Solutions to the Endless Addition of Transaction Volume in Blockchain

Hongping Cao[1]
School of Data Science
Guangzhou Huashang College
Guangzhou, China

Hongxing Cao[2]
School of Electrical and Computer Engineering
Guangzhou Nanfang College
Guangzhou, China

***Abstract*—In blockchain system, the problem of endless addition of transaction volume results in larger space occupation, heavier network transmission burden and the like. But the way of abandoning historical data is hindered by the characteristic of blockchain-tamper-proof. To solve this problem, this dissertation takes bitcoin system as an example, and gives a definition of expired transaction. By abandoning expired transactions and packing the rest transactions in several blocks into a new substitute block to replace old blocks, help to overcome the difficulty in clearing historical data. However, this solution fails to clear ineffective intermediate transactions. Thus, a follow-up solution is proposed, abandon the transactions whose outputs have been run out and retain the transactions where not all the outputs have been spent. Additionally, include records of the spending details of each transaction output, which allows us to clear ineffective intermediate transactions. In the end, an experiment is conducted to confirm the effectiveness of the two solutions as to clearing transactions.**

***Keywords*—Blockchain; endless addition; expired transactions; substitute; storage problem; consensus algorithm**

I. Introduction

Blockchain [1] technology has become a new data storage technology with its decentralization, anti-tampering, and traceability. In addition to the field of digital cryptocurrency [2], it also enjoys promising prospects for application in the fields of data notarization [3], resource sharing [4] and supply chain traceability [5]. It is estimated that by 2027, 10% of global GDP will be stored through blockchain technology [6].

To realize decentralization and traceability, blockchain technology uses chain storage structure [7] and multiple complete nodes to save data copies in full. However, such a storage method inevitably causes endless addition of data, and its anti-tampering feature means once the data (even the wrong data) is uploaded, it cannot be modified or deleted. As time goes on, fast-growing data of block chain leads to space occupation, which has become a problem that blockchain technology has to face. In addition, when a new complete node joins the system, the full amount of data must be synchronized, resulting in an increase in the network load of other nodes. Currently, it takes several days for Bitcoin nodes to synchronize the full amount of data every time, and every node needs to verify the correctness of the transaction when it receives a synchronization block, which will consume massive CPU computing power.

In the following sections, an overview of solutions in bitcoin system and research field, is given firstly, and find that these ideas merely mediate the effects of the problem instead of solving it. Then we define the expired transactions, based on which gives a solution of re-packing to generate substitute block to replace old blocks safely and thus abandon historical data. But this solution can only clean about 20% transactions. To clean more middle expired transactions, give a follow-up solution by recording signals of unspent transaction outputs. Finally, we verify the correctness and effectiveness of two solutions by experiments to find the second solution can clean about 80% expired transactions. With comparison to traditional solutions, reusing consensus algorithm (Power of Work) to generate substitute block to replace old blocks safely, is a brand-new and effective idea in this field.

II. Solutions in Bitcoin

The most typical application of blockchain technology is Bitcoin. To make things easier, Bitcoin will be taken as an example. Bitcoin has been in operation for 12 years since 2019. As of February 14, 2021, there were 670,540 blocks and 18,628,343 bitcoins, with a total transaction volume of 1,232,212,522, a total data capacity of about 306 GB, a total number of 786,328,292 authentication addresses, and a market capitalization of 905.9 billion dollars [8]. According to statistics from BitNodes [9], there are currently about 10,000 complete nodes online at the same time in the entire network, and the disk space required by a single complete node is about 306 GB. Each complete node stores one copy of data, so the data capacity required by the whole system is about 3.1 PB. At present, it is growing rapidly at the rate of about 330,000 transactions per day [10]. With such high-speed transaction growth, the storage space occupation of Bitcoin has become an increasingly obvious problem.

Satoshi Nakamoto, the author of Bitcoin, once proposed a solution to the storage problem-space recycling. He believes that if the recent transaction has been included in enough blocks, the data before the transaction can be discarded to recover the hard disk space [1]. He pruned the expired transaction data, yet kept their Hash values to ensure that the Merkle root of the block is a verifiable value, as is shown in Fig. 1:

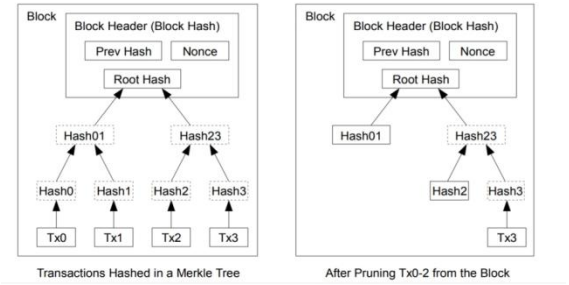

Fig. 1. Before and after Pruning Tx0-2 from Block [1].

The data before the transaction mentioned by Satoshi Nakamoto in that article can be described as follows: For example, Coinbase transaction is transferred from A to B, and then B transfers to C. If the two transactions are included in the block chain and are confirmed by sufficient subsequent blocks (more than six), the legality of the transaction can be confirmed. When the transaction from B to C is confirmed, the transaction from A to B, which is regarded as a transaction before the transaction from B to C, is no longer valuable in verifying the transaction. It only has retrospective value as an expired transaction.

