Chinese Journal of Electronics
Vol.32, No.3, May 2023

RESS: A Reliable and Effcient Storage Scheme for Bitcoin Blockchain Based on Raptor Code

SHI Dongxian[1,2], WANG Xiaoqing[3], XU Ming[1], KOU Liang[1], and CHENG Hongbing[3]

(1. *School of Cyberspace, Hangzhou Dianzi University, Hangzhou 310018, China*)

(2. *College of Information Technology, Zhejiang Institute of Economics and Trade, Hangzhou 310018, China*)

(3. *College of Computer, Zhejiang University of Technology, Hangzhou 310000, China*)

Abstract — The Bitcoin system uses a fully replicated data storage mechanism in which each node keeps a full copy of the blockchain. As the number of nodes in the system increases and transactions get more complex, more and more storage space are needed to store block data. The scalability of storage has become a bottleneck, limiting the practical application of blockchain. This paper proposes a node storage scheme, called RESS, to integrate erasure coding technology into the blockchain to encode multiple blocks. Under the proposed block grouping method, nodes can reduce the times of coded block decoding. In addition, the coding scheme based on Raptor codes proposed in this paper has linear coding and decoding complexity. The rateless feature of Raptor code helps to achieve high decentralization and scalability of the Bitcoin network. RESS ensures data availability, efficiency and blockchain robustness based on achieving storage space scalability. Experimental results show that the proposed scheme reduces the storage requirements of nodes by nearly an order of magnitude.

Key words — Bitcoin blockchain, Storage scalability, Erasure coding, Raptor code.

I. Introduction

Blockchain is increasingly being utilized in the supply chain [1], Internet of things [2], [3], medical [4], audit [5], and other industries as a distributed, decentralized, and trusted database system. Each node in the blockchain network must follow cryptography-based rules, and each transaction must attain consensus with other nodes in the network. The node keeps all transaction information separately in a completely replicated manner, eliminating the need for endorsement by any third-party entity, and thus resulting in considerable cost savings. Table 1 shows the storage capacity of a full node as of August 2022, using Bitcoin and Ethereum as examples. The current Bitcoin blockchain size has exceeded 470 GB (gigabyte), and the Ethereum ledger has reached more than 350 GB [6]. Not every user has the capacity to dedicate such a large quantity of storage space to Bitcoin or Ethereum.

Table 1. Qualitative comparison of existing pruning approaches

Blockchain	Size of ledger	Number of blocks	Block generation time
Bitcoin	470.60 GB	736,550	10 min 15 s
Ethereum	350.36 GB	14,578,680	16.3 s

With the continuous expansion of the blockchain system, the full replication method has been unable to accommodate the growth of massive amounts of data. In the full replication storage mechanism, the amount of data grows exponentially with the increase of the number of blocks, and the entire system needs to invest huge storage space to save the block information. The gradual increase in transaction throughput requires huge storage space, which severely limits the application of blockchain and becomes a bottleneck for blockchain development [7]. Therefore, it is urgent to find an efficient scheme to solve the scalability problem of blockchain.

In order to overcome this problem and provide a suitable solution, Wang *et al.* [8] propose an efficient storage scheme (ESS) based on the distribution characteristics of the unspent transaction outputs (UTXO) in Bitcoin network. ESS sets a UTXO-weight for each block. According to UTXO-weight, it dynamically prunes the blocks which have lower query frequency,

Manuscript Received Oct. 7, 2022; Accepted Jan. 7, 2023. This work was supported by the National Natural Science Foundation of China (62072407) and in part by the “Leading Goose Project Plan” of Zhejiang Province (2022C01086, 2022C03139).

improving the scalability of the Bitcoin network. The Bitcoin Core team released the function of block pruning in version 0.11.0 [9]. Users firstly specify how much storage space to allocate for blockchain blocks. If the block data approaches the storage limit, the application will begin to erase the oldest block information while continuing to download new blocks at the same time, therefore conserving disk space. This consumes additional CPU resources and places a strain on the hard disk since it must read, write, and erase data at the same time. At this point, nodes only keep block header data, which is used to relay blocks to other nodes, handle reorgs, find old transactions, or rescan wallets. If pruning mode is enabled, clients must re-download the complete blockchain to return to the unpruned state. Anyway, this solution is a very basic and rudimentary pruning method that needs additional computational resources, and which make it a suboptimal option for Bitcoin storage.

Furthermore, several academic research advocate transferring the blockchain's original data to third-party devices for storage, such as the cloud or interplanetary file system (IPFS) [10]. Xu *et al.* [11] proposed a cloud-based optimization scheme. They chose to keep older blocks on the cloud, which were previously produced and were less likely to be queried, in order to relieve storage demand on blockchain peers. Zhou *et al.* [12] proposed a hybrid storage scheme to reduce the size of blocks by compressing some fixed-length fields in those transactions. And then, the newly generated block files were deposited to the IPFS private network to improve the scalability of Bitcoin network. Chou *et al.* [13] proposed a BC-Store framework. The framework deploys a data access model on the IPFS cluster system to classify hot and cold blockchain data. Hot data are stored in the local cache, while cold data are stored in the IPFS cluster, which greatly reduces the initial synchronization time of the blockchain and saves a lot of data storage space. However, storing blockchain data on a third-party device requires extra concerns for third-party device security. In addition, during the data update process, the data cannot be synchronized in time due to network delay, which can also lead to problems such as blockchain forks.

