# A Study on Nine Years of Bitcoin Transactions: Understanding Real-world Behaviors of Bitcoin Miners and Users

Binbing Hou*
LinkedIn Co.
bhou@linkedin.com

Feng Chen
Louisiana State University
fchen@csc.lsu.edu

***Abstract*—Bitcoin is the world's first blockchain-based, peer-to-peer cryptocurrency system. Being tremendously successful, the Bitcoin system is designed to support reliable, secure, and trusted transactions between untrusted peers. Since its release in 2009, the Bitcoin system has rapidly grown to an unprecedentedly large scale. However, the real-world behaviors of miners and users in the system and the efficacy of the original Bitcoin system design in the field deployment still remain unclear, hindering us from understanding its internals and developing the next-generation cryptocurrency system.**

**In this paper, we study the behaviors of Bitcoin miners and users and their interactions based on quantitative analysis of more than nine years of Bitcoin transaction history, from its first release on January 3rd, 2009 to April 30th, 2018. We have analyzed over 300 million transaction records to study the transactions' processing, confirmation, and implementation. We have obtained several critical findings regarding how the miners and users exploit the high degree of freedom provided by the Bitcoin system to achieve their own interests. For example, we find that miners often attempt to maximize their profits even by sacrificing system performance; users could try to speed up the transaction processing by mistakenly trading off security for reduced latency. Such unexpected behaviors, to some degree, deviate from the original design purposes of the Bitcoin system and could bring undesirable consequences. Besides revealing several unexpected behaviors of the Bitcoin miners and users in the real world, we have also discussed the associated system implications as well as optimization opportunities in the future.**

## I. Introduction

Bitcoin is the world's first blockchain-based, peer-to-peer cryptocurrency system [1]. Since its first release in 2009, the acceptance of Bitcoin has been dramatically expanding from a niche community to the general public worldwide. In the recent years, its peak market capitalization has surpassed $100 billion [2].

Unlike conventional financial systems, which rely on a trusted third party for financial endorsement and system management, the Bitcoin system is a peer-to-peer electronic cash system maintained and used by participants, who play two main roles—*Miners* run the nodes that form the Bitcoin network and work collaboratively to process transactions, which are written into a publicly shared database, called *blockchain*; *Users* submit and finalize transactions to make payments. Being designed to support reliable, secure, and trusted transactions between untrusted peers, the Bitcoin system provides a high degree of freedom for the participants, allowing them to prioritize transaction processing, customize transaction implementation, determine transaction confirmation, etc. In other words, except following several basic system protocols and rules, the participants in the Bitcoin system have a high degree of flexibility and freedom to make decisions based on their own interests. Unfortunately, such "optimizations" could sometimes be against the original design purposes and cause unexpected system performance deficiency and even security loopholes, as we will discuss later in this paper.

Due to its great success, the Bitcoin system and its underlying blockchain technology have raised widespread interests in both academia [3], [4], [5], [6], [7], [8], [9], [10], [11] and industry [12], [13], [14], [15], [16]. Some public websites, online documents, and research papers have introduced the Bitcoin system to the public [17]. Statistical data are also reported on Bitcoin usage [18], [19], [20], [21]. Some academic research has studied the Bitcoin system based on theoretical models [3], [5], [9], [22]. However, the Bitcoin system is a large-scale peer-to-peer system composed of over one million miners and millions of users [23], [24], who work individually and collaboratively together in a distributed manner. It still remains a highly interesting but unanswered question—*how these Bitcoin participants behave and interact with each other in the real-world system deployment?*

In this paper, we study the behaviors of Bitcoin miners and users and particularly investigate the effect of these behaviors in practice. To achieve this, we conduct a quantitative study on more than nine years of Bitcoin transaction history (from its first release on January 3rd, 2009 to April 30th, 2018), which contains over 300 million transaction records. We analyze the transaction records in three aspects, i.e., transaction processing, confirmation, and implementation, which together reflect how the miners and users use the Bitcoin system to achieve their interests. It is worth noting that since our purpose is to investigate how the critical system mechanisms are utilized in real-world deployment, we mostly focus on the behaviors of

*This work was done while the author was a Ph.D. student at Louisiana State University.

miners and users observed from the system level via statistical analysis, rather than the behaviors of individual miners and users.

Based on our quantitative study, we have obtained several important findings:

- To incentivize the miners for processing transactions, the miners are allowed to collect and compete for transaction fees and mining rewards. Our data analysis shows an unexpected negative effect—the miners tend to maximize the obtainable benefits in various means, even at the cost of transaction processing efficiency. For example, the current fee-rate-based prioritization policy adopted by miners is highly biased against low-fee-rate transactions, which can cause about 15%-16.6% of the coins in the system to be frozen and not spendable. Even worse, to win the block competition for obtaining the incentives, the miners prefer to create a relatively small block, which not only degrades the already-low transaction processing performance but also undermines the effort of improving system performance by increasing the block size limit.
- To guarantee the security of a transaction, users are recommended to wait for a high number of confirmations to finalize the transaction. However, we find that at least 55.22% of all the transactions are completed with at most five confirmations (although the standard recommendation is six confirmations). More surprisingly, at least 21.27% of all the transactions are zero-confirmation transactions, which clearly violate the basic rule for using the Bitcoin system that a transaction should be finalized with at least one confirmation. This means that a large percentage of users are making a risky decision to sacrifice the transacting security to improve transacting speed by shortening the necessary waiting time for confirmations.
- To enable the customization of peer-to-peer transactions, the users are provided with a scripting language for transaction implementation. Our findings are two-fold. First, we find that 99.7% of all the transactions are standardized rather than customized, indicating that most users show very low interests in, or are simply incapable of customizing transactions. It means that the need for customizing transactions in the original Bitcoin system design was likely to be overrated. Second, and more importantly, we find that such an unnecessary flexibility for transaction implementation could even lead to erroneous or harmful transaction implementation, causing undesirable fund loss and exposing the system to implementation bugs and malicious attacks.

These observed behaviors, to some degree, deviate from the original design purposes of the Bitcoin system and could bring many unexpected consequences. Our study strongly suggests that it is the time to revisit many design choices in the Bitcoin system, especially on the current high level of freedom provided to the miners and users. We believe that the issues revealed in this work not only expose many challenges but also open numerous research opportunities for optimizing the system design and implementation of Bitcoin and other similar cryptocurrency systems.

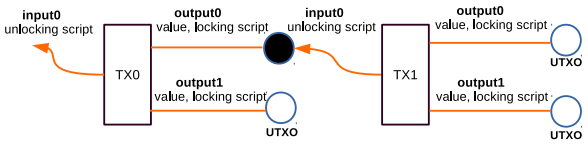


Fig. 1: An illustration of two Bitcoin transactions `TX0` and `TX1`. The black circle represents the transaction output that has been spent, and the white circles represent the unspent transaction outputs.

The rest of the paper is organized as follows. Section II introduces the background. Section III describes our methodology for data analysis. Section IV to VI present our observations on transaction processing, confirmation, and implementation. Section VII discusses the system implications. Section VIII gives the related work. The final section concludes the paper.

## II. Background

### A. Transaction Implementation

**Coin-based transacting model.** A Bitcoin transaction represents the fund transfer between a sender and a receiver. A transaction contains a list of *inputs* for the sender to spend (send) the previously received funds, and a list of *outputs* for the receiver to receive funds. Shown in Figure 1 is an illustrative example of two transactions `TX0` and `TX1`, each having one input and two outputs. Each output is associated with a value indicating the amount of fund and a *locking script* locking the fund; each input references a previously *Unspent Transaction Output* (UTXO) and provides an *unlocking script* to unlock and spend the fund. A UTXO is generally called a *coin*. The value of a coin is the value associated with the UTXO, and is measured as *BTC*. One BTC can be sliced into smaller units, such as *Satoshi* (1 BTC = 100 million Satoshis).

**Transaction scripts.** The Bitcoin system provides a unique scripting language to implement the locking and unlocking scripts. The scripting language supports 256 opcodes. Each opcode is an instruction [25]. A script is composed of a set of opcodes and the related data. To illustrate the scripting mechanism, we take a standard transaction type, P2PKH (Pay-to-Public-Key-Hash), as an example:

P2PKH utilizes the Elliptic Curve Digital Signature Algorithm (ECDSA) [26] based on public/private key pairs. For receiving the fund, the receiver first generates a pair of public/private keys. The public key is hashed into a 160-bit binary value, `pubkey hash`, using two cryptographic hash functions SHA-256 [27] and RIPEMD-160 [28]. The `pubkey hash` is further encoded using the BASE58 encoding scheme [29] to create a human-readable alphanumeric string, called *Bitcoin address*. The receiver then sends the Bitcoin address to the sender.