Whether the pruning is correct or not, the block verification is unfailingly correct so long as the hash algorithm is not destroyed. In this regard, verifying the correctness of the pruning result will be a problem. In practice, Bitcoin does not use the above pruning strategies when solving this problem. Instead, it uses the following strategies:

1) Use prune parameters. Prune parameter is used to reduce the usage of local hard disk. When launching Bitcoin core for the first time, the software will require the user to select the folder location where the blockchain is stored. All users need to do is to create a setup document named "bitcoin. conf" under the root directory of this folder, and write the code prune=N, where the unit of n is MB. Among them, N is the blockchain size stored locally, and N=0 means no limitation and complete download; If a limitation is required, the minimum value is 550 (that is, 550MB). It should be noted that even if the -prune mode is started, the node will still need to download the data of the entire blockchain network, but instead of storing all the data, it will simply delete the old blocks after building the UTXO library. In addition, if the prune mode is started, data on the entire blockchain network will have to be downloaded again during re-scanning, and the client usage will be greatly affected.

2) Provide SPV (Simplified Payment Verification) lightweight client. In the SPV method, SPV nodes only store the header information of the block, not the complete data. Therefore, when a transaction occurs and the validity of the transaction needs to be verified, it downloads the latest block from the complete node. When it is verified that the transaction is packed into the block and the number of blocks generated after the block is greater than six, the transaction is thus verified. However, SPV nodes completely depend on complete nodes, and verify transactions by using simple payment verification method, so they are highly vulnerable to service denial attacks [11], witch attacks [12] and other attacks. In addition, it reduces the number of complete nodes, which will lead to a more centralized system.

3) Use SegWit technology. SegWit technology was officially activated on August 24th, 2017, with a block height of 481,824 [7]. SegWit thinks that the witness data (scriptSig) in the transaction input is the main contributor to the total transaction size. By migrating the witness data out of the transaction, the Bitcoin node can remove the witness data after verifying the signature, and the witness data does not need to be stored in the hard disk by all nodes, thus saving the storage space of a single node.

4) Lightning network and SegWit technology in research and development stage. Lightning Network [7] establishes a payment channel between the two parties in day trading, and pre-store part of the funds in the channel. The fund allocation plan after each transaction will be jointly confirmed while signing to declare invalid of the old version. When settlement is required, the final transaction result is written in the blockchain network for final confirmation. Since transactions only need to go through the blockchain when they are settled, the number of transactions submitted to the blockchain is greatly reduced.

The above solutions can optimize blockchain storage by being user-friendly and reducing the size and quantity of transactions submitted to the blockchain. However, they can neither clean out expired transaction nor address the root of the problem: endless addition of blockchain data. Therefore, they cannot solve the problem of increasing storage space essentially.

III. Solutions in Research Field

A great number of researches have so far been done on storage problems in the research field, and the solutions in the research can be divided into the following categories:

1) Sharding strategy [13-15]. In this strategy, a complete block is divided into K blocks, each block is stored in a certain proportion of nodes, and the storage conditions of nodes are recorded. When a block is needed, data is obtained from these nodes to restore the block. This strategy reduces the storage space required by a single node. But if several nodes stored in a certain block are inaccessible, the block cannot be accessed. This is a chain storage strategy, according to which, all data are stored on block chain.

2) Distributed storage strategy [16-17]. With the latest distributed storage technology, this kind of system distributes and stores blocks in RAID (Redundant Array of Independent Disks), Clouds, P2P [16] (Peer to Peer) or Elastic Chain [17], etc. Data can be accessed from these systems when needed. These systems can effectively guarantee the storage security and access efficiency of large-scale data. This is an off-chain storage method, which moves the data in the block to the off-chain, and only keeps the Hash of the data in the block for easy

searching and tamper-proofing. In the decentralized network, the problems of data centralization and the selection of the data remains storage.

3) Editing [18] or deleting strategy [19]. The editing strategy uses secret sharing strategy to manage trap door keys, which enables multiple verifiers cooperate to manage the modification rights of data on the chain, and avoid a single or a small number of malicious verifiers to tamper with the data at will after getting the trap door, thus improving the security and credibility of blockchain data [18]. The deleting strategy adopts some consensus mechanisms. For instance, PoSpace (proof of space), based on the consensus mechanism of space proof, deletes the expired data after being agreed and signed by most users, and the structure of the blockchain remain unchanged [19]. The above two strategies can only modify or delete at the block level, that is, the whole block must be completely replaced, so the operation granularity is too large [18].

4) Abandon strategy. *Bitcoin-Core* [20] proposes a method to save storage space by deleting all blocks after UTXO (Unspent Transaction Output) library is built. In fact, some full nodes use UTXO library to speed up data verification. The prune parameter mentioned above also adopts this storage scheme. However, in this way, the UTXO library will be at the risk of being tampered with, and the node cannot provide block sharing services, which will further aggravate the centralization.

5) Lightweight client strategy. In addition to full nodes, different blockchain systems also define their own lightweight clients. These lightweight clients including SPV node in Bitcoin, Jaxx, Status and Trust Wallet in Ethereum do not store all data but only recent blocks. So, data verification completely depends on full nodes. They are not only vulnerable to attacks but will also exacerbate centralization. Therefore, this strategy, which is merely a scheme to improve the user experience, is not enough to tackle the increasingly growing data volume.