Some scholars have been inspired by distributed storage systems in recent years and proposed a strategy to adapt erasure coding technology to blockchain [14]. Erasure coding is a data protection method, which was originally used to solve the problem of packet loss in network transmission, and was later extended to the storage field to improve the reliability of storage. Qi *et al.* [15] proposed a novel storage engine for permissioned blockchain systems, called BFT-Store, to enhance storage scalability by integrating erasure codes with Byzantine fault tolerant (BFT) consensus protocol. However, BFT-Store encodes blocks through reliable sorage (RS) coding. According to the principle of RS erasure coding, excessive matrix inversion computing resources will be consumed during the decoding process. Wu *et al.* [16] proposed a new method that uses LDPC (low-density parity-check) code to reduce the memory requirements of blockchain nodes. They encode data across multiple blocks and use LDPC codes that provide simple decoding.

Currently, the most difficult challenge is to integrate the erasure coding technology in the blockchain and how to make the technology improve the data while minimizing the resource consumption of the access efficiency [17]. Erasure coding trades storage space optimization at the cost of high computing resources, so additional bandwidth and delay will be generated in the encoding and decoding process. Furthermore, compared with the full replication approach, the read performance of the blockchain is degraded by the decoding of data exchanges between nodes. To recover a block, a node needs to request a set of blocks from enough nodes to decode the block, which increases the blockchain network burden.

Motivation From the above analysis, we can find that there still exists some problems need to be solved, therefore, in this paper, we will study how to choose a suitable erasure code to combine with the blockchain to optimize the storage of Bitcoin network. After analyzing various erasure coding techniques, we finally choose Raptor code as the block coding technique for the proposed scheme. Raptor code is a kind of rateless erasure codes, that is, the encoded packets generated from the original data packet encoding are infinite. According to this feature, the Raptor code can combine k original data blocks into any number of coding blocks, while the receiver only needs to receive k' coding blocks (k' is slightly larger than k), and the original data can be recovered with high probability. This feature of the Raptor code enables the encoded node to restore the original block after receiving a fixed number of encoded blocks. When a few nodes break down, it is not necessary to decode all blocks of a node in order to restore the full blockchain, then code again as the approach proposed in [15]. In addition, the Raptor code can complete the reliable recovery of the original data with the current coding and decoding complexity, which greatly reduces the coding and decoding complexity.

Considering the extra bandwidth and overhead involved in the encoding and decoding process, if the entire block is encoded, the encoded block will require frequent encoding and decoding operations when being accessed. To improve the throughput of the blockchain node, we only encode and store the old data in the block, and keep the complete block information for the frequently accessed data. In this way, storage space can be saved without putting too much burden for network load, which is very different with and more efficient than current existing works.

Contribution In summary, the main contributions of this paper can be listed as:

1) This paper examines the characteristics of existing corrective coding techniques and blockchain networks. A block coding scheme based on Raptor codes is proposed for blockchain networks, which reduces the storage burden of full nodes in the Bitcoin network.

2) The proposed block coding scheme in this paper has the complexity of linear coding and decoding, which improves the reading performance of coded blocks. Furthermore, we refer to the encoded nodes as RESS nodes for short. RESS nodes support parallel access to a few nodes to recover specific block data without causing excessive network load.

3) This paper proposes a block grouping strategy, which regularly encodes and saves a group of blocks. Keep the newly generated set of blocks and group-encode the old blocks. When a new block is created, all transactions can be verified by decoding them only once or twice during the verification process, and nodes do not need to read data frequently for decoding operations.

4) Comprehensive simulation, analysis and comparison are performed among the proposed scheme and other related schemes, the results verified that the proposed scheme can be reliable and efficient applied in Bitcoin network.

Organization The rest of this paper is organized as follows: Section II introduces some knowledge related to our research. Section III provides a detailed design scheme of the block encoded method. Section IV illustrates the performance analysis of RESS. Section V performs a comprehensive experiment and comparison for the proposed scheme, and Section VI describes the conclusion and future work.

II. Related Work

In this section, we will introduce two types of node storage technology and related works about erasure encoding.

1. Bitcoin blockchain storage

This section takes the Bitcoin network as an example to introduce the general storage mode of the blockchain. In the blockchain, there are two approarches for nodes to store blocks: full nodes and light nodes. The full node stores all transaction history data from the day Bitcoin was created to the present. The light node only keep block header information for each block and do not need to keep information about the entire blockchain transactions.

Full nodes are responsible for monitoring transaction information on the Bitcoin network and then verifying the legitimacy of each transaction. Any transactions and blocks that do not conform to the rules will be discarded by full nodes. Once a transaction or a block is verified by full nodes, it will forward the data to other full nodes to reach consensus.

Unlike full nodes, light nodes do not validate blocks or store a copy of the blockchain, they collect information from full nodes. When a light node needs to verify a transaction, firstly, the light client needs to initiate a special confirmation request to broadcast the transaction to all neighbour full nodes on the network. After the full node receives the transaction information, it will search which block the transaction belongs to, and then calculate the Merkle tree of this block. Send the partial Merkle tree related to the current transaction to the light node. In this way, after the light node receives this partial Merkle tree, it performs a series of operations locally. Firstly calculate the hash of this transaction, and then obtain the Merkle root according to each hash value on the partial Merkle tree. If this value is equal to the Merkle root value in its own block header, it means that the transaction verification is successful.

According to the above principles, light nodes can only verify whether the transaction already exists. That is, to determine whether the transaction used for "payment" has been verified by the full node, and to be protected by how much computing power (how many confirmations). Due to the lack of complete transaction records, light nodes cannot verify that a transaction does not exist, and this vulnerability can easily lead to denial of service attacks and double-spending attacks [18]. In addition, light nodes need to randomly link multiple nodes, increasing the probability of connecting with at least one reliable node, but this random link requirement is also vulnerable to network partitioning and sybil attacks.