After receiving the address, the sender decodes it to regenerate the 160-bit `pubkey hash` using the BASE58 decoding

scheme [29]. The sender embeds `pubkey hash` in the locking script to lock the fund as a coin. The locking script is as follows:

```
OP_DUP OP_HASH160 <pubkey hash> OP_EQUALITY
OP_CHECKSIG
```

The above locking script requires the receiver to provide an unlocking script to unlock and spend the coin. The unlocking script contains two elements:

```
<sig> <pubkey>
```

The provided public key `<pubkey>` is valid only if its hash is equal to `<pubkey hash>`. The provided signature `<sig>` is derived from the receiver's private key, and its validity can be verified by using the public key `<pubkey>`. The unlocking script is valid only if both the public key and the signature are valid. When processing a transaction, the miners verify the validity of spending a coin based on the locking and unlocking scripts, which are combined and executed in a stacked-based manner [30].

In addition to P2PKH, the Bitcoin system supports several other standard transactions [30], such as P2PK, P2SH, OP_Multisig, and OP_RETURN. Users can also customize the implementation of transaction scripts.

**Transaction fee.** A transaction fee is the difference between the value of the coins being sent and the value of the coins being received, which is a reward to incentivize miners to process user transactions. In Figure 1, for example, the fee for `TX1` is the difference between the value of `output0` of `TX0` and the sum of the value of `output0` and `output1` of `TX1`. A user can specify the fee for processing a transaction.

### B. Transaction Processing

Miners are responsible for transaction processing: each miner runs a node to process transactions and maintain transaction records (i.e., the ledger).

**The longest-chain protocol.** The miners process and group transactions into blocks, which are stored in a distributed database, called *blockchain*. Blocks in the blockchain are organized as a singly linked list, each block (except the first one) containing the cryptographic hash of its previous block. Block conflicts happen when two blocks are linked to the same previous block within a short time period (1 minute). To address such conflicts, the Bitcoin system adopts the *longest-chain* protocol. It temporarily reserves all the conflicting blocks, forming different branches, and the newly generated blocks can be added to any branch. Only the blocks of the longest branch are finally kept and the other blocks are dropped. Figure 2 shows an illustrative example: a branch appears after `Block 1` since both `Block 2` and `Block 2′` are pointing to `Block 1`. In this example, if the chain of blocks `0` ← `1` ← `2′` ← `3` is finally the longest chain, `Block 2` is dropped.

**Incentives.** To incentivize the miners, the system allows them to obtain incentives, which include the transaction fees and mining reward. The transaction fees are charged from user transactions, and the mining reward is endowed by the system, which is initially 50 BTCs and halved every 210,000 blocks. The miner who creates the block that is finally kept in the blockchain receives all the incentives, and the other miners get none. Since these incentives are of high economic values (1 Bitcoin equals about $8,000 as of May 2019), the miners tend to optimize their mining strategy to maximize the received financial benefits, often disregarding the potential performance and security implications to the whole Bitcoin system.

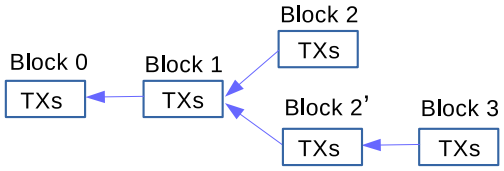


Fig. 2: An illustration of block conflicts.

### C. Transaction Confirmation

**Double spending.** When a block is dropped due to block conflicts, all the transactions included in the dropped blocks are reversed. This potentially results in a *double spending* problem. Let us consider a simple scenario:

At the time `Block 1`, a consumer purchases a product from a vendor and pays the fund to the vendor with a transaction `TX` spending a coin `C`. After observing that `TX` has been included in `Block 2`, the vendor confirms with the consumer that the fund is received and sends the product to the consumer. However, if a block conflict shown in Figure 2 happens and `Block 2` including `TX` is dropped, a double spending problem could happen. Since the transaction is revoked, the vendor loses the product without receiving the payment; the consumer gets the product without paying anything and can spend `C` again.

Besides accidental block conflicts, attackers may intentionally launch a block race for the purpose of dropping the blocks created by honest miners [7], [8], [9], [31], causing the related transactions to be reversed and exacerbates the double spending problem further.

**The number of confirmations.** To address this problem, the receiver has to wait for a certain number of *confirmations* before confirming with the sender that the fund has been received. A transaction initially has zero confirmation when submitted. After it is included in a block (e.g., `Block 2′`), it has one confirmation. Each time when a subsequent block is added to the longest chain containing `Block 2′`, the number of confirmations of the transaction is incremented by one. In Figure 2, for example, the transactions in `Block 1` have three confirmations while the transactions in `Block 3` have only one confirmation.

Obviously, a block that has more subsequent blocks is more likely to be kept in the longest chain, thus demanding a high number of confirmations is effective to reduce the probability of a successful double spending attack. Based on the estimation given by Satoshi Nakamoto's white paper [1], when the attacker has 10% of the hashrate of the system, increasing the number of confirmations from 1 to 6 can reduce the probability of double spending from 20.5% to 0.024%. However, since it takes about 10 minutes to generate a block on average in the Bitcoin system,

a large number of confirmations requires a long waiting time for finalizing the transaction. As a rule of thumb, six-confirmation is commonly considered to be a standard choice (i.e., about 1 hour for completing a transaction).

## III. Methodology

### A. Data Collection and Processing

Our analysis is based on the Bitcoin ledger data, which is publicly available. To retrieve the ledger data, a straightforward method is to run a full Bitcoin client and synchronize with its peers. The ledger data can also be downloaded from public websites [32] and forums of the Bitcoin community [33]. As for data parsing, many Bitcoin clients [34] and websites [17], [35], [36] provide APIs for users to query transaction details. In this paper, we analyze the ledger which contains the transaction history from January 3rd, 2009 to April 30th, 2018, including 520,683 blocks and 313,586,424 transactions. For convenience, we use the APIs provided by blockchain.info [35], [37], which is one of the most popular public websites for providing data analysis of the Bitcoin ledger data. Besides the basic APIs provided by the website, we have also developed homemade tools to parse the ledger (e.g., decoding the transaction scripts) and conduct quantitative analysis.

### B. Time Scale for Data Analysis

To study the effect of the critical mechanisms of the Bitcoin system over time, an important issue is the time scale for analysis. In the Bitcoin ledger data, the only physical time information is the timestamps included in each block to indicate the block generation time in the UNIX format [38]. Due to clock drift or timestamp hacking [39], the timestamps declared by miners may be inaccurate. The accepted timestamps are limited in a certain range, i.e., larger than the median timestamp of previous 11 blocks and smaller than the network-adjusted time plus two hours. Thus, a reported timestamp could deviate from the real block generation time for about two hours [40]. To offset the effects of such a time variance in our study, we take one month as the basic time unit for data analysis.

## IV. Processing Efficiency

To incentivize the miners for processing transactions, the system allows the miners to collect the incentives, including transaction fees and mining rewards (see Section II-B). Our study reveals that the policy and strategy taken by the miners only focus on maximizing their benefits and can even degrade the already-low system performance. In this section, we present our findings on the observed behavior of miners.

### A. Fee-rate-based Prioritization Policy

**Observation #1**: The miners adopt a fee-rate-based policy to prioritize transaction processing, which is biased against low-fee-rate transactions and may freeze small-value coins and degrade processing performance.

Transaction fees are an important incentive provided by the Bitcoin system for miners to process transactions. Currently, the miners adopt a *fee-rate-based prioritization policy*: The *fee rate* of a transaction refers to the transaction fee per size unit (i.e., per Byte). The miners prioritize transaction processing based on fee rates rather than the absolute value of transaction fees, because the block size is limited. First choosing the transactions that pay a higher fee rate for processing allows the miners to collect a higher total amount of fees from generating a block, which maximizes their financial gains. Such a prioritization policy benefits the miners, but is *biased* against low-fee-rate transactions.

A low fee rate affects the chance of a transaction to be processed timely. Figure 3 shows the 1st, 50th, and 99th percentiles of transaction fee rates (in Satoshis per Byte) from 2012 to 2018. Transactions in earlier years are not included here, because most of them are coinbase transactions with zero transaction fees. Since the miners prioritize transaction processing based on fee rates, the priority for a transaction to be processed is inversely proportional to the percentile of its fee rate among all the transactions. For example, if a transaction is only willing to pay a low fee rate (e.g., the bottom 1%), it is very likely to be processed behind 99% of the transactions. Figure 3 shows that the transaction fee rates fluctuate over time. The top 1% pays over 100 times higher than the bottom 1%. In particular, the bottom 1% transaction fee rates have recently reduced to about 1 Satoshi per Byte, compared to over 45 Satoshis per Byte in 2017. This low rate is already close to the minimum fee rate (1 Satoshis per Byte) set by default at the release of Bitcoin Core 0.15 [41]. We also find that a few transactions paying lower than the minimum fee rate have still been processed. These transactions may have been created by a miner, or been processed by the miner who neglected to set the minimum fee rate.

