all trickles, u_4 will learn that t_3 must have originated from u_2.

In the future u_1 u_3, u_4 and u_5 will have t_3 in their transaction pools. If u_5 and u_3 both trickle t_3 at the same time, u_6 will also learn that t_3 did not originate from any of them. This negation can be useful when connected to a big number of peers. Bitcoin's unstructured network is designed to avoid super nodes with this privacy risk in mind.

Trickling makes it more difficult to locate origin peers, but it also withholds some transactions and thereby adds latency in relays. This unwanted latency can be tolerated in transactions, but not in blocks. Blocks are always relayed as soon as they arrive. Despite this Donet et al. [16] report that "the block traffic is very messy,

and best 20 nodes (in terms of transaction and block propagation speed) are responsible for relaying more than 70% of both blocks and transactions".

On the Bitcoin network the peer from which a transaction is heard for the first time has a very high probability to be the origin peer, but this heuristic is not always correct [27]. Nevertheless, research works have found that transaction origin can be guessed with some success [35, 41].

A worrying sign is that the Bitcoin network has very high degree nodes. Although a peer is supposed to have minimum 80, maximum 125 peers, Millet et al. [35] found that "extremely high degree nodes are persistent over time". These nodes hold 80 times many peers than usual, and can be used to discover transactions origins. A possible solution is a costly redesign of the underlying P2P protocol with anonymity guarantees [10].

REFERENCES

[1] Lukas Abegg. 2016. Code is Law? Not Quite Yet. Online. (2016). https://www.coindesk.com/code-is-law-not-quite-yet/

[2] Elli Androulaki, Ghassan O Karame, Marc Roeschlin, Tobias Scherer, and Srdjan Capkun. 2013. Evaluating user privacy in bitcoin. In *International Conference on Financial Cryptography and Data Security*. Springer, 34–51.

[3] Malik Khurram Awan and Agostino Cortesi. 2017. Blockchain Transaction Analysis Using Dominant Sets. In *IFIP International Conference on Computer Information Systems and Industrial Management*. Springer, 229–239.

[4] Adam Back et al. 2002. Hashcash-a denial of service counter-measure. (2002).

[5] Annika Baumann, Benjamin Fabian, and Matthias Lischke. 2014. Exploring the Bitcoin Network.. In *WEBIST (1)*. 369–374.

[6] Karthikeyan Bhargavan, Antoine Delignat-Lavaud, Cédric Fournet, Anitha Gollamudi, Georges Gonthier, Nadim Kobeissi, Natalia Kulatova, Aseem Rastogi, Thomas Sibut-Pinote, Nikhil Swamy, et al. 2016. Formal verification of smart contracts: Short paper. In *Proceedings of the 2016 ACM Workshop on Programming Languages and Analysis for Security*. ACM, 91–96.

[7] Bitcoin. 2016. Block size limit controversy. Online. (2016). https://en.bitcoin.it/wiki/Block_size_limit_controversy

[8] Bitcoin. 2016. Scalability. Online. (2016). https://en.bitcoin.it/wiki/Scalability

[9] Stefano Boccaletti, Ginestra Bianconi, Regino Criado, Charo I Del Genio, Jesús Gómez-Gardenes, Miguel Romance, Irene Sendina-Nadal, Zhen Wang, and Massimiliano Zanin. 2014. The structure and dynamics of multilayer networks. *Physics Reports* 544, 1 (2014), 1–122.

[10] Shaileshh Bojja Venkatakrishnan, Giulia Fanti, and Pramod Viswanath. 2017. Dandelion: Redesigning the Bitcoin Network for Anonymity. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 1, 1 (2017), 22.

[11] Demian Brener. 2016. On Tokens and Crowdsales: How Startups Are Using Blockchain to Raise Capital. Online. (2016). https://www.coindesk.com/tokens-crowdsales-startups/

[12] Vitalik Buterin. 2014. DAOs, DACs, DAs and More: An Incomplete Terminology Guide. Online. (2014). https://blog.ethereum.org/2014/05/06/daos-dacs-das-and-more-an-incomplete-terminology-guide/

[13] Konstantinos Christidis and Michael Devetsikiotis. 2016. Blockchains and smart contracts for the internet of things. *IEEE Access* 4 (2016), 2292–2303.

[14] Mauro Conti, Chhagan Lal, Sushmita Ruj, et al. 2017. A Survey on Security and Privacy Issues of Bitcoin. *arXiv preprint arXiv:1706.00916* (2017).

[15] Giuseppe Di Battista, Valentino Di Donato, Maurizio Patrignani, Maurizio Pizzonia, Vincenzo Roselli, and Roberto Tamassia. 2015. Bitconeview: visualization of flows in the bitcoin transaction graph. In *Visualization for Cyber Security (VizSec), 2015 IEEE Symposium on*. IEEE, 1–8.

[16] Joan Antoni Donet Donet, Cristina Pérez-Sola, and Jordi Herrera-Joancomartí. 2014. The bitcoin P2P network. In *International Conference on Financial Cryptography and Data Security*. Springer, 87–102.

[17] Sebastian Feld, Mirco Schönfeld, and Martin Werner. 2014. Analyzing the Deployment of Bitcoin's P2P Network under an AS-level Perspective. *Procedia Computer Science* 32 (2014), 1121–1126.

[18] Erwin Filtz, Axel Polleres, Roman Karl, and Bernhard Haslhofer. 2017. Evolution of the Bitcoin Address Graph. (2017).

[19] Michael Fleder, Michael S Kester, and Sudeep Pillai. 2015. Bitcoin transaction graph analysis. *arXiv preprint arXiv:1502.01657* (2015).