Since each full node maintains the entire network data, even if some of the nodes have problems, such as disconnection or hacker attack, it will not affect the security of the entire blockchain network. This is also the advantage of a decentralized accounting system. At the same time, the more the number of full nodes, the more copies of the complete blockchain ledger will be saved, the stronger the immutability of block data, and the higher the security of the entire blockchain network. Once someone attempts to organize a mutiny of computing power, he/she tries to change the block consensus or launches a double-spending attack, other normal full nodes can immediately verify and reject these transactions.

The existence of a full node is crucial to maintaining the stability and security of the blockchain. The more nodes there are, the more resilient the network is, providing better overall protection against malicious parties and attackers. When the number of full nodes is large, malicious behaviors in the network can be detected early. Assuming such an extreme scenario, evil miners gain control of the Bitcoin network. They adjust network rules to their will, such as increasing block rewards. In this case, if the number of full nodes in the network is high, light clients will quickly realize that they cannot send transactions to full nodes and avoid using the Bitcoin blockchain until the attacker is eliminated. Conversely, when there are only a few full nodes in the blockchain network, the lightweight client will actively utilize the BTC scheme, resulting in the successful hijacking of the Bitcoin blockchain by potential malicious miners.

Furthermore, by running a full node, users have a complete copy of the blockchain on the device, because of this reason, they can get faster query speed. When a light node requests information from a full node, the full node must request all transactions sent and received by the light client to confirm them, and this payment verification obviously violates users' privacy. Therefore, compared with lightweight nodes, full nodes have higher privacy guarantee for users.

Based on the aforementioned issues, as well as the massive storage issues confronting the present blockchain, it is critical to investigate and improve the blockchain's storage structure.

2. Erasure code

Erasure code is a forward error correction technology that can achieve high availability and high reliability in storage systems and communication systems [19]. In communication network transmission, it is mainly used for data packet loss recovery. In the network storage system, it is mainly used to improve the reliability of data storage. Compared with the redundant replication method of multiple copies, erasure coding technology can reduce the redundancy of data storage on the basis of ensuring data reliability. Erasure coding can reduce storage overhead as much as possible while guaranteeing the same fault tolerance performance.

Table 2 shows the performance evaluation comparison of RS codes, LDPC codes, luby transform (LT) codes, and Raptor codes. Traditional RS codes and LDPC codes are typical block codes with a fixed code rate. That is, for the whole of n coded symbols, k input symbols are used to generate $(n-k)$ fixed number of redundant codes, and the code rate $r = k/n$.

Table 2. Qualitative comparison of existing coding approaches

Erasure coding	Encoding complexity	Decoding complexity	Rateless
Cauchy-based RS code	$\mathcal{O}(n^2)$	$\mathcal{O}(n^2)$	No
LDPC code	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$	No
LT code	$\mathcal{O}(\ln k)$	$\mathcal{O}(k\ln(k/\delta))$	Yes
Raptor code	$\mathcal{O}(k\ln(1-\varepsilon))$	$\mathcal{O}(k\log(1/\varepsilon))$	Yes

Raptor code is a digital fountain code. Compared with traditional erase codes, there is no need to determine the coding rate in advance. The fountain code divides the original file into a certain number of data packets, and then encodes any number of valid encoded packets through the data packets. When the sender transmits the data, as long as the receiver receives a subset of encoded packets slightly larger than the number of original packets, the original data can be restored. This is a very efficient way of transferring data in lossy connection scenarios. Because during transmission, the receiving terminal neither considers the packet loss rate nor requires feedback on the received packets. After the concept of “digital fountain” was proposed, Luby realized the first practical digital fountain code LT code [20]. In view of the shortcomings of LT code, Shokrollahi proposed Raptor code [21] based on LT code, which has greatly improved the coding and decoding efficiency as well as decoding success rate, the scheme almost achieves ideal coding and decoding performance.

In view of the advantages of the special Raptor code, such as fountain code 4, and at the same time, it is a kind of technology without code rate and linear encoding and decoding. Combined with the characteristics of blockchain, our proposed scheme will use Raptor code to encode blocks for Bitcoin block. The proposed scheme will be introduced and designed in Section III.

III. System Model and Scheme Design

In this section, we propose an efficient storage model to reduce the local storage cost for Bitcoin blockchain by coding blocks. We refer to the node using this method as an RESS node, which no longer saves the original block transaction Bitcoin block data. Instead, the Bitcoin nodes will store encoded block information after encoding a set of blocks in the proposed scheme.

1. System model

In order to describe the proposed scheme in detail, we construct a system model based on Bitcoin blockchain networks, which is shown in Fig.1 . The nodes firstly divide the input Bitcoin blocks into different groups, and then encode each grouped blocks to obtain the encoded blocks. After storing the corresponding coded blocks according to the specified sequence, the node deletes the original transaction information of the coded blocks. After all packets having been encoded, the final node state will be obtained. When the block information needs to be recovered, the node only needs to download necessary enough encoded blocks from the peer node to perform the decoding operation. After these operations, the original Bitcoin block will be recovered successfully.

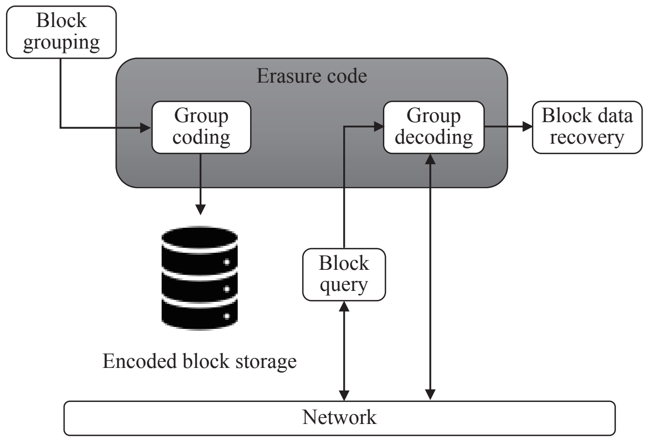


Fig. 1. System model.