A severe consequence of the fee-rate-based prioritization policy is that some small-value coins may be "frozen", i.e., never have a chance to be spent. This is because a coin may carry a small value that cannot pay off the transaction fee to spend itself. To understand such an effect, we create a simple model to determine the size of the transaction spending one coin, as follows.

Since a transaction contains a list of inputs and a list of outputs, we can model a transaction as `x-y`, in which `x` denotes the number of its inputs and `y` denotes the number of its outputs. From the perspective of coin generation and spending, the model `x-y` means that the transaction spends `x` coins and generates `y` coins (Figure 4 shows the distribution of transactions using the `x-y` model). Based on the transaction model, we find by curve fitting that the transaction size (in Bytes) can be modeled with a two-dimensional linear function in terms of its number of inputs and outputs: $f(x, y) = 153.4 \times x + 34 \times y + 49.5$, in which $x$ denotes the number of the inputs, and $y$ denotes the number of the outputs. The coefficient of determination of the curve fitting (i.e., $R^2$) is 0.91, which indicates that our model has a good fit. Since spending one coin is most likely to involve one input and at most three outputs (see percentages of transactions in models `1-1`, `1-2`, and `1-3`

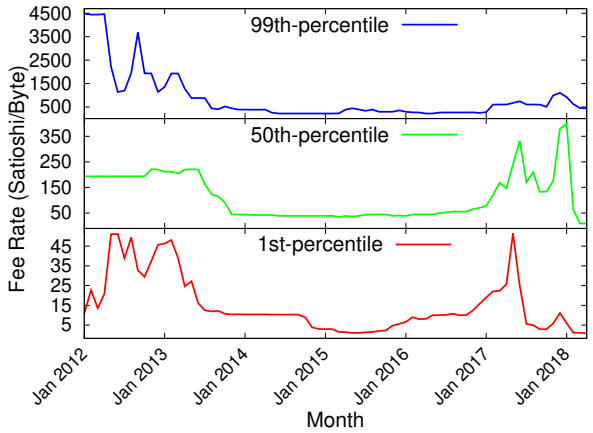


Fig. 3: Transaction fee rates.

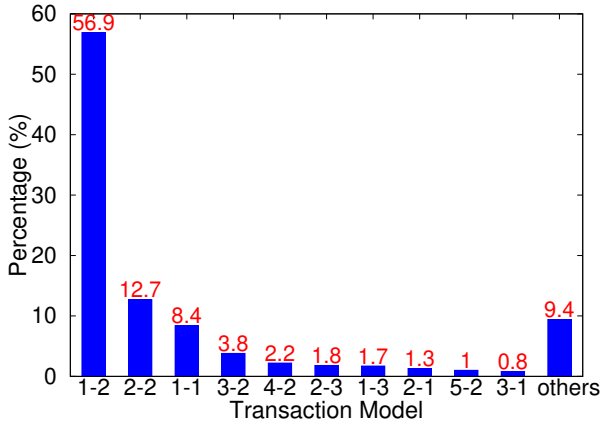


Fig. 4: Transaction model.

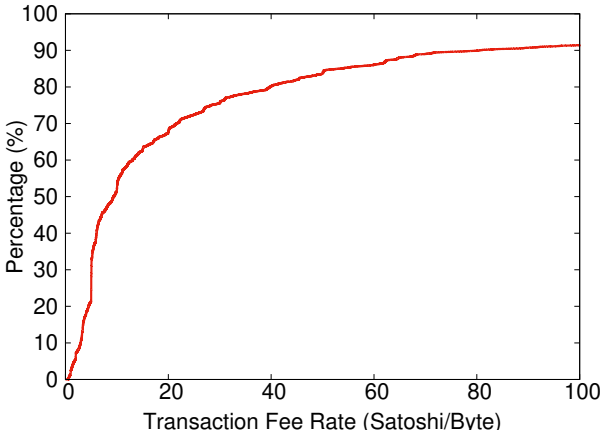


Fig. 5: CDF of fee rates (April 2018).

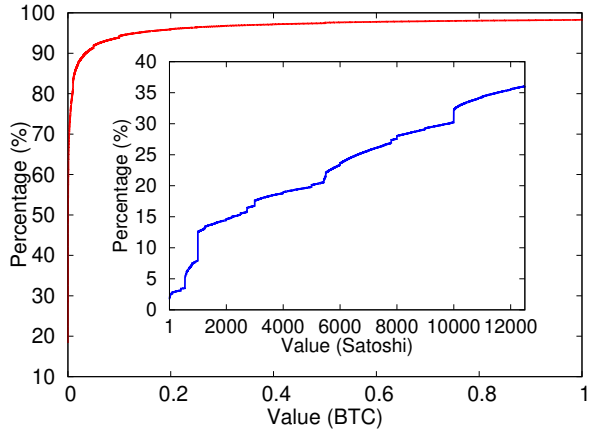


Fig. 6: CDF of coin values.

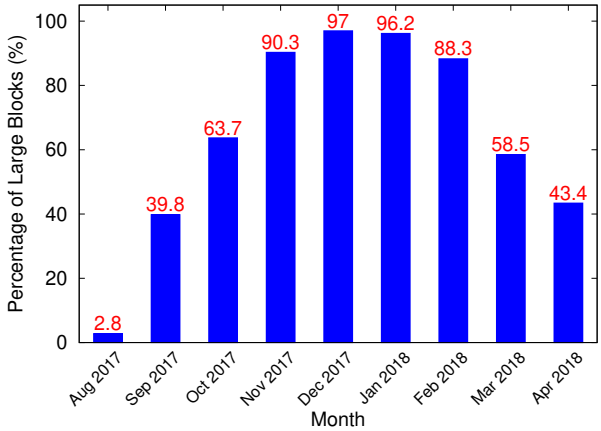


Fig. 7: Percentage of blocks larger than 1 MB.

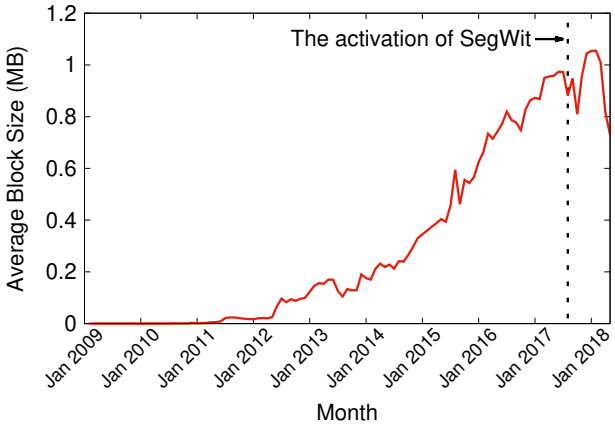


Fig. 8: Average block size.

in Figure 4), based on our transaction size model, the size of a transaction spending one coin is between 237 Bytes and 305 Bytes.

We further calculate the transaction fees for spending a coin by using the transaction fee rates as of April 2018, as shown in Figure 5. If using the minimum fee rate set by Bitcoin Core 0.15 [41], i.e., 1 Satoshi per Byte, the corresponding transaction fee would be 237 to 305 Satoshis. In Figure 6, we show the CDF of the values of coins. About 2.97% of the coins have a value less than 237 Satoshis, and 3.06% of the coin values are below 305 Satoshis. If using the 50th percentile (i.e., the median) of the fee rates, 9.35 Satoshis per Byte, as a reference, 15%-16.6% of the coins cannot afford the processing fee. It is worth noting that a competitive fee rate may be much higher than the median of the fee rates. For example, setting the fee rate to the 80th percentile of all the fee rates, i.e., 40

Satoshis per Byte, can gain a processing priority higher than 80% of the transactions (see Figure 5). However, 30%-35.8% coins cannot afford such a high fee, as shown in Figure 6.

In short, the current fee-rate-based prioritization policy taken by the miners is unfriendly to small-value coins, since they may even not afford the transaction fees for spending themselves. Even worse, a non-trivial portion of coins cannot afford the minimum fee rate. These coins, in effect, will be "frozen", either being not spendable or having to wait for an excessively long time to be processed. Another side effect of the frozen coins is that such coins are not spendable but still reside in the UTXO set (i.e., the database used in the Bitcoin system for coin management), which may slow down the database accesses and potentially reduce the transaction processing speed.

### B. Competition-driven Packing Strategy

**Observation #2**: To win the block competition for incentives, the miners prefer to group a relatively small number of transactions into a block, which further undermines the transaction processing speed.