[20] Alex Greaves and Benjamin Au. 2015. Using the Bitcoin Transaction Graph to Predict the Price of Bitcoin. (2015).

[21] Ken Griffith. 2014. A quick history of cryptocurrencies BBTC-Before Bitcoin. *Bitcoin Magazine. April* 16 (2014).

[22] Bernhard Haslhofer, Roman Karl, and Erwin Filtz. 2016. O Bitcoin Where Art Thou? Insight into Large-Scale Transaction Graphs.. In *SEMANTiCS (Posters, Demos, SuCCESS)*.

[23] Adam Hayes. 2016. Decentralized Autonomous Organizations: IoT Today. Online. (2016). http://www.investopedia.com/articles/investing/022916/decentralized-autonomous-organizations-iot-today.asp

[24] IBM. 2017. Hyperledger. Online. (2017). https://www.ibm.com/blockchain/hyperledger.html

[25] Dániel Kondor, István Csabai, János Szüle, Márton Pósfai, and Gábor Vattay. 2014. Inferring the interplay between network structure and market effects in Bitcoin. *New Journal of Physics* 16, 12 (2014), 125003.

[26] Dániel Kondor, Márton Pósfai, István Csabai, and Gábor Vattay. 2014. Do the rich get richer? An empirical analysis of the Bitcoin transaction network. *PloS one* 9, 2 (2014), e86197.
[27] Philip Koshy, Diana Koshy, and Patrick McDaniel. 2014. An analysis of anonymity in bitcoin using p2p network traffic. In *International Conference on Financial Cryptography and Data Security*. Springer, 469–485.
[28] Leslie Lamport, Robert Shostak, and Marshall Pease. 1982. The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)* 4, 3 (1982), 382–401.
[29] Matthias Lischke and Benjamin Fabian. 2016. Analyzing the bitcoin network: The first four years. *Future Internet* 8, 1 (2016), 7.
[30] Damiano Di Francesco Maesa, Andrea Marino, and Laura Ricci. 2016. Uncovering the Bitcoin Blockchain: An Analysis of the Full Users Graph. In *Data Science and Advanced Analytics (DSAA), 2016 IEEE International Conference on*. IEEE, 537–546.
[31] Juri Mattila et al. 2016. *The Blockchain Phenomenon–The Disruptive Potential of Distributed Consensus Architectures*. Technical Report. The Research Institute of the Finnish Economy.
[32] Greg Maxwell. 2013. CoinJoin: Bitcoin privacy for the real world. In *Post on Bitcoin Forum*.
[33] Dan McGinn, David Birch, David Akroyd, Miguel Molina-Solana, Yike Guo, and William J Knottenbelt. 2016. Visualizing dynamic Bitcoin transaction patterns. *Big data* 4, 2 (2016), 109–119.
[34] Sarah Meiklejohn, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoffrey M Voelker, and Stefan Savage. 2013. A fistful of bitcoins: characterizing payments among men with no names. In *Proceedings of the 2013 conference on Internet measurement conference*. ACM, 127–140.
[35] Andrew Miller, James Litton, Andrew Pachulski, Neal Gupta, Dave Levin, Neil Spring, and Bobby Bhattacharjee. 2015. Discovering bitcoin's public topology and influential nodes. *et al.* (2015).
[36] Malte Moser, Rainer Bohme, and Dominic Breuker. 2013. An inquiry into money laundering tools in the Bitcoin ecosystem. In *eCrime Researchers Summit (eCRS), 2013*. IEEE, 1–14.
[37] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. (2008).
[38] Micha Ober, Stefan Katzenbeisser, and Kay Hamacher. 2013. Structure and anonymity of the bitcoin transaction graph. *Future internet* 5, 2 (2013), 237–250.
[39] Marc Santamaria Ortega. 2013. The bitcoin transaction graph anonymity. *Master's thesis, Universitat Oberta de Catalunya* (2013).
[40] Stephen L Reed. 2014. Bitcoin cooperative proof-of-stake. *arXiv preprint arXiv:1405.5741* (2014).
[41] Fergal Reid and Martin Harrigan. 2013. An analysis of anonymity in the bitcoin system. In *Security and privacy in social networks*. Springer, 197–223.
[42] Dorit Ron and Adi Shamir. 2013. Quantitative analysis of the full bitcoin transaction graph. In *International Conference on Financial Cryptography and Data Security*. Springer, 6–24.
[43] Tim Ruffing, Pedro Moreno-Sanchez, and Aniket Kate. 2014. CoinShuffle: Practical decentralized coin mixing for Bitcoin. In *European Symposium on Research in Computer Security*. Springer, 345–364.
[44] Mariano Sorgente and Cristian Cibils. 2014. The Reaction of a Network: Exploring the Relationship between the Bitcoin Network Structure and the Bitcoin Price. (2014).
[45] Michele Spagnuolo, Federico Maggi, and Stefano Zanero. 2014. Bitiodine: Extracting intelligence from the bitcoin network. In *International Conference on Financial Cryptography and Data Security*. Springer, 457–468.
[46] Tim Swanson. 2014. Learning from Bitcoin's past to improve its future. (2014).
[47] Florian Tschorsch and Björn Scheuermann. 2016. Bitcoin and beyond: A technical survey on decentralized digital currencies. *IEEE Communications Surveys & Tutorials* 18, 3 (2016), 2084–2123.
[48] M Walport. 2016. *Distributed Ledger Technology: Beyond Blockchain. UK Government Office for Science*. Technical Report. Tech. Rep.
[49] Steve Y Yang and Jinhyoung Kim. 2015. Bitcoin Market Return and Volatility Forecasting Using Transaction Network Flow Properties. In *Computational Intelligence, 2015 IEEE Symposium Series on*. IEEE, 1778–1785.