IV. Solution

Whatever in practice research, traditional solutions to blockchain storage space can neither clean out previous transactions nor stop new transaction data from adding. That means these solutions fail to solve the problem of increasing blockchain data. Obviously, the addition of new transaction data is unavoidable, so it's necessary to clean out expired previous transaction. To be exaggerated, in order to ensure the traceability of data, should expired transactions 10,000 years ago continue to be kept in the blockchain? The answer is definitely "no". However, cleaning out expired transactions can affect the traceability of previous data. So, need to clean out expired transaction data and meanwhile ensure the correctness of the blockchain traceability, which is the root of the problem. Next, describe how to identify an expired transaction, rise a corresponding solution and then elaborate on the design of the substitute block and substitute chain of the scheme.

A. Expired Transactions

In Bitcoin system, transactions fall into two categories: Coinbase transaction and ordinary transaction (non-Coinbase). I will illustrate how they two can be deemed to be expired transactions according to different situations.

1) Coinbase transaction. Suppose the Coinbase transaction is transferred from A to B, and B transferred it to C. If the two transactions are included in blockchain and confirmed by a substantial number of subsequent blocks (more than 6), then the legality of the transactions is confirmed. When the transfer from B to C is confirmed, then that from A to B becomes an expired transaction since it no longer has the value of verifying a transaction but only had the value of tracing.

2) Ordinary transaction. The standard to judge whether an ordinary transaction is expired are as follows.

a) All the inputs of ordinary transactions are from expired transactions. As clean out expired transactions in advance, ensure all the ordinary transactions are from expired transactions by not finding such ordinary transactions in the unexpired transaction list.

b) All the outputs of ordinary transactions are spent before the expiration date.

What is expiration date? It should be noted that testing whether a transaction is expired or not at different time cut-off points (blocks) produces different results. For example, A transferred to B on February 5th, 2019, B transferred to C on March 5th, 2019, and C transferred to D on April 5th, 2019. If the deadline is March 8th, 2019, the transfer from A to B will be overdue. If the deadline is April 8th, 2019, both the transfer from A to B and the transfer from B to C are overdue.

What's worth mentioning is that to ensure the data are unalterable and traceable, many intermediate transactions cannot be defined as expired transactions. For example, a Coinbase transaction is transferred from A to B, and then B transfers to E, and E transfers to F. In order to make sure that the transfer from B to C is not miscalculated, the ordinary transaction from B to E (non-Coinbase transaction) is not defined as an expired transaction.

B. Solution

This paper proposes a solution to the problem of blockchain storage. That is re-blocking. In this solution, a certain block is taken as the cut-off block. After calculation, the unexpired transactions of several old blocks before the cut-off block (until the cut-off block) are extracted, repackaged into blocks, and these old blocks are replaced with new blocks. Thus, a substitute chain comes into being. In this way, not only can the expired transactions be cleared, but also the unexpired transactions can be kept. While replacing blocks, if the replaced block converges the cut-off block, new cut-off blocks can be re-selected to produce a new generation of substitute blockchain, as is shown in Fig. 2.

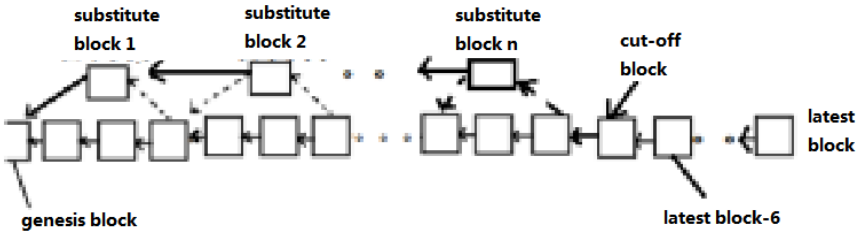

Fig. 2. Alternatives for Old Blocks.

To ensure that the data are correct, only if the latest block accepts more than six subsequent blocks can it serve as the cut-off block, which can be defined as required. A suggested figure is 3 months, because accounts in recent three months are usually included in the bank's billing system, while those three months ago are regarded as old accounts.

Bitcoin system makes miners willing to dig through reward mechanism. But substitute blocks will not generate new bitcoin rewards. If there is no reward then who will produce the substitute blocks? Rewards generate incentives, so do punishments. If a malicious node takes the opportunity to clear unexpired transactions, it will cause asset losses. If someone wants to protect its' own assets then participates in the generation of substitute blocks. The participation of plenty of nodes will generate computing power, thus safeguarding the correctness of data.

C. Design of Substitute Blocks

In Bitcoin system, one block is composed of a block header and a block body. Coinbase transaction must be the first transaction in the block body, followed by other ordinary transactions. Except for the structure of Coinbase transaction in the block head and the block body, the rest of the substitute block is consistent with the general blocks. Next, this paper will provide explanation for the block header and Coinbase transaction structure in substitute blocks.