2. Code block selection

Generally, in the Bitcoin network, a block can contain more than 2,000 transactions. Under such situation, nodes will visit multiple blocks at the same time when validating transactions. If each block is encoded separately, the decoding operation of multiple blocks will be involved when verifying all transactions in a bitcoin block, which will be in low efficiency because it will consume a lot of computing resources.

In this scheme, we encode consecutive t blocks. When a node conducts block transaction verification, it only needs to decode the latest one or two sets of blocks to verify all transactions, which saves computational resources to a great extent. In addition, in order to allow nodes to quickly perform block verification, we do not perform encoding operations on the most recently generated t blocks. Therefore, keeping the latest block allow nodes to verify most of the transactions in the block without performing a decoding operation. For a small number of transactions from older blocks, a node may only need to perform only one decoding operation.

3. Block grouping

Assuming that the number of blocks in a node is n, the node starts from the genesis block with t as the unit, and divides the consecutive t blocks into a group with a total of m groups. Let $G_m = [B_{m1}, \ldots, B_{mt}]$, where G_m represents the m-th group, and B_{mt} represents the t-th block in the m-th group, as shown in Fig.2.

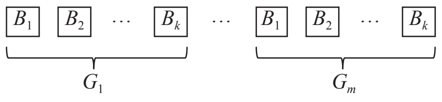


Fig. 2. Block grouping.

The blocks are group-coded according to G_m. N is an certain integer which it is not necessarily a kind of multiple integer t. At this time, we consider two cases. Let $c = N - \lfloor N/t \rfloor t$, if $c < t/2$, then keep the complete information of $t + c$ new blocks. If $c \geq t/2$, then only c new blocks are unencoded.

Divide G_m groups of blocks into k original blocks on average. If a block only needs $k' < k$ symbols to represent, then the remaining $k - k'$ symbols are filled with zero. Therefore, each group of blocks G_m includes kt symbols, and each symbol is represented by a_{ij} $(1 \leq i \leq k, 1 \leq j \leq t)$. Arrange the kt symbols of the G_m group into a $k \times t$ matrix such that the j-th column contains the symbols of the j-th block constituting G_m, as shown in matrix (1).

$$\begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1t} \\ b_{21} & b_{22} & \cdots & b_{2t} \\ \vdots & \vdots & \ddots & \vdots \\ b_{k1} & b_{k2} & \cdots & b_{kt} \end{bmatrix} \tag{1}$$

After encoding each group of blocks, the node saves a certain number of coded blocks and deletes the transaction information of the original block. Each full node executes the encoding process independently and keeps the encoded blocks.

4. Encoder of Raptor code

In this paper, we will take a set of chunks as an example to illustrate the encoding process, Fig.3 shows the two stages of encoding for each chunk in the proposed scheme. Firstly, the block data is pre-coded to obtain the intermediate coding block, and then the intermediate coding block is LT-coded to obtain the final coding block. Next, we will introduce the two stages in detail.

1) Pre-coding. The pre-coding stage uses regular LDPC codes. Using Gaussian elimination and column transformation, the check matrix H is transformed into a standard matrix $H_0 = \begin{bmatrix} P^{\mathrm{T}} & I_{N-t} \end{bmatrix}$, where I represents a unit matrix of order t, and P represents a $t \times (n-t)$ matrix containing parity bits. Next, calculate the generator matrix of the LDPC code $G = \begin{bmatrix} I_t & P \end{bmatrix}$. Finally, use the generator matrix G to encode the vector $b_i = \begin{bmatrix} b_{i1} & b_{i2} & \cdots & b_{it} \end{bmatrix}$, $i = 1, \ldots, t$.

$$(v_{i1}, \ldots, v_{in'}) = a_i G \tag{2}$$

The equation (2) obtains the precoding block BP

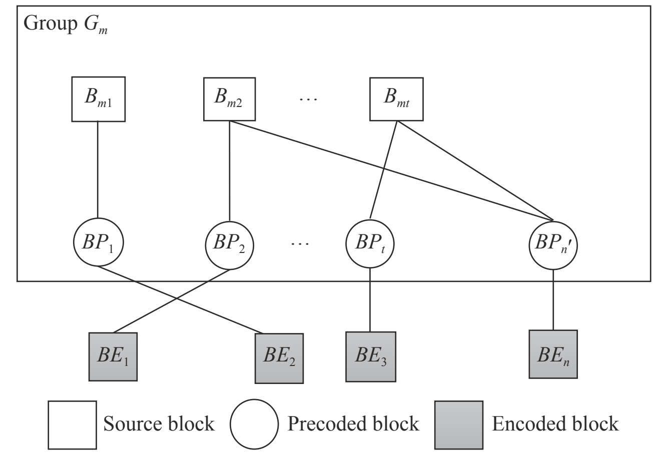


Fig. 3. Encoding process.

and because BP is systematic, $\{BP_1, BP_2, \dots, BP_{n'}\} = \{[B_{m1}, B_{m2}, \dots, B_{mt}], [P_1, P_2, \dots, P_{n'-t}]\}$, the first t outputs correspond to the original input block, followed by the parity block P.