In the Bitcoin system, a block is the basic unit for processing transactions. The miners verify and group the transactions into blocks and solve a cryptographic puzzle called *Proof-of-Work (PoW)* [1] to gain the right of adding the block to the blockchain and obtaining the incentives.

The transaction processing speed is largely determined by two block-based system protocols: block generation rate and block size limit. Since the block generation rate is controlled to be 10 minutes per block on average by the system, the block size becomes a critical factor that determines the transaction processing performance of the system. Bitcoin Core [34] explicitly set the block size limit to 1 MB in 2013.

To improve transaction processing performance of the system, a soft fork, SegWit (Segregated Witness), was activated on August 23rd, 2017. It separates the witness data (transaction signatures) from the block, which allows to bypass the 1 MB size limit and virtually enlarges the maximum block size to 4 MB [42].

However, according to the transaction history, we find that increasing the block size limit does not necessarily increase actual block sizes. Shown in Figure 7 is the percentage of the blocks larger than 1 MB in all blocks. Interestingly, in the first six months after the SegWit protocol was activated in August 2017, the ratio of large blocks increased from 2.8% to 97%; however, after reaching the peak, the ratio dropped in the following four months from 97% to 43.4%. Not as expected, the SegWit protocol did not increase the actual block size—the average block size dropped to 0.73 MB in April 2018, which is even lower than the average block size in July 2017 (0.88 MB) as shown in Figure 8, the month before the activation of the SegWit protocol.

That is because a large block size limit simply allows the miners to create a large block, but this is not mandatory. The miners can make free decisions whether to process transactions and how many transactions to be grouped in one block. Due to the longest-chain protocol, only the miners whose blocks are finally kept in the blockchain can obtain the incentives, while the others get none. Such a "winner-takes-all" rewarding mechanism drives the miners to take a strategy of generating a block quickly rather than organizing a large block each time. Since it takes a longer time to process more transactions and to broadcast a larger block over the network, generating a larger block comes with a higher risk of losing the competition for a miner to successfully merge the produced block into the main blockchain. From the miners' perspective, in order to win an edge in the competition, they tend to process less transactions and generate a smaller block quickly.

Therefore, the processing strategy taken by the miners are highly profit-driven, which not only degrades the already-low transaction processing performance, but also limits the efficacy of improving the transaction processing performance by setting a large block size limit.

### C. Summary and Remarks

The policy and strategy taken by miners reflect that the miners are rational and profit-driven. To maximize their economic gains, they prefer to process the transactions that pay the highest fee rates and tend to quickly organize a relatively small number of transactions into a block to win the competition. This is the basic rationale behind the success of the Bitcoin system, but as a peer-to-peer distributed system, it inevitably degrades the already-low system performance and may impede the evolution of the Bitcoin system towards being a more efficient system. This would also further lead to other issues. For example, if the Bitcoin system's transaction processing performance remains low, it would force the users to make risky decisions to trade security for speed, as we will see in Section V.

## V. Transaction Confirmation

Due to the longest-chain protocol, the latest several blocks added to the blockchain may be dropped, and the transactions included in those blocks may be reversed. To avoid the double spending problem, the Bitcoin system recommends users to finalize a transaction after waiting for a certain number of confirmations (see Section II-C). In this section, we estimate the number of transaction confirmations determined by users in practice, and present the related observations.

### A. Estimation and Classification

Since the number of confirmations to finalize a transaction is solely determined by the sender and receiver and is transparent to others, we cannot directly obtain the accurate number of confirmations by simply parsing the ledger data. To address this challenge, we propose a method to determine the maximum number of confirmations that a transaction could receive (i.e., the upper bound) based on the fact that a coin can only be spent after the transaction generating that coin has been confirmed and finalized. The methodology is described as below:

Assuming that a transaction generating $n$ coins $C_0$, $C_1$, ..., $C_{n-1}$ of which $m$ coins are spent, the transaction is included in

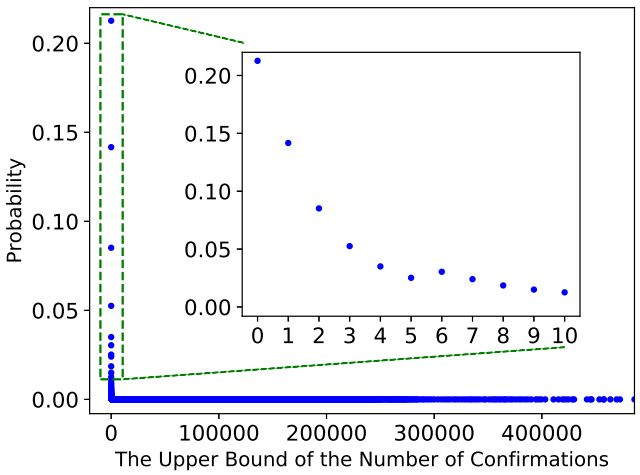

Fig. 9: PDF of the estimated number of confirmations.

Block $G$, and the transactions spending the coins are included in Blocks $B_0$, $B_1$, ..., $B_{m-1}$, respectively, the earliest time that the receiver spends the coins is $S = min(B_0, B_1, ..., B_{m-1})$. Since the receiver can only spend the funds after confirming with the sender for receiving the funds, the latest time that the receiver confirms the completion of the transaction would be at Block $S$. Hence the maximum number of confirmations being possibly received is $N_{conf} = S - G$. A special case is that a block contains both the transactions generating and spending a coin. In this case, $N_{conf} = 0$, meaning that the receiver finalizes the transaction with zero confirmation. Such a zero-confirmation transaction violates the basic rule for using the Bitcoin system that a transaction should be finalized with at least one confirmation. In reality, however, we have observed a non-trivial amount of zero-confirmation transactions (See Section V-B).

We should note that if all the coins generated by a transaction are not spent (i.e., `m = 0`), we cannot determine the upper bound of the number of received confirmations for the transaction. However, such transactions account for less than 1% of all the transactions we have analyzed.

Figure 9 shows the Probability Density Function (PDF) of the estimated maximum number of confirmations that could be received (the upper bound), in which the probability is the ratio of the transactions receiving a certain estimated number of confirmations in all transactions. The PDF is heavy-tailed, following a negative exponential distribution. Obviously, the numbers of confirmations scatter in a wide range (from 0 to more than 0.4 million).

This PDF, to some extent, reflects the randomness of users' trading behaviors. In order to uncover meaningful information from the distribution, we classify the estimated number of transaction confirmations by dividing the range of the numbers into 10 levels based on two rules:

We first select the empirically critical numbers of confirmations as the boundary of several levels. For example, six is the standard number of confirmations. Some users may directly adopt the number of required confirmations (e.g., 1 or 3) set by some Bitcoin wallets as the minimum number of confirmations. Some Bitcoin exchanges, such as Coinbase, require three confirmations [43], and some Bitcoin merchants, such as BitPay, require at least one confirmation [44].

We further estimate the waiting time caused by the required numbers of confirmations and divide the range of waiting time by using the traditional banking systems as a reference. For example, the international wire transfer generally takes about 3 days to 7 days. Considering that the average time for generating one block (i.e., the average time for waiting for one confirmation) is 10 minutes, such a range corresponds to 432 to 1,008 confirmations. Similarly, a domestic wire transfer can generally be completed within one day, corresponding to 144 confirmations. The rest of the range of the required confirmations is divided based on the waiting time in hours (i.e., 2 hours, 6 hours, and 12 hours).

It is worth noting that this classification is based on the estimated upper bound. It does not aim to make an accurate evaluation, but helps us uncover certain meaningful information that can reflect users' preferences on transacting security and efficiency, which will be discussed in the next section (See Section V-B).

### B. Distribution of Confirmation Levels

**Observation #3**: The majority of the users set a small number of confirmations to accelerate the process of transaction finalization, and surprisingly, at least 21.27% of all the transactions are zero-confirmation transactions.

Based on our estimation and classification of the number of confirmations, we have several important findings. Table I shows the ranges and percentages of different levels, and Figure 10 shows the number of the transactions of different levels over time.

First, at least 55.22% of all the transactions are completed with at most five confirmations (from L0 to L2, see Table I). This means that although the recommended and standard setting is 6 confirmations, a large number of users still prefer to set a smaller confirmation number at the cost of transaction security.

Meanwhile, at least 86.2% of all the transactions (from L0 to L6) are completed within 144 confirmations and at least 94.7% of all the transactions (from L0 to L8) within 1,008 confirmations (see Table I). In traditional banking systems, the transaction waiting time for domestic wire transfer is about 1 day (144 confirmations) and for international wire transfer is 7 days (1,008 confirmations). This distribution means that most users do not want to wait for a longer time to complete the transaction on the Bitcoin system than on traditional banking systems, which may also explain why more than half of the transactions are completed with fewer confirmations being received than the standard setting.