1) Adjustment of the block header. In order to construct the block header, the mining node needs to fill in six fields: version number, timestamp, extra nonce, Hash value of the previous block, Hash of the Merkel root and difficulty coefficient. The first version number of the substitute blocks is set as to 0XFFFF, and the version number of each version upgrade will be reduced by 1 every time it upgrades. The Hash of the previous block is the Hash of the previous substitute block or the Hash of the genesis block. In order to stay consistent with the original block as much as possible, other parameters of block header are the same as those of ordinary blocks.

2) Structural adjustment of Coinbase transactions. In bitcoin system, the result of double SHA256 operation on the block header is the Hash of the block, which needs to meet the requirement of difficulty coefficient. The Hash of the block header adjusts as the variable nonce changes until the block Hash meets the requirement of the difficulty coefficient. That's when the nonce is obtained. However, the mining capacity continues to improve. When the computing power of mining nodes reaches 4GH/s, the variations of nonce in the block header will be exhausted within one second [7], and if the block Hash meeting the requirements is not found till now, the block cannot be generated. In order to solve the problem, introduce the Coinbase in the block body.

Like ordinary transactions, Coinbase transactions are divided into two parts: input and output. However, the input of the Coinbase is empty, and only the output is used to record the reward given by the mining node. Therefore, the empty input is used to fill in extra nonce. As the miners change the input of Coinbase, the Merkel root Hash in the block header changes accordingly so that it can meet the requirements of the difficulty coefficient.

Although part of the input of Coinbase is used to fill in extra nonce, the rest remains unused. Besides, the Coinbase transaction in substitute block does not produce an output, which means that the output is also idle. As a result, these idle spaces can be utilized to record all kinds of data that need to be recorded in the substitute block.

In Coinbase transaction, it's required to record the information of several merged blocks in block chain. To safeguard the correctness of data, add another two parameters intentionally: the Hash of the previous and next block of the original chain. These two parameters make it possible to have a two-way chain structure in the substitute chain, which helps the substitute chain to be attached to the original chain. This will be elaborated in the subsequent part about the design of substitute chain.

What needs to be recorded about Coinbase transaction in the substitute blocks are as follows:

1) Hash and time stamp of the cut-off block.

2) Block height of the final merged block in the original chain and the block height of the present block in the substitute chain.

3) The substitute block needs to connect blocks before and after it when replacing the original block, so need to record the total number of merged blocks in the original chain, the Hash of the first and the final merged block in the original chain as well as the Hash of the precious and the next block in the original chain.

4) Information about all the merged blocks, including the time stamp, version number, the number of unexpired transactions of the blocks on the original chain, details of which are in Table III.

5) Extra nonce is used to control the difficulty coefficient in Coinbase transaction.

Detailed information about the substitute blocks is shown in Fig. 3:

Next, solve the problem of how to record parameters mentioned above into the Coinbase transaction. The original input structure of Coinbase transaction is shown in Table I:

The input structure of Coinbase transaction of the substitute block in substitute chain is shown in Table II. Through Coinbase transaction, input an idle Hash and time stamp of the input transaction and stored the Hash and time stamp of the cut-off block. Store majority of information of the original chain and part of the information of the substitute chain into the addible transaction data.

Table III shows the original and substitute output structure of Coinbase transaction.

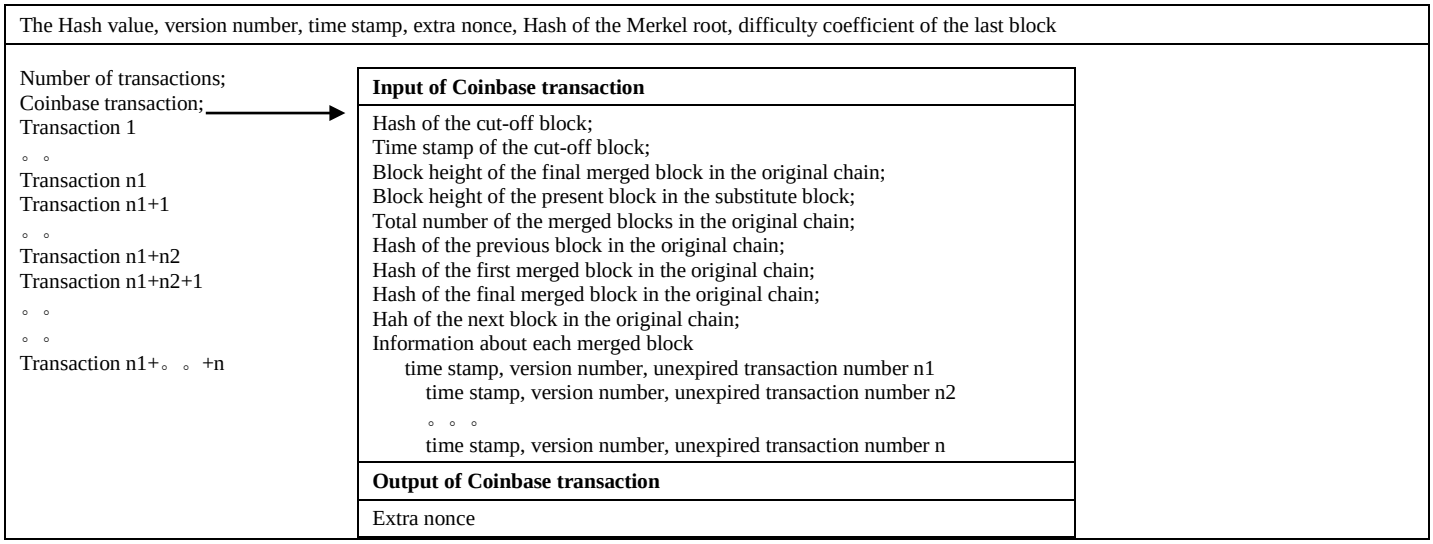

Fig. 3. Design of Substitute Block.