2) LT-coding. Secondary encoding is performed on the pre-coding block $\{BP_1, BP_1, \dots, BP_{n'}\}$ using the LT code. The degree distribution is shown in equation (3), set ε is a positive number, $D = [4(1+\varepsilon)/\varepsilon]$, and $\mu = (\varepsilon/2) + (\varepsilon/2)^2$ [21].

$$\Omega_D(x) = \frac{1}{\mu+1}\left(\mu x + \sum_{i=2}^{D} \frac{x^i}{i(i-1)} + \frac{x^{D+1}}{D}\right) \tag{3}$$

The specific process is mainly divided into several steps:

a) Generate a random number d according to the degree distribution function $\Omega_D(x)$.

b) Randomly select d intermediate blocks from $\{BP_1, BP_2, \dots, BP_{n'}\}$, and obtain a coding block BE by performing XOR operation on these d intermediate blocks.

c) Repeat the above process until n coding blocks $\{BE_1, BE_2, \dots, BE_n\}$ are generated and stop coding.

In order to ensure that all encoding blocks are allocated to node storage, we make the following settings. Assuming that n is the number of coding blocks and h is the block height, let $i = h - \lfloor h/n \rfloor n$, then the block height is h and the block saves the coding block BE_i. In this way, the coded blocks are guaranteed to be evenly distributed across other nodes in the network.

The encoded node block structure is shown in Fig.4. for the first m block groups, the node stores one or more coded block obtained by encoding each block group. The nodes do not encode the remaining new blocks and instead store the entire block information.

5. Decoder of Raptor code

When a node validates a transaction in a block, it may need to access the encoded block. Nodes need to collect k encoded blocks for decoding to get the original block. The decoding step is mainly divided into two stages. The first stage is decoding and restoring the precoded block $\{BP_1, BP_1, \dots, BP_{n'}\}$. And the second stage is to decode and restore the original block $\{B_{m1}, B_{m2}, \dots, B_{mt}\}$.

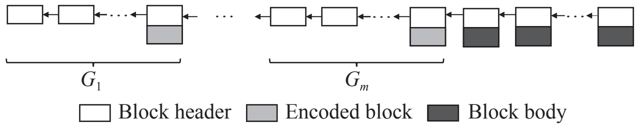


Fig. 4. Node state after encoding.

1) LT-decoding. The goal of the first stage is to obtain precoded blocks. The decoding process mainly includes the following steps:

a) First, the adjacent edges and degree values of the coding blocks are obtained through the control information, after then a bipartite graph g of n intermediate blocks and $k(1+\varepsilon)$ coding blocks will be formed.

b) Set the precoding block BP to the empty start algorithm and find a coding block BE, which is only connected to one precoding block BP_j in g. Copy the value of BE to the precoding block BP_j connected to the BE.

c) The XOR operation is performed on the precoding block BPj and the coding block connected to it, then the value of BP_j is updated. In the bidirectional graph, delete all edges of the precoding block BP_j, and reduce the degree of the corresponding coding block by 1.

d) If there is a new BE degree of 1 in the new bipartite graph, repeat the above process until it does not meet the condition, and finally decode precoding blocks BP.

In the above process, if there is no coding block with degree 1, decoding fails. The node needs to re-request new encoded blocks for decoding. According to the characteristics of the intermediate block BP_j, in the first stage of decoding, the original block vector v_j $(1 \le j \le t)$ can be decoded. Hash the decoded block to get the Merkle root and compare it with the Merkle root parameter in the block header. If they are the same, it means the block decoding was successful. If not, the second stage LDPC decoding operation will be performed.

2) LDPC-decoding. The parity check matrix is used in the second step to decode the original block from the intermediate block. Firstly, the generator matrix G is used to generate the standard check matrix H_0, and then H_0 is used to restore the original block. Similarly, the Merkel root in the block header is used to determine if the block is correctly decoded. If decoding failed, keep requesting the encoded block from the network and decoding based on the recovered block until

users have the proper block transaction information.

IV. Analysis of RESS

In this section, we will discuss the proposed scheme-RESS in terms of data availability, storage scalability, robustness and efficiency of coding complexity in detail.

1. Data availability

The coding rate of the standard RS erasure code is k/n, which means that k input symbols obtain $n-k$ redundant symbols for n coded symbols. Furthermore, in the RS coding scheme, it is often important to ensure that the coded blocks be distributed and stored in a certain number of nodes. Historical blocks must be decoded and re-encoded when the number of nodes changes, which consumes additional network traffic.

The Raptor code employed in this scheme is a rateless erasure code that dynamically adjusts the coding rate to adapt to diverse conditions. The encoder and decoder of Raptor codes use proportionate amounts of memory, and decoding time is only determined by the data length, regardless of how many coded symbols are created and conveyed by the encoder. Raptor codes have linear coding and decoding complexity, allowing nodes to recover blocks with low-complexity block recovery methods. When a few nodes do down, there is no need to re-encode the entire network. Sometimes nodes go down or lose data on blockchain network. There is no need to re-encode the entire network in this situation, and as long as a specific number of encoded blocks are gathered, the Raptor code can retrieve the original block with a high probability.

2. Storage scalability

Assume that at a certain moment in the transaction for Bitcoin network, there are h blocks in a node divided into m groups with t blocks in each group, and each block body consists of k symbols. The block header is a fixed 80 bytes occupying 640 symbols, and there are n nodes in the Bitcoin network. In a standard bitcoin blockchain, the storage required by these nodes is $s' = ht(640+k)$ symbols. In this proposed scheme, assuming that each group consists of t blocks, the storage resources required by these nodes are at least $nm(640+k)$, and compared with uncoded blocks, the storage resource which have been saved can reach t times at most.