A surprising finding is that at least 21.27% of all the transactions are *zero-confirmation* transactions (L0, see Table I). This violates the fundamental rule for trading on the Bitcoin system—at least one confirmation should be received to make sure that the transaction has been verified by the miners and

| Level | Confirmation Num. Range | Waiting Time | Percentage (%) |
|---|---|---|---|
| **L0** | 0 | $<$ 10 min | 21.27 |
| **L1** | [1, 2] | 10 min $\sim$ 30 min | 22.68 |
| **L2** | [3, 5] | 30 min $\sim$ 1 hour | 11.27 |
| **L3** | [6, 11] | 1 hour $\sim$ 2 hours | 11.14 |
| **L4** | [12, 35] | 2 hours $\sim$ 6 hours | 10.40 |
| **L5** | [36, 71] | 6 hours $\sim$ 12 hours | 4.82 |
| **L6** | [72, 143] | 12 hours $\sim$ 1 day | 4.60 |
| **L7** | [144, 431] | 1 day $\sim$ 3 days | 5.35 |
| **L8** | [432, 1,007] | 3 days $\sim$ 1 week | 3.18 |
| **L9** | [1,008, $\sim$) | $>$ 1 week | 5.29 |

TABLE I: Classification of confirmation numbers.

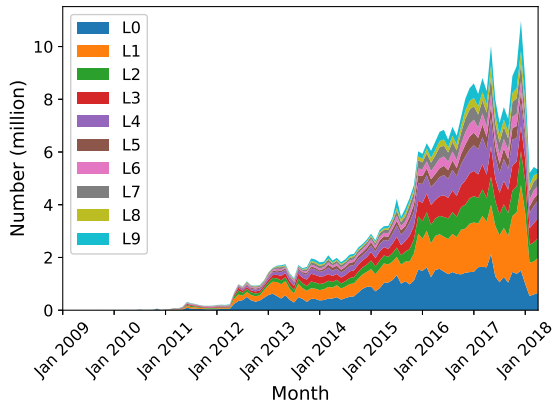


Fig. 10: Breakdown of transactions over time.

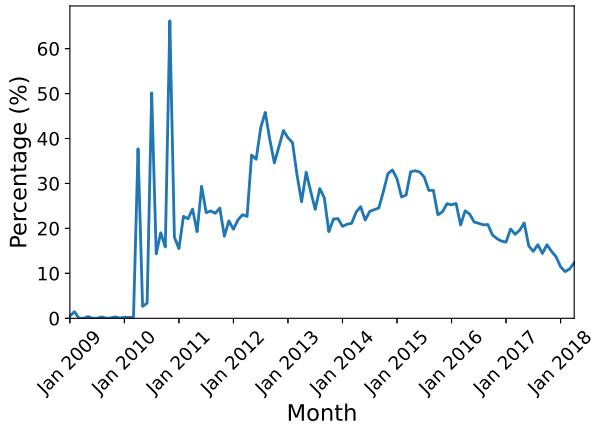


Fig. 11: Percentage of zero-confirmation transactions.

written to the blockchain. Zero-confirmation transactions are not confirmed, being rather vulnerable to attacks.

We initially expected that the zero-confirmation transactions were carried out for transferring a small amount of funds, especially in the early years when the exchange rate between BTCs and USDs was low. However, as shown in Figure 10, though fluctuating over time, the number of zero-confirmation transactions (L0) remains high. We also investigate the value of the funds transferred with zero-confirmation transactions in BTCs and USDs, respectively (the realtime exchange rate between BTCs and USDs is cited from a popular website monitoring the BTC market [45]). One may expect that such unsecure transactions are only used for transacting a small amount of funds. However, we find that the value of the transferred funds of a single transaction can be as high as 0.45 million BTCs or 334 million USDs. This is out of our expectation.

To understand the reason behind creating such high-risk transactions, we further investigate the Bitcoin addresses of the coins spent and generated in the zero-confirmation transactions. We find that about 36.7% of all the zero-confirmation transactions have at least one Bitcoin address used by both the spent coins and generated coins. This means that many users might transfer funds between their own Bitcoin addresses. In this scenario, if the transactions are reversed, the users do not lose anything, and thus they do not concern about the transaction security. We note that such transactions account for 46% of the funds in BTCs or 61.1% in USDs in all the funds transferred by zero-confirmation transactions. A purpose of creating such transactions is to protect the funds by using different cryptographic keys and to help improve the transacting privacy. Since the funds are transferred between the addresses owned by the same user, the reversion of such transactions will not cause fund loss. Thus, it is not necessary to set a high number of confirmations in this case.

Another reason is that some zero-confirmation transactions are created in an unintentional or intentional manner. Some users are not familiar with the Bitcoin system and its confirmation mechanism and thus mistakenly carry out such highly vulnerable transactions. Figure 11 shows the percentage of zero-confirmation transactions over time. From the figure, we can observe that the percentage of zero-confirmation transactions can be very high in early years (e.g., 66.2% in November 2010 and 45.8% in August 2012), but has gradually decreased since 2015. This is largely consistent with our expectation that users gain more knowledge about the confirmation mechanism of the Bitcoin system over time, which contributes to the decrease of zero-confirmation transactions. Meanwhile, it is also possible that some other users can understand the confirmation mechanism, but they simply believe the double-spending attacks can rarely happen in practical scenarios and thus intentionally create zero-conformation transactions for the purpose of reducing the confirmation waiting time.

Finally, some zero-confirmations might be transactions generated to forge a false impression of the wide acceptance of Bitcoin in the market to the general public. In particular, we find that 8,1462 zero-confirmation transactions have the same addresses for the spent coins and the generated coins, indicating that the funds are transferred between the same addresses. These transactions are not sensible and cannot bring any benefits for regular users. They might be transactions created by Bitcoin proponents with investment interests (e.g., Bitcoin investors and gambling websites).

| Script Type | Number | Percentage (%) |
|---|---|---|
| **P2PK** | 1,581,435 | 0.185 |
| **P2PKH** | 732,721,201 | 85.82 |
| **P2SH** | 111,151,759 | 13.02 |
| **OP_Multisig** | 576,205 | 0.067 |
| **OP_RETURN** | 5,234,468 | 0.613 |
| **Others** | 2,519,011 | 0.295 |

TABLE II: Transaction script types.

### C. Summary and Remarks

Based on our observations, although the Bitcoin system is a globally distributed electronic cash system supporting international fund transfer, many users still prefer to have the transaction completed in a short period of time. To reduce the time of waiting for transaction confirmations, many users choose to set a smaller number of required confirmations, which decreases the transaction completion time but sacrifices the transaction security. Such a trading behavior is based on trusting the miners, assuming that the miners are benign players. However, such an assumption is questionable, because the Bitcoin system allows any miner to join or leave the system at will. As the hashrate in the system has centralized to several largest mining pools (the first five largest mining pools have about 60% of all the hashrate of the system [46]), blindly trusting the miners is a risky decision.

## VI. Implementation Flexibility

The Bitcoin system provides a scripting language to allow users to customize transaction implementations. In this section, we present our observations on how real-world Bitcoin users utilize the scripting language for implementing transactions and the potential risks behind such a highly flexible mechanism.

### A. Distribution of Transaction Types

**Observation #4**: About 99.71% of all the transactions are standard transactions, and the flexibility of customizing transactions is rarely utilized.

Bitcoin transactions are classified by the locking scripts in transaction outputs. In the ledger data that we have analyzed, there are 853,784,079 locking scripts. We decode these scripts and show the results in Table II. The five standard transaction scripts, i.e., P2PK, P2PKH, P2SH, OP_Multisig, and OP_RETURN, account for 99.71% of all the transaction scripts.

P2PKH is the most popular transaction type, accounting for 85.82% of the transaction scripts. P2PK is similar to P2PKH, but it directly embeds the public key in the locking script. P2PK is used in the early stage of the Bitcoin system, but is currently obsolete.

Another popular transaction type, P2SH, accounts for 13.02% of the transaction scripts. P2SH offloads the responsibility of implementing verification methods from the sender to the receiver. The receiver is responsible for preparing a script for verification. The sender only has to specify the receiver in the locking script (by using the receiver's Bitcoin address). Due to such advantages, P2PH is becoming popular.

OP_Multisig and OP_RETURN are also standard transaction types, but are used for particular purposes. OP_Multisig requires one or more signatures out of multiple public keys to improve fund security. OP_RETURN allows users to store arbitrary data of limited length (initially 40 Bytes and currently 80 Bytes [18]) in the locking script. These two types of transaction scripts account for about 0.68% of all the transaction scripts in total.