TABLE I. ORIGINAL COINBASE INPUT STRUCTURE

Size	Field	Description
32 bytes	Transaction Hash	Hash of transaction set to 0. No transaction is quoted.
4 bytes	Output Index	The index number of the UTXO to be spent, set to 0xFFFFFFFF
1-9 bytes	Coinbase Data Size	Coinbase Data length ranging from 2 to 100 bytes(VarInt)
Variable	Coinbase Data	Coinbase Data
4 bytes	Sequence Number	Sequence Number, currently-disabled Tx-replacement feature, set to 0XFFFFFFFF

TABLE II. SUBSTITUTE COINBASE INPUT STRUCTURE

<table>
<tr><th>Size</th><th>Field</th><th>Description</th></tr>
<tr><td>32 bytes</td><td>Cut-off Block Hash</td><td>Hash of the cut-off Block</td></tr>
<tr><td>4 bytes</td><td>Cut-off Block Time stamp</td><td>Time stamp of the cut-off Block</td></tr>
<tr><td>3-9bytes</td><td>Coinbase Data Size</td><td>Coinbase data length (VarInt)</td></tr>
<tr><td>Variable</td><td>Coinbase Data</td><td>Coinbase Data contains the size of original block height (1 byte),
original block height, the size of substitute block height (1 byte),
alternative block height, the total number of merged blocks (4 bytes),
the Hash of the previous block in the original chain (32 bytes),
the Hash of the first merged block in the original chain (32 bytes),
the Hash of the final merged block in the original chain (32 bytes),
the Hash of the next block in the original chain (32 bytes),
information of each merged block in the original chain as below:
<table>
<tr><th>Size</th><th>Field</th><th>Description</th></tr>
<tr><td>2bytes</td><td>Non-expire Transaction Number Size</td><td>The size of non-expire transaction number</td></tr>
<tr><td>Variable</td><td>Non-expire Transaction Number</td><td>Non-expire transaction number</td></tr>
<tr><td>4bytes</td><td>Version</td><td>Version</td></tr>
<tr><td>4bytes</td><td>Timestamp</td><td>Timestamp</td></tr>
</table></td></tr>
<tr><td>4 bytes</td><td>Sequence Number</td><td>Sequence Number, currently-disabled Tx-replacement feature set to 0XFFFFFFFF</td></tr>
</table>

TABLE III. COINBASE OUTPUT STRUCTURE

Size	Field	Description in Original Coinbase	Description in Substitute Coinbase
8 bytes	Amount	Bitcoin value in Satoshis	Bitcoin value in Satoshis, set to 0
1-9 bytes (VarInt)	Locking-script Size	Locking-Script length in bytes	Data length, between 2 and 100 (length of extra nonce)
variables	Locking-script	Locking-script	Extra nonce

D. Design of Substitute Chain

Connecting the above substitute blocks one by one gets a substitute chain. But there are two special blocks that cannot be merged into the substitute chain: the genesis block (stored by clients) and cut-off block (destination of the substitute chain).

Next, questions arise: how to deal with bifurcation, how to use consensus mechanism and what kind of driving mechanism to use, how to ensure data security, how to link to the original chain, how to reach consensus on the period duration, etc.

1) Bifurcation: If there is bifurcation in the original chain, choose the bifurcating block as the cut-off block. After substitution is finished, no new substitute chain produced until the problem of bifurcation is solved.

If there is bifurcation in the substitute chain, employ the principle of "short chain first". Then what if bifurcating chains are of equal length? The answer is, prevent such equally long bifurcating chains from existing. According to the design principle of POW (Proof of Work) consensus algorithm, the longer it last, the less likely that conflicts will arise; thus, prevent bifurcation from existing by extending the time of blocking of the substitute chain. It is unnecessary to produce substitute blocks at the same rate as new blocks; instead, produce substitute blocks twice as fast as the default speed. When a new generation of substitute chains is being produced and meeting bifurcation, such bifurcation can be ignored and thus wiped out.

2) Consensus mechanism: Although choosing POW consensus algorithm, yet, re-blocking bases on any safety consensus. POW algorithm can safeguard data security but that's through sacrificing the inefficiency of performance. If use other consensus algorithms, consider how to protect data security.

3) Driving mechanism: The driving mechanism of producing substitute chain is based on punishment rather than reward, as is mention above.

4) Data security: There are two kinds of nodes in the bitcoin system: lightweight nodes and full nodes. Main functions of full nodes include data storage, data verification and data sharing. Most full nodes are produced for the sake of mining.

Lightweight nodes are not plagued by the problem of space recycling. There are two kinds of full nodes: existing full nodes and newly added full nodes.