The major storage overhead of the block in a full node comes from the transaction data in the block body. Disregarding the storage space occupied by block headers, the average block body storage consumption $\bar{s_B}$ of RESS nodes is

$$\bar{s_B} = \frac{(m+c)k}{h} \tag{4}$$

In equation (4), c denotes the number of uncoded blocks. When a new block is added in Bitcoin network, the storage growth rate of the node is $s' = 1/thk$, decreases continuously with the increase of t. Therefore, more small capacity devices can be supported to join the blockchain network and improve the scalability of the whole Bitcoin blockchain network.

3. Robustness

In order to get the encoding block, the Raptor encoding method randomly selects the original data block to perform XOR operation according to the degree distribution, which will change the content of the original data. Even if the attacker receives the stored coding block, he/she cannot know the precise content, since it is not a collection of input blocks and check blocks. This improves the blockchain's security obviously. When a node wishes to tamper with the content of a block, it must collect the minimum necessary coded blocks. However, according to the security mechanism of our proposed scheme, the probability of an attacker obtaining the minimum necessary encoded block is extremely low.

In our proposed scheme, If a malicious node tries to change the content of the encoded block, the node will be unable to decode it because it is not possible for the malicious node learn the decrypted key. However, this has no impact on the network's stability since if a node fails to decode, it can request data from other secure nodes to re-decode. Suppose there is a node A in the network that is requesting an encoded block from node B. When node A receives $\alpha\ (\alpha \geq 1)$ incorrect encode blocks from node B, then node A no longer sends block requests to node B and broadcasts the message that node B has a malicious block to the blockchain network. When $\beta\ (\beta \geq 1)$ nodes in the network send the message that node B has malicious coding blocks, it can be judged that node B is a malicious node and the neighboring nodes of node B can choose to disconnect from it. α and β can be adjusted flexibly according to the current state of the Bitcoin network.

Furthermore, in our proposed scheme, each node independently encodes blocks and stores the coding blocks. Using Raptor codes, nodes can choose to generate any number of coding blocks, and the success rate of decoding depends only on the number of coding blocks k and overload ε. A failure of a single RESS node has very little impact on the stability of the network, and nodes that need to decode can do so by requesting coding blocks from other nodes. There is an extreme case where the number of honest RESS nodes in the network is less than $k(1+\varepsilon)$, in which case. Nodes can choose to use the currently existing coding blocks to recover a part of the original blocks firstly, and then recover the rest of the blocks based on this part of the original blocks.This extreme case is in a very low possibility and will not happen in practical applications.

4. Coding complexity and efficiency

In this section we will analyze the efficiency of our proposed scheme based on its complexity. Raptor codes' encoding overhead is determined by the encoding algorithm which has been used twice. In this scheme, LDPC code is utilized for precoding, and LT is for secondary coding. According to the Raptor coding principle, it exists a positive real number c (depending on ε) such that any set of $(1+\varepsilon/2)n+1$ output symbols of an LT code with parameters $(n, \Omega_D(x))$ has the largest error probability is e^{-cn}, which suffices to recover at least $(1-\delta)/n$ input symbols by BP decoding, where $\delta = (\varepsilon/4)/(1+\varepsilon)$. Assuming that the probability of malicious nodes in the network is σ, and the precoding stage generates $n' = k/R$ intermediate blocks $\{BP_1, BP_2, \ldots, BP_{n'}\}$, then $k(1+\varepsilon)$ coding blocks are necessary to reliably restore original block data. Assuming that ε is a fixed positive real number, for each n there is a linear code C_n of length n, let $R = (1+\varepsilon/2)/(1+\varepsilon)$, then the belief propagation (BP) decoder can use arithmetic operation $\mathcal{O}(n\log(1/\varepsilon))$ decodes C_n with erasure probability $\varepsilon = (\varepsilon/4)/(1+\varepsilon) = (1-R)/2$ on the deletion channel [21]. As a result, our proposed scheme's coding and decoding overhead is $\mathcal{O}(\frac{\log(1/\varepsilon)}{t})$. In comparison to [15], the proposed technique achieves linear encoding and decoding complexity, and it does not require decoding and re-encoding all blocks when the number of nodes changes, so our proposed scheme is efficient.

V. Experimental Results

We extracted data from the first 600,000 blocks of the Bitcoin network and used these data to conduct simulation experiments to evaluate our proposed scheme. The experiment run on a computer that equipped with a CPU with 3.1 GHz, an i5 version Intel Core, a memory with 16 GB.

1. Storage consumption

As mentioned in the analysis step of Section IV, our proposed scheme will decrease storage space for Bitcoin node efficiently, compared with other related schemes which almost keep full nodes as copy, our proposed scheme can save a lot of storage space. Fig.5 depicts the storage occupancy of the original block file and the encoded block file with 1000 blocks as a group, with each block divided into $k=10$, 100, and 1000 segments. The storage requirement of the node decreases as k grows.

Table 3 illustrates the storage requirements of a single node for $t=1000$, 2000, 4000, and 10000 blocks in each group. This experiment uses the control variable $k=10$ to analyze the influence of changing the number t of each group of blocks on storage space. It can be concluded that as t rises, the node storage requirement does not vary considerably, and the storage decrease increases gradually. However, because there are a high number of nodes in the actual blockchain, the storage space saved by the blockchain is enormous.

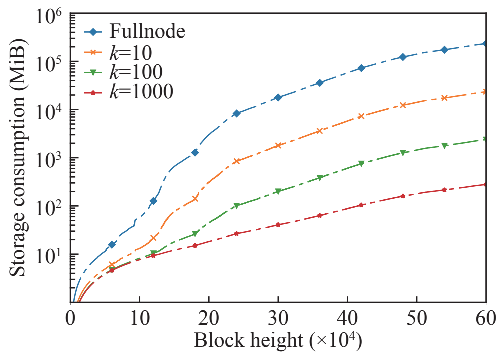


Fig. 5. Node storage requirements.