In addition to the standard transactions, we find only a small portion, about 0.295%, of transaction scripts are not in the five standard script types. The non-standard transactions are largely implemented manually by technology enthusiasts, Bitcoin software developers, or hackers.

Based on the distribution of transaction types, we can see that most Bitcoin transactions are standardized, despite that the highly customizable and flexible scripting interface is provided by the system. A main reason is that most users are non-experts. It is difficult for regular users to understand and compose the scripts correctly. Comparatively, it is more convenient to use Bitcoin wallets to execute transactions and manage their coins.

### B. Erroneous and Harmful Transactions

**Observation #5**: The flexibility of transaction customization exposes the system and users to erroneous, inappropriate, and even dangerous transactions, which can cause fund loss or even be harmful to the system.

In the transaction scripts that we have studied, we note that 252 scripts among the non-standard transaction scripts are erroneous. These scripts cannot be correctly decoded according to the defined scripting language [25]. This may be caused by implementation errors.

Some transactions can be correctly decoded, but are not implemented in a semantically proper manner. Most of these transactions involve OP_RETURN transactions, OP_Multisig transactions, and coinbase transactions. Here we discuss four main types of implementation errors in the transactions.

**Erroneous value settings**: OP_RETURN scripts are expected to be associated with zero values, but 56,695 OP_RETURN scripts are mistakenly associated with nonzero values. OP_RETURN scripts allow users to store a small piece of arbitrary data and store it in the Bitcoin system. These scripts are not supposed to be associated with any value because these values are not spendable, meaning that associating nonzero values with the OP_RETURN scripts is only a waste of money, without benefiting any miners or users.

**Improper use of opcodes**: OP_Multisig scripts are used for multiple users to share the possession of a coin and are expected to require the signatures of multiple public keys. However, we find that 2,446 OP_Multisig scripts involve only one public key. It is worth noting that these scripts are not grammatically wrong, but they are not used in a proper scenario. Such scripts have the same functionality as P2PK scripts, but have a larger size and thus need to pay a higher transaction fee [47].

**Redundant opcodes**: Three transaction scripts contain an unreasonably large number of opcodes. The Bitcoin system allows users to customize transactions, but this also opens opportunities for creating potentially harmful transactions. We find three "suspicious" transaction scripts. These scripts are similar to P2PKH scripts, but contain as many as 4,002 OP_CHECKSIG opcodes. In contrast, normal P2PKH scripts have only one OP_CHECKSIG. The OP_CHECKSIG opcode is used for checking the validity of the transaction signature. It is not a sensible implementation to include multiple OP_CHECKSIG in a P2PKH locking script to repeatedly verify the same signature. These three scripts containing thousands of OP_CHECKSIG only waste the computing resources of the miners, being harmful to the system.

**Wrong rewards settings**: Two coinbase transactions are associated with wrong mining rewards. Coinbase transactions are created by miners to send the mining rewards to themselves. It is surprising that two coinbase transactions do not set correct mining rewards. Specifically, mining `block 124,724` should receive 50 BTCs, but 49.99999999 BTCs is set in the coinbase transaction; mining `block 501,726` should receive 12.5 BTCs, but 0 BTC is set in the coinbase transaction. Such errors cause fund loss to the miners, especially for the latter case. The date of mining block 501,726 is December 30th, 2017 and the price of 1 BTC was $12,630 at that time [45]. The miner lost 12.5 BTCs due to this mistake, which is about $157,875, a non-trivial amount.

The erroneous and harmful implementations of the transactions indicate that the flexibility of customizing transactions is provided to facilitate users, but may bring negative effects: for the users who do not have sufficient knowledge for transaction implementation, the erroneous and inappropriately implemented transactions would cause fund loss; for the malicious users, they may exploit the scripting language to make harmful transactions.

### C. Summary and Remarks

Our observations show that most users show low interests or capability of implementing customized transactions. Such an unnecessary flexibility may lead to undesirable consequences, e.g., erroneous and harmful transactions. Compared to the flexibility of implementing transactions, the convenience of implementing transaction is a more important concern for common users. In practice, Bitcoin wallets can automatically implement transactions based on the transacting information provided by users. However, such a choice offloads the responsibility of implementing transactions to Bitcoin wallets, making the security of transacting largely depend on Bitcoin wallets. Since many Bitcoin wallets are third-party applications provided by centralized services, heavily relying on Bitcoin wallets weakens the decentralization of Bitcoin services and creates security risks—implementation errors, such as bugs in the beta version of some Bitcoin wallets, can quickly spread and cause system-wide damage.

## VII. Discussions

In previous sections, we have presented our observations on the behaviors of real-world miners and users. Our findings have reflected how these behaviors deviate from the original design purposes of the Bitcoin system. In this section, we discuss several prospective methods to optimize the design and implementation of future cryptocurrency systems.

### A. Bitcoin Variants

Following Bitcoin, more than 2,000 Bitcoin-like cryptocurrencies have been released. These cryptocurrency systems deeply mimic the Bitcoin system and some are tuned to overcome some limitations of the original Bitcoin system. In particular, many systems increase the block size limit to increase transaction processing efficiency. In Table III, we can see that most of the major forks of Bitcoin adopt an enlarged block size limit. A notable example is that Bitcion Cash [52], [53] increases the block size limit to 32 MB.

An assumption behind such forks is that a larger block size limit allows to process more transactions in a block and will lead to a higher transaction processing speed. However, according to our observations on Bitcoin, the miners prefer to organize small blocks to win the mining rewards regardless of the block size limit. Thus, such an assumption is intuitively true but practically invalid. Since the Bitcoin forks mostly inherit the same rewarding mechanism of Bitcoin, we can infer that some of our findings are also applicable to such systems. In other words, simply increasing the block size limit will not help incentivize miners to organize large blocks and consequently would not improve system performance. Our inference has been confirmed with observations on some of the Bitcoin forks. For example, though setting the block size limit to 32 MB, the reported average block size of Bitcoin Cash is much smaller than 1 MB [57].

In addition to Bitcoin forks, some Bitcoin-like systems have also attempted to increase system performance by tuning system protocols. For example, Ethereum [58] changes the structure of the blockchain from a singly linked list to a directed acyclic graph (DAG). This method keeps all the branches of the blockchain, and the miners who create the blocks that do not reside in the longest chain can still obtain certain rewards. This is effective to alleviate the block competition and improve system performance, but cannot address problems caused by the overly provided freedom of the miners, such as the frozen coin problem, because these methods do not essentially change the right of the miners to pick transactions, and the users still do not have the right to evaluate and control the quality of the services provided by the miners.

### B. Next-generation Blockchain Systems

To overcome the limitations of the Bitcoin-like systems, especially for the purpose of utilizing the blockchain technology in scenarios other than cryptocurrencies, simply changing system parameters is not enough. We need to fundamentally reconsider the roles of miners and users for improving system performance and scalability. Based on our findings, we discuss

| Year | Project Name | Fork Type | Block Size Limit | Current Status |
|---|---|---|---|---|
| **2009** | Bitcoin [1] | The original system | initially no explicit limit, later 1 MB | Active |
| **2014** | Bitcoin XT [48] | Hard fork | 8 MB (doubling every two years) | Inactive |
| **2016** | Bitcoin Classic [49] | Hard fork | 2 MB (this value can be customized) | Inactive |
| **2016** | Bitcoin Unlimited [50] | Hard fork | 16 MB (the value can be customized) | Inactive |
| **2017** | SegWit [51] | Soft fork | virtually 4 MB | Active |
| **2017** | Bitcoin Cash [52], [53] | Hard fork | initially 8 MB, currently 32 MB | Active |
| **2017** | Bitcoin Gold [54] | Hard fork | 1 MB | Active |
| **2017** | SegWit2x [55] | Hark fork | 2 MB | Cancelled |
| **2018** | Bitcoin Private [56] | Hark fork | 2 MB | Active |

TABLE III: The history of the Bitcoin system and its major forks. This table lists the major forks of the Bitcoin system. A hard fork is a new version of the system, which changes the rules of the original system and requires all participating nodes to upgrade and agree on the new rules. A soft fork is a backward compatible modification to system protocols, in which the upgraded nodes follow the new rules and the non-upgraded nodes continue to follow old rules. This table shows that increasing the block size limit is an important factor in Bitcoin forks for improving transaction processing efficiency.

two possible directions that the next-generation blockchain systems could evolve.

**Evolution Direction 1: User-determined rewarding mechanism.** With the current rewarding mechanism, the policy and strategy taken by miners could degrade system performance. The root cause for such a phenomenon is that the incentive that motivates the miners to process a transaction comes from the system by solving a cryptographic puzzle (i.e., PoW [1]), rather than the feedback of users.