Existing full nodes in the bitcoin system have already been loaded with all the data, so when they receive substitute block, so long as the latter passes verification, the latter could be accepted; otherwise, the latter will be abandoned.

Talking about newly added full nodes, as expired transactions in old blocks have been cleaned out and the old blocks have been replaced with substitute blocks, some transactions cannot be confirmed. In this case, use the data confirmation strategy of lightweight nodes (based on POW consensus algorithm) to safeguard the correctness of data. If more than 6 subsequent substitute blocks are accepted, rest assured that the degree of data security of this substitute block is acceptable. But how about the ultimate 6 substitute blocks in the substitute chain? They do not have enough subsequent blocks (over six) to ensure the security of data; therefore, they cannot be linked to the original chain as substitute blocks. But set them aside for other nodes to use.

5) Linking mechanism: For the sake of data security, only blocks before the ultimate seven ones in the substitute chain can be linked to the original chain. The principles of linking are as follows.

According to the Hash of the next block of the original chain recorded in the last but seven blocks, corresponding block could be found in the original chain.

Make sure whether the Hash of the previous block (recorded in the original chain block) equals the Hash of the ultimate merged block listed in the last but seven substitute blocks.

Link substitute chain to the original chain after the blocks of both the chains has confirmed each other.

This means that any substitute block in the substitute chain can be linked to the original chain. So long as the substitute blocks at the linking area are confirmed by six subsequent blocks, such practice of linking is secure.

6) A new generation of substitute chain: A new generation of substitute chain can be reproduced in accordance with the predefined period rules only after the ultimate substitute block comes into being. Once one substitute chain fails to go through the whole substitute process, a new generation of substitute chain cannot be produced; otherwise, the substitute chain would be in disorder.

If there is bifurcation in the original chain which is not wiped out yet, the cut-off block can only converge the bifurcation area, failing to move further until the problem of bifurcation in the original chain is solved.

Besides, as is mentioned before, the substitute chain bifurcation can be ignored and wiped out when a new generation of substitute chain is being produced.

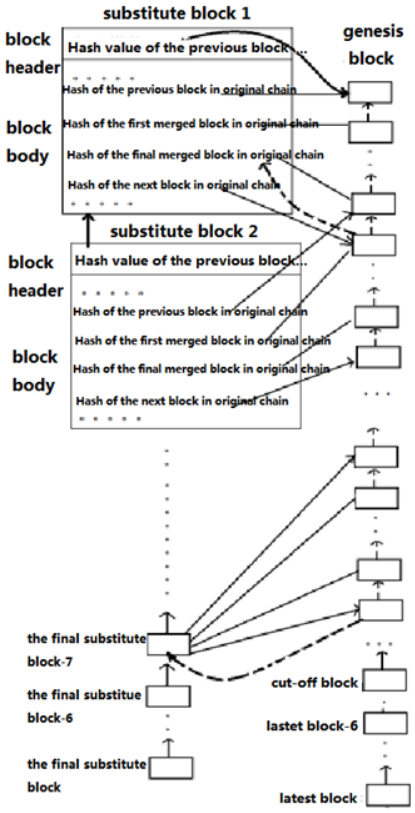

Fig. 4. Design of Substitute Blockchain.

Fig. 4 shows how a substitute chain can be linked to the original chain. The linking between the original chain and substitute chain is shown in dotted lines. This is because the two chains are linked together according to the association of block Hash, but they do not have real directivity association with each other.

V. Follow-up Solution

As is mentioned in the previous chapter, to make sure the temper-proof and traceability of data, some intermediate transactions cannot be cleaned out. For example, a Coinbase transaction is transferred from A to B and C, B transfers to E, and E transfers to F. If the transfer from B to E is cleaned out while that from A to B and C is retained, the bitcoin system would take it for granted that the bitcoins transferred to B by A are not spent. Because of this, ordinary (non-Coinbase) transactions from B to E cannot be cleaned out.

If record the fact that the bitcoins transferred from A to B have already been spent, B to E could be cleaned out. Such as retaining A to B and C transaction, and label the output of the transaction (for example, 1 means the output has been spent while 0 means the output hasn't been spent), as 10, which means the first output A to B has been spent but the second one A to C hasn't.

Since the information of merged blocks in the substitute blocks would be recorded, use the same way to retain the transactions were not all the outputs are spent and include the marks of these transactions' outputs. In this way, many intermediate transactions can be cleaned out while data security is still assured. This is in fact the way used in UTXO (unspent transaction outputs) library.

Based on the follow-up solution, the principles as to how to deal with transactions have changed as follows:

If all the transaction outputs are spent, clean out this transaction.

If part of the transaction outputs is spent, retain this transaction, and record the spending details of the transaction output.

To record how each transaction output is used, use 1 byte to mark the spending details (0 or 1) of 8 transaction outputs, then calculate how many bytes are there in total through calculation. Suppose use 1 bit to mark one output. And there are 36 outputs that need marking, since 1 byte equals 8 bits, use 5 bytes to mark 40 outputs. Then 2 bytes are needed to store that data 5, the follow-up 5 bytes to record spending details (0 or 1) of 36 transaction outputs, the surplus 4 bits being filled in with the Fig. 1.