Table 3. The effect of the number of coding blocks on storage space

t	Storage consumption (MB)	Storage reduction
1,000	23,420.96318	89.9806%
2,000	23,421.19205	89.9805%
4,000	23,420.96317	89.9806%
10,000	23,416.84329	89.9823%

When a block is retrieved, the more encoded blocks a node maintains, the fewer encoded blocks are requested from the other nodes. While increasing the coding time, it also raises the storage requirements. Table 4 evaluates the percentage of node storage space savings when $k=10$ and a single node stores $n=1$, 4, 6, and 8 encoded blocks. Saving a certain number of encoded blocks not only reduces the block decoding time, but also relieves the storage requirements of nodes to a certain extent.

Table 4. Storage space saved when nodes store different numbers of encoded blocks

n	Storage consumption (MB)	Storage reduction
1	23,420.96319	89.9806%
4	93,500.74697	60.0007%
6	140,251.1205	40.0011%
8	187,001.4939	20.0015%

In addition, in order to show the storage saving of the proposed scheme, we compare our proposed scheme with [14]. Fig.6 shows that when the number of blocks per group is considerable, our proposed scheme saves a lot of storage space. In the case of a blockchain network with a high number of blocks, adopting this approach to combine more blocks can greatly reduce storage space which will be great advantage for the development of Bitcoin blockchain network.

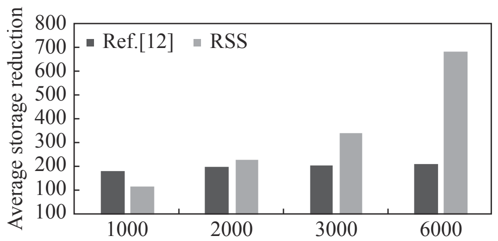


Fig. 6. Average storage reduction.

2. Qualitative comparison

For our best knowledge, there is currently just a small amount of literature that uses erasure coding technology as a blockchain storage scalability solution, and the entire field is still in its early stages. This paper collects the existing solutions and makes a qualitative analysis for them and our proposed scheme, the results are as shown in Table 5.

Table 5. Qualitative comparison of existing coding approaches

Existing solutions	Coding scheme	Linear complexity	Compatibility of existing blockchain
LS node [22]	Random linear codes	Yes	Yes
BFT-Store [15]	RS code	No	Yes
[14]	LDPC code	No	No
SeF [23]	LT code	No	Yes
RESS	Raptor code	Yes	Yes

Loose synchronized (LS) node [22] is an earlier work that proposed the application of erasure codes for blockchains. This scheme encodes and allocates each block, reducing the required memory. As the number of blocks increases, encoding a single block will take a lot of computational resources. BFT-Store [15] proposed a block coding scheme based on RS code for permission blockchain, but using this fixed-rate coding scheme, when the nodes in the network change, all nodes need to re-encode the block, which will consume huge computing resources. The above previous schemes all based on the research of erasure codes with fixed code rates. Distributed error correction coding [14] used LDPC codes to encode multiple blocks, but LDPC codes have high encoding and decoding complexity, which is not an excellent choice for public blockchains with massive data.

SeF [23] first proposed a block coding scheme based on fountain codes, using LT codes to encode a group of blocks. RESS also uses fountain codes to encode blocks. The difference is that our proposed scheme chooses Raptor codes with linear coding and decoding complexity, which can reduce the storage of the blockchain network at a lower cost and does not change the underlying mechanism of the blockchain network, at the same time, it has better compatibility for different coding technology.

VI. Conclusions and Future Work

Considering the serious storage burden of Bitcoin network, in this paper, we proposed a coding scheme to solve the storage explosion problem in the Bitcoin blockchain system. The proposed coding approach encoded a group of blocks using Raptor codes firstly, and then only part of the coded blocks in Bitcoin network was kept at each node in the blockchain. Even if blocks were no longer stored in their original form because it will cost too much storage space, they can still be recovered by downloading fragments from other nodes and performing inverse linear operations. The analysis and experiment results show that our proposed scheme can improve the availability, robustness and efficiency of Bitcoin block chains without affecting data integrity. Accordingly, by reducing the burden on existing nodes and the proposed scheme will also encourage new nodes to participate in the Bitcoin network efficiently.

In the future work, we will consider how to integrate the erasure code technology into existing research, such as block sharding technology, side-chain technology, etc., that will achieve high-performance storage operation of Bitcoin nodes in a more efficient way.

References

[1] S. Saberi, M. Kouhizadeh, J. Sarkis, *et al.*, "Blockchain technology and its relationships to sustainable supply chain management," *International Journal of Production Research*, vol.57, no.7, pp.2117–2135, 2019.

[2] X. L. Xu, X. Y. Zhang, H. H. Gao, *et al.*, "BeCome: Blockchain-enabled computation offloading for IoT in mobile edge computing," *IEEE Transactions on Industrial Informatics*, vol.16, no.6, pp.4187–4195, 2020.

[3] M. L. Dai, S. Y. Xu, S. J. Shao, *et al.*, "Blockchain-based reliable fog-cloud service solution for IIoT," *Chinese Journal of Electronics*, vol.30, no.2, pp.359–366, 2021.

[4] Y. Chen, S. Ding, Z. Xu, *et al.*, "Blockchain-based medical records secure storage and medical service framework," *Journal of Medical Systems*, vol.43, no.1, article no.5, 2019.

[5] K. F. Fan, F. Li, H. Y. Yu, *et al.*, "A blockchain-based flexible data auditing scheme for the cloud service," *Chinese Journal of Electronics*, vol.30, no.6, pp.1159–1166, 2021.