In order to incentivize miners to process transactions in a user-oriented manner, we need to change the role of users in the system: the users should not only "use" the Bitcoin system; they should have certain capability to determine which miners can process transactions and gain the incentives. For example, the users can rank a miner based on its processing history. The miners who only process high fee-rate transactions and create small blocks will be given a low ranking and voted out of work. This can effectively restrict the freedom of the miners for addressing the frozen coin problem (see Section IV) and improving performance. By using such a user-determined rewarding mechanism, the users can have ability to evaluate and control the quality of Bitcoin services.

In practice, the user-determined rewarding mechanism should not be as simple as some online review systems, e.g., Amazon Marketplace. We may need additional mechanisms to prevent users from abusing their rights and to differentiate the weights of different users' votes. For example, in some protocols that aim to address the limitations of Bitcoin, such as *Delegated Proof of Stake (DPoS)* [59], the voting power of a user is proportional to the number of coins that are held by the user. The rationale behind it is that the users who own more coins have more interest in maintaining the entire system efficient and stable. Although how to create an efficient and secure voting system still deserves further study, we believe that the emerging DPoS-like protocols have reflected the evolution direction towards limiting the freedom of miners and giving more rights to users.

**Evolution Direction 2: Limiting the overly provided flexibility.** Considering Bitcoin is mostly used for fund transfer, its main usage is to support a limited number of standard transaction types. To avoid implementation errors and scripting attacks, we can define strict scripting grammars by putting necessary constraints to the overly provided flexibility of customizing transactions. Meanwhile, the miners should also take responsibility for grammar checking. Together with the above-mentioned user-determined incentivizing mechanism, the miners that do not correctly report implementation errors can be given a low credit and have less opportunities to process transactions and obtain the incentives.

Moving towards the next-generation blockchain systems, some new protocols for decentralized systems (e.g., DPoS [59]) have been proposed. DPoS fundamentally changes the underlying protocols of Bitcoin-like systems (e.g., PoW-like protocols) and allows users to evaluate and select miners. This adjustment of the provided freedom in the system would encourage the miners to provide more user-centered services. Due to its potential advantages, DPoS has been adopted by some blockchain systems (e.g., EOS [60], BitShares [61], Lisk [62]). Facebook's recently announced digital currency, Libra, also adopts a protocol sharing some similarities with DPoS in terms of limiting the freedom of validators [63]. Our findings indicate that the DPoS-like protocols that give more rights to users and constraint the freedom of miners might be promising in the future.

### *C. Other System Implications*

In addition to changing the protocols and rules of the Bitcoin system, our observations also provide guidance for optimizing system implementation. For example, we find that due to the fee-rate-based prioritization policy, small-value coins may be "frozen". This effect may degrade the performance of the coin database (i.e., LevelDB [64] in the current implementation). To improve the performance of the database, we can design a value-aware optimization. For example, the records of small-value coins can be given a low caching priority and stored in low-performance storage devices. This method can separate active coins from frozen coins and is thus helpful to improve the access efficiency of the database.

Another optimization is to reduce the possibility of generating small-value coins with better coin selection algorithm used

by Bitcoin clients or wallets. When selecting the coins to pay the target fund (the transfer amount plus the transaction fee), *Bitcoin Core* [34] always attempts to select the coins that have the smallest value to satisfy (be equal to or larger than) the target [65]. Such a coin selection algorithm helps minimize the number of change coins (to receive the difference between the value of the coins and the target), but may generate many small-value coins. Our finding suggests that the coin selection algorithm should be optimized to avoid generating small-value coins.

## VIII. Related Work

In this section, we present the prior works that are most related to our work. Many of prior studies focus on investigating the Bitcoin system and its variants in terms of performance, scalability, and security, based on theoretical models of the underlying blockchain protocols. Many of these prior studies aim to improve the Bitcoin system by adjusting critical protocol parameters [3], [4], [5], optimizing system implementation [6] or proposing new system protocols [66]. Another focus is to investigate potential attacks on the Bitcoin system, such as double spending attacks [7], selfish mining attacks [8], [9], [31], and eclipse attacks [10]. Unlike these prior works, our study focuses on studying the real-world behaviors of miners and users in the Bitcoin system, by quantitatively analyzing the publicly shared ledger data.

Our work is also related to prior studies on the ledger data. Transaction graphs [67], [68], [69], [70] have been widely used to investigate user activities, such as linking Bitcoin addresses to users, and speculating the relations between different users. These studies investigate the user behaviors mostly from the perspective of privacy and anonymity. In particular, Ron and Shamir conducted a quantitative analysis of the ledger data in early years, which primarily focuses on studying how users move their funds between different accounts to protect transacting privacy [69]. By contrast, our work focuses on investigating the behaviors regarding how miners and users take advantage of the provided freedom to achieve their own interests in practice. Our study provides the first-hand observations and insights on the behaviors of miners and users, which also make our work different from the public websites or technical reports that simply present the statistical data (e.g., [57], [17], [18], [19], [20], [21]).

Some other works study the particular use of the Bitcoin system, such as using blockchain to store arbitrary data. These works study the metadata of OP_RETURN transactions [18], the methods of inserting data on the Bitcoin blockchain [20], and the content of the data stored on the current Bitcoin blockchain [21]. We focus on studying the behaviors and practical concerns of miners and users. Our work is also orthogonal to prior works that design tools for analyzing the ledger data [71], [47] or use the ledger data to predict Bitcoin price [72].

In addition to academic research, the blockchain technology, which is originated from the Bitcoin system, has also been utilized in industrial products to provide peer-to-peer services. These blockchain based systems [12], [13], [14], [15] and applications [73], [74], [16] more or less inherit certain properties of the Bitcoin system. Our study is also valuable and can provide guidance to understand the user behaviors of these systems and applications.

## IX. Conclusion

The Bitcoin system enables trusted transactions based on its underlying blockchain technology. It provides a high degree of freedom to both miners and users for achieving decentralized trust. To investigate the behaviors of miners and users, we have analyzed the Bitcoin transaction history of more than nine years. Our study has revealed how the miners and users use the Bitcoin system in real-world deployment, which reflects the gap between system design and user concern. We believe that our study can not only help us clarify the behaviors of Bitcoin miners and users, but also bring many useful insights for future system design and optimization.


## Acknowledgements

We thank the anonymous reviewers for their valuable feedback and insightful comments. This work was partially supported by the U.S. National Science Foundation under Grants CCF-1453705, CCF-1629291, and CCF-1910958.


## References


[1] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System." https://bitcoin.org/bitcoin.pdf.

[2] CoinMarketCap, "All Cryptocurrencies." https://coinmarketcap.com/all/views/all/.

[3] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, E. G. Sirer, *et al.*, "On Scaling Decentralized Blockchains," in *Proceedings of the 20th International Conference on Financial Cryptography and Data Security (FC '16)*, pp. 106–125, Springer, 2016.

[4] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the Security and Performance of Proof of Work Blockchains," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16)*, pp. 3–16, ACM, 2016.

[5] S. Kasahara and J. Kawahara, "Effect of Bitcoin Fee on Transaction-confirmation Process," *arXiv preprint arXiv:1604.00103*, 2016.

[6] Y. Sompolinsky and A. Zohar, "Accelerating Bitcoin's Transaction Processing," *Fast Money Grows on Trees, Not Chains*, 2013.

[7] M. Rosenfeld, "Analysis of Hashrate-based Double Spending," *arXiv preprint arXiv:1402.2009*, 2014.

[8] I. Eyal and E. G. Sirer, "Majority is Not Enough: Bitcoin Mining is Vulnerable," in *Proceedings of the 18th International Conference on Financial Cryptography and Data Security (FC '14)*, pp. 436–454, Springer, 2014.

[9] A. Sapirshtein, Y. Sompolinsky, and A. Zohar, "Optimal Selfish Mining Strategies in Bitcoin," in *Proceedings of the 20th International Conference on Financial Cryptography and Data Security (FC '16)*, pp. 515–532, Springer, 2016.

[10] E. Heilman, A. Kendler, A. Zohar, and S. Goldberg, "Eclipse Attacks on Bitcoin's Peer-to-Peer Network," in *Proceedings of the 24th USENIX Security Symposium (USENIX Security '15)*, pp. 129–144, 2015.

[11] L. Luu, J. Teutsch, R. Kulkarni, and P. Saxena, "Demystifying Incentives in the Consensus Computer," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS '15)*, pp. 706–719, ACM, 2015.

[12] Ethereum. https://www.ethereum.org/.

[13] Hperledger, "Hyperledger Fabric." https://www.hyperledger.org/projects/fabric.

[14] Parity. https://www.parity.io/.

[15] Corda. https://www.corda.net/.

[16] M. Ali, J. C. Nelson, R. Shea, and M. J. Freedman, "Blockstack: A Global Naming and Storage System Secured by Blockchains," in *Proceedings of the 2016 USENIX Annual Technical Conference (USENIX ATC '16)*, pp. 181–194, 2016.