As the solution changes, the information of each merged blocks recorded in the substitute block should also change, i.e., the information of each merged blocks in the original chain should change, including the time stamp, version number, the number of unexpired transactions (n), the size needed to store the transaction outputs (2n bytes), the spending details of each transaction output recorded by subsequent bytes. The design is shown in Table IV.

TABLE IV. Information of One Merged Block

Size	Field	Description
4 bytes	Version	Version number
4 bytes	Timestamp	Time stamp
2 bytes	Non-expire Transaction Number Size	Size of the number of unexpired transactions
Variable	Non-expire Transaction Number	The number of unexpired transactions n
Variable	Non-expire Transaction Output Flags Byte Number Per 2 Bytes	2*n bytes, the number stored by every two bytes refers to the number of bytes used to record the spending details of each transaction output
Variable	Non-expire Transaction Output Flags	Record the output spending details one by one according to the bytes needed to record a transaction as stipulated before. If it takes less than 1 bytes, it is still regard as 1.

Except the above changes, others are the same as before. But in this way, Coinbase transaction may become huge, which is a problem worth consideration.

VI. Experimental Analysis

Illustrate the correctness and effectiveness of the solutions mentioned above. The solutions are based on consensus mechanism. And since the correctness of POW consensus mechanism has been confirmed by bitcoin and many other decentralized systems, rest assured that the solutions are correct. Next, prove the effectiveness of the solution through an experiment.

How much space can the solutions in Sections III and IV save, respectively? According to the design of the algorithm, let's take the No.697546 block which get at the test time 2021-08-26 00:00 as the cut-off block and collect the data of each transaction from No.1 block to No.105,000 block along the block chain. In the solution mentioned in Chapter 3, if all the inputs of the transactions are from the outputs of expired transactions, and all the outputs have been used, then label such transactions as "expired transaction 1", otherwise as "unexpired transaction 1". In the design in Chapter 4, let's label unexpired transaction 1 whose transactions have been run out as "expired transaction 2", otherwise as "unexpired transaction 2".

Fig. 5 shows the data collected, including the number of expired transactions 1, the number of unexpired transactions 1 and additional expired transactions 2.

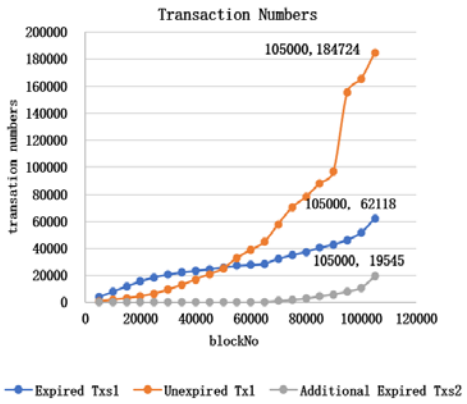

Fig. 5. Total Expired and Unexpired Transaction Number.

Applying the third solution, in bitcoin system, expired transactions account for 74.8% of the total transactions at the No. 105,000 block, and account for 77% (the peak) at the No. 95,000 block. In contrast, applying the fourth solution, more expired transactions would be cleaned out. Statistics show that cleaning out expired transactions can effectively reduce storage space occupation, which means this solution is effective. The ratio of expired transaction 1 as well as the ratio of expired transaction 2 is shown as follows in Fig. 6.

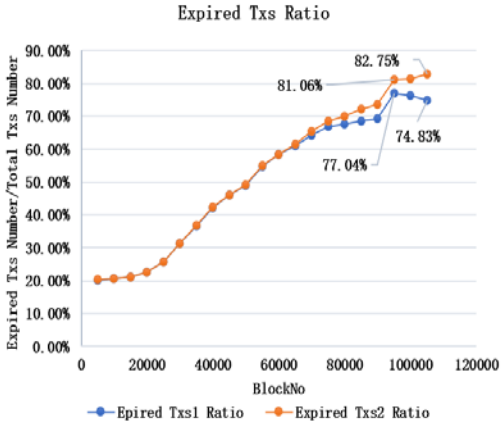

Fig. 6. Expired Transaction Ratio.

The above statistics only cover the data of the first 105,000 blocks. And how much space is unleashed? According to the solution mentioned in Section III, the expired transaction ratio will decrease after reaching the peak of 77%. And according to the solution mentioned in Chapter 4, the data of the change in size of the UTXO library would be gotten eventually. Due to the statistics of Bitaps [20], as of March, 21st, 2022 when this thesis is finished, there were 719,197,481 transactions, and 130,561,354 unspent transaction outputs in UTXO, and a total storage volume of 369.42 GB. Among them, the percentage of spent transactions is the same as that of the cleaned transactions. Roughly, the ratio of spent transactions to total transactions [(719 197 481-130 561 354)/719 197 481] reached 81.8%. Although the fourth solution does not make a big difference for the first 105,000 blocks, it will play an increasingly significant role in the recycling of subsequent blocks after the 105,000th block. So, the fourth solution is also an effective one.

VII. Conclusion

The increasing popularity of blockchains reveals all the possibilities a decentralized ledger can offer. Nonetheless, blockchains also have their limits: the fast-growing data cause single full nodes to occupy larger storage space, which will bring increasingly heavy burdens to online transmission, CPU computing power and storage space.