[6] "Bitinfocharts," Available at: *https://bitinfocharts.com/*, 2022.

[7] Y. Chen, Y. Lu, L. Bulysheva, *et al.*, "Applications of blockchain in industry 4.0: A review," *Information Systems Frontiers*, 2022.

[8] X. Q. Wang, C. P. Wang, K. Zhou, *et al.*, "ESS: An efficient storage scheme for improving the scalability of Bitcoin Network," *IEEE Transactions on Network and Service Management*, vol.19, no.2, pp.1191–1202, 2022.

[9] BitcoinCore, "Bitcoin core version 0.11.0 released," Available at: *https://bitcoin.org/en/release/v0.11.0*, 2015.

[10] A. I. Sanka and R. C. C. Cheung, "A systematic review of blockchain scalability: Issues, solutions, analysis and future research," *Journal of Network and Computer Applications*, vol.195, article no.103232, 2021.

[11] M. T. Xu, G. R. Feng, Y. L. Ren, *et al.*, "On cloud storage optimization of blockchain with a clustering-based genetic algorithm," *IEEE Internet of Things Journal*, vol.7, no.9, pp.8547–8558, 2020.

[12] K. Zhou, C. P. Wang, X. Q. Wang, *et al.*, "A novel scheme to improve the scalability of bitcoin combining IPFS with block compression," *IEEE Transactions on Network and Service Management*, vol.19, no.4, pp.3694–3705, 2022.

[13] I. T. Chou, H. H. Su, Y. L. Hsueh, *et al.*, "BC-Store: A scalable design for blockchain storage," in *Proceedings of the 2nd International Electronics Communication Conference*, Singapore, pp.33–38, 2020.

[14] D. Mitra and L. Dolecek, "Patterned erasure correcting codes for low storage-overhead blockchain systems," in *Proceedings of the 53rd Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, CA, USA, pp.1734–1738, 2019.

[15] X. D. Qi, Z. Zhang, C. Q. Jin, *et al.*, "A reliable storage partition for permissioned blockchain," *IEEE Transactions on Knowledge and Data Engineering*, vol.33, no.1, pp.14–27, 2021.

[16] H. H. Wu, A. Ashikhmin, X. D. Wang, *et al.*, "Distributed error correction coding scheme for low storage blockchain systems," *IEEE Internet of Things Journal*, vol.7, no.8, pp.7054–7071, 2020.

[17] M. H. Nasir, J. Arshad, M. M. Khan, *et al.*, "Scalable blockchains — A systematic review," *Future Generation Computer Systems*, vol.126, pp.136–162, 2022.

[18] Y. L. Zhao, B. N. Niu, P. Li, *et al.*, "Blockchain enhanced lightweight node model," *Journal of Computer Applications*, vol.40, no.4, pp.942–946, 2020. (in Chinese)

[19] H. Jin, R. K. Luo, Q. He, *et al.*, "Cost-effective data placement in edge storage systems with erasure code," *IEEE Transactions on Services Computing*, 2022.

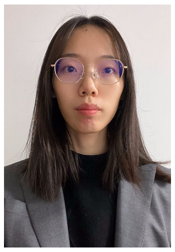
[20] M. Luby, "LT codes," in *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science*, Vancouver, BC, Canada, pp.271–280, 2002.

[21] A. Shokrollahi, "Raptor codes," *IEEE Transactions on Information Theory*, vol.52, no.6, pp.2551–2567, 2006.

[22] D. Perard, J. Lacan, Y. Bachy, *et al.*, "Erasure code-based low storage blockchain node," in *Proceedings of the 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, Halifax, NS, Canada, pp.1622–1627, 2018.

[23] S. Kadhe, J. Chung, and K. Ramchandran, "SeF: A secure fountain architecture for slashing storage costs in blockchains," *arXiv preprint*, arXiv: 1906.12140, 2019.

SHI Dongxian was born in 1983. He received the M.S. degree in computer software and theory from the Zhejiang University of Technology, Hangzhou, China, in 2010. He is currently pursuing the Ph.D. degree in cyberspace security with the Hangzhou Dianzi University, China. He is also an Assistant Professor with the Zhejiang Institute of Economics and Trade. His research interests include deep learning, intrusion detection, and blockchain security.
(Email: wildstone@hdu.edu.cn)



WANG Xiaoqing was born in 1995. She is currently pursuing her master's degree with the Zhejiang University of Technology, Hangzhou, China. Her research interest includes blockchain storage, networks security, and cloud computing.
(Email: 2111912087@zjut.edu.cn)

XU Ming was born in 1970. He received the Ph.D. degree from Zhejiang University in 2004. He is currently a Professor of Hangzhou Dianzi University. His research interests include digital forensics and network security.
(Email: mxu@hdu.edu.cn)

KOU Liang was born in 1986. He received the Ph.D. degree from Harbin Engineering University, Harbin, China, in 2019. Now he is a Lecturer in the Cyberspace College, Hangzhou Dianzi University, China. His current research interests include malware detection, network traffc abnormal traffc detection of SDN. He has published more than 10 academic papers in important journals and conferences.
(Email: kouliang@hdu.edu.cn)

CHENG Hongbing (corresponding author) was born in 1979. He received the Ph.D. degree from the Nanjing University of Posts and Telecommunications, and completed post-doctoral research in the State Key Laboratory of New Software Technology of Nanjing University. He is currently a Professor in college of computer, Zhejiang University of Technology and has published numerous research papers in high-quality international journals and conferences. Prof. Cheng served as an Invited Editor of several international journals in some international conferences; and has been invited to give keynote speeches and chair committees, reviewed papers for many international journals and conferences. His research interests include blockchain, cryptography, and information security.
(Email: chenghb@zjut.edu.cn)