[17] BitcoinExplorer. https://blockexplorer.com/.

[18] Massimo Bartoletti and Livio Pompianu, "An Analysis of Bitcoin OP_RETURN Metadata." http://arxiv.org/abs/1702.01024.

[19] BitcoinTalk, "Bitcoin Forum." https://bitcointalk.org/.

[20] A. Sward, I. Vecna, and F. Stonedahl, "Data Insertion in Bitcoin's Blockchain," *Ledger*, vol. 3, 2018.

[21] R. Matzutt, J. Hiller, M. Henze, J. H. Ziegeldorf, D. Müllmann, O. Hohlfeld, and K. Wehrle, "A Quantitative Analysis of the Impact of Arbitrary Blockchain Content on Bitcoin," in *Proceedings of the 22nd International Conference on Financial Cryptography and Data Security (FC '18)*, 2018.

[22] Huberman G, Leshno JD, Moallemi CC., "Monopoly without a Monopolist: An Economic Analysis of the Bitcoin Payment System." https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3025604.

[23] S. Bose, "How Many Bitcoin Miners Are There?." https://www.btcwires.com/round-the-block/how-many-bitcoin-miners-are-there/.

[24] A. Lielacher, "How Many People Use Bitcoin in 2020?." https://www.bitcoinmarketjournal.com/how-many-people-use-bitcoin/.

[25] bitcoin.it, "Bitcoin Script." https://en.bitcoin.it/wiki/Script.

[26] D. Johnson, A. Menezes, and S. Vanstone, "The Elliptic Curve Digital Signature Algorithm (ECDSA)," *International Journal of Information Security*, vol. 1, no. 1, pp. 36–63, 2001.

[27] A. K. Kasgar, J. Agrawal, and S. Shahu, "New Modified 256-bit MD 5 Algorithm with SHA Compression Function," *International Journal of Computer Applications*, vol. 42, no. 12, 2012.

[28] H. Dobbertin, A. Bosselaers, and B. Preneel, "RIPEMD-160: A Strengthened Version of RIPEMD," in *International Workshop on Fast Software Encryption*, pp. 71–82, Springer, 1996.

[29] Wikipedia, "BASE58." https://en.wikipedia.org/wiki/Base58.

[30] bitcoin.org, "Bitcoin Developer Guide." https://bitcoin.org/en/developer-guide#transaction-fees-and-change.

[31] I. Eyal, "The Miner's Dilemma," in *Proceedings of the 2015 IEEE Symposium on Security and Privacy (SP '15)*, pp. 89–103, IEEE, 2015.

[32] flo.sh, "Download Bitcoin Blockchain." https://flo.sh/bitcoin-qt-bootstrap-dat/.

[33] bitcointalk.org, "Download the Blockchain Here, Updated Regularly." https://bitcointalk.org/index.php?topic=1310261.0.

[34] bitcoin.org, "Bitcoin Core." https://bitcoin.org/en/bitcoin-core/.

[35] blockchain.com, "Bitcoin Block Explorer - Blockchain." https://www.blockchain.com/explorer.

[36] chain.so, "SoChain." https://chain.so/api.

[37] blockchain.com, "Blockchain Data API." https://www.blockchain.com/api/blockchain_api.

[38] Wikipedia, "UNIX Time." https://en.wikipedia.org/wiki/Unix_time.

[39] S. M. English, "Timestamp Hacking: Debunking the Myth of Precision Timestamps." https://cointelegraph.com/news/timestamp-hacking-debunking-the-myth-of-precision-timestamps.

[40] bitcoin.it, "Bitcoin Timestamp." https://en.bitcoin.it/wiki/Block_timestamp.

[41] bitcoin.org, "Bitcoin Core Version 0.15.0 Released." https://bitcoin.org/en/release/v0.15.0#removal-of-coin-age-priority.

[42] bitcoinwiki.org, "SegWit Block Size." https://en.bitcoinwiki.org/wiki/Block_weight.

[43] CoinCenter, "How Long does It Take for A Bitcoin Transaction to Be Confirmed?." https://coincenter.org/entry/how-long-does-it-take-for-a-bitcoin-transaction-to-be-confirmed.

[44] bitpay.com, "Why Your Bitcoin Transactions Are Taking So Long to Confirm?." https://blog.bitpay.com/transaction-delays/.

[45] coindesk.com, "Bitcoin (USD) Price." https://www.coindesk.com/price/.

[46] blockchain.com, "Bitcoin Hashrate Distribution." https://www.blockchain.com/en/pools.

[47] S. Delgado-Segura, C. Pérez-Sola, G. Navarro-Arribas, and J. Herrera-Joancomartı, "Analysis of the Bitcoin UTXO set," in *Proceedings of the 5th Workshop on Bitcoin and Blockchain Research Research*, 2018.

[48] Wikipedia, "Bitcoin TX." https://en.wikipedia.org/wiki/Bitcoin_XT.

[49] Wikipedia, "Bitcoin Classic." https://en.wikipedia.org/wiki/Bitcoin_Classic.

[50] Wikipedia, "Bitcoin Unlimited." https://en.wikipedia.org/wiki/Bitcoin_Unlimited.

[51] Wikipedia, "SegWit." https://en.wikipedia.org/wiki/SegWit.

[52] cryptoground.com, "32MB Bitcoin Cash Hard Fork Planned for May 2018." https://www.cryptoground.com/a/32mb-bitcoin-cash-hard-fork-may-2018.

[53] Wikipedia, "Bitcoin Cash." https://en.wikipedia.org/wiki/Bitcoin_Cash.

[54] Wikipedia, "Bitcoin Gold." https://en.wikipedia.org/wiki/Bitcoin_Gold.

[55] Wikipedia, "SegWit2x." https://en.wikipedia.org/wiki/SegWit2x.

[56] Wikipedia, "Bitcoin Private." https://en.wikipedia.org/wiki/Bitcoin_Private.

[57] bitinfocharts.com, "Bitcoin Cash Size Chart." https://bitinfocharts.com/comparison/bitcoin%20cash-size.html.

[58] Ethereum, "Ethereum White Paper." https://github.com/ethereum/wiki/wiki/White-Paper.

[59] bitcoinwiki.org, "DPoS." https://en.bitcoinwiki.org/wiki/DPoS.

[60] whitepaperdatabase.com, "EOS." https://whitepaperdatabase.com/eos-whitepaper/.

[61] BitShared, "BitShared." https://bitshares.org/.

[62] Lisk, "Lisk." https://lisk.io/documentation/home.

[63] Libra, "An Introduction to Libra." https://libra.org/en-US/wp-content/uploads/sites/23/2019/06/LibraWhitePaper_en_US.pdf.

[64] LevelDB, "LevelDB." https://github.com/google/leveldb.

[65] M. Erhardt, "An Evaluation of Coin Selection Strategies." http://murch.one/wp-content/uploads/2016/11/erhardt2016coinselection.pdf.

[66] I. Eyal, A. E. Gencer, E. G. Sirer, and R. Van Renesse, "Bitcoin-NG: A Scalable Blockchain Protocol," in *Proceedings of the 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI '16)*, pp. 45–59, 2016.

[67] M. Fleder, M. S. Kester, and S. Pillai, "Bitcoin Transaction Graph Analysis," *arXiv preprint arXiv:1502.01657*, 2015.

[68] D. D. F. Maesa, A. Marino, and L. Ricci, "Uncovering the Bitcoin Blockchain: An Analysis of the Full Users Graph," in *Proceedings of the 2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA '16)*, pp. 537–546, IEEE, 2016.

[69] D. Ron and A. Shamir, "Quantitative Analysis of the Full Bitcoin Transaction Graph," in *Proceedings of the 17th International Conference on Financial Cryptography and Data Security (FC '13)*, pp. 6–24, Springer, 2013.

[70] M. Harrigan and C. Fretter, "The Unreasonable Effectiveness of Address Clustering," in *Proceedings of the 2016 International IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*, pp. 368–373, IEEE, 2016.

[71] M. Bartoletti, S. Lande, L. Pompianu, and A. Bracciali, "A General Framework for Blockchain Analytics," in *Proceedings of the 1st Workshop on Scalable and Resilient Infrastructures for Distributed Ledgers*, p. 7, ACM, 2017.

[72] H. Jang and J. Lee, "An Empirical Study on Modeling and Prediction of Bitcoin Prices with Bayesian Neural Networks based on Blockchain Information," *IEEE Access*, vol. 6, pp. 5427–5437, 2018.

[73] CryptoKitties, "Cryptokitties." https://www.cryptokitties.co/.

[74] blockgeeks.com, "Smart Contracts." https://blockgeeks.com/guides/smart-contracts/.