Base on analyzing transactions in blockchain, propose a new solution: re-blocking. In this solution, pack unexpired transactions in old blocks into a new alternative block which constantly replaces old blocks. In this way, a substitute chain comes into being, which helps to clean out expired transactions in the bitcoin system and thus solve the problem of increasing data. we also prove the effectiveness of the solution by conducting an experiment in bitcoin system. Believe that this system would highly beneficial to blockchain applications.

Further researches:

1) Realize this idea to verify whether it works well in reality.

2) Whether split one chain into two chains, for storing expired transactions and unexpired transactions to retain historical information, and merge them two to one when necessary.

3) We only give experiments for the bitcoin system. In other systems like electronic medical record system and produce tracing system, further analysis is needed to define and clean expired transactions.

4) In this system, POW consensus algorithm is used to protect date safety. Although highly probable, it is to be confirmed: algorithms can reach a consensus is ok.

5) Once long bifurcation chains arise like in Ethereum, how can the solution work well.

6) The second solution would change the traditional system mechanisms greatly, not only to realization, but also to usage. And the Coinbase transaction will become very large. It should be considered.

ACKNOWLEDGMENT

The study was supported by "Innovation Team Project of Humanities and Social Sciences in Colleges and Universities of Guangdong Province (No. 2020WCXTD008)".

DECLARATION

The authors declare that there are no conflicts of interest regarding the publication of this paper.

REFERENCES

[1] NAKAMOTO S. Bitcoin: a peer-to-peer electronic cash system[R]. Bitcoin, 2009.

[2] GUO Shang-tong, WANG Rui-jin, ZHANG Feng-li. Summary of Principle and Application of Blockchain[J]. Computer Science, 2021, 48(02):271-281.

[3] KORPELA K, HALLIKAS J, DAHLBERG T. Digital supply chain transformation toward blockchain integration[EB/OL]. https://scholarspace.manoa.hawaii.edu/bitstream/10125/41666/1/paper0517.pdf.

[4] TURKANOVIĆ M, HÖLBL M, KOŠIČ K, et al. EduCTX: a blockchain-based higher education credit platform[J]. IEEE Access, 2018, 6:5112-5127.

[5] CHOWDHURY M J M, COLMAN A, KABIR M A, et al. Blockchain as a notarization service for data sharing with personal data store[C]//Proceedings of the 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering. Piscataway: IEEE, 2018:1330-1335.

[6] World economic forum. World economic forum survey [EB/OL]. http://www.coinfox.info/news/3184-world-economic-forum-survey-10-of-global-gdp-maybe-stored-with-blockchain-technology-by-2027.

[7] Andreas M. Antonopoulos. Mastering Bitcoin: Programming the Open Blockchain[M]. O'Reilly Media, Inc, 2017:180.

[8] CoinMarketCap Website. CoinMarketCap's Statistics of Bitcoin [EB/OL]. https://coinmarketcap.com/zh/currencies/bitcoin/.

[9] Bitnodes Website. 730 days nodes statistics of Bitcoin[EB/OL]. https://bitnodes.earn.com/dashboard/?days=730(#nodes.).

[10] Blockchair Website. Blockchair' statistics of Bitcoin[EB/OL]. https://blockchair.com/bitcoin.

[11] LAU F, RUBIN S H, SMITH M H, et al. Distributed denial of service attacks[C]//Proceedings of the 2000 IEEE International Conference on Systems, Man and Cybernetics. Piscataway: IEEE, 2000:2275-2280.

[12] DING Qian. Research on Data Storage and Sharing Algorithm Based on Blockchain[D]. China University of Mining and Technology, 2020.

[13] Jia Dayu, XIN Junchang, WANG Zhiqiong, et al. Storage Capacity Scalable Model for Blockchain[J]. Journal of Frontiers of Computer Science and Technology, 2018(04):525-535.

[14] SUN Zhixin, ZHANG Xin, XIANG Feng, et al. Survey of Storage Scalability on Blockchain[J]. Journal of Software, 2021, 32(01):1-20.

[15] CAI Zhenhua, LIN Jiayun, Liu Fang. Blockchain storage: technologies and challenge[J]. Chinese Journal of Network and Information Security, 2020, 6(5):11-20.

[16] JIA Da-Yu1, XIN Jun-Chang, WANG Zhi-Qiong, et al. Efficient Query Model for Storage Capacity Scalable Blockchain System[J]. Journal of Software, 2019, 30(09):2655-2670.

[17] YUAN Yong, WANG Fei-Yue. Editable Blockchain: Models, Techniques and Methods[J]. ACTA AUTOMATICA SINICA, 2020, 46(05):831-846.

[18] REN Yanli, XU Danting, ZHANG Xinpeng, et al. Deletable blockchain based on threshold ring signature[J]. Journal on Communications, 2019, 40(04):71-82.

[19] Github website. Bitcoin source code[EB/OL]. https://github.com/bitcoin/bitcoin/blob/v0.21.0/doc/release-notes.md.

[20] Bitaps websit. Bitaps' statistic data of bitcoin[EB/OL]. https://btc.bitaps.com/.