A Binary Matrix-based Data Representation for Data Compression in Blockchain

Abdelkader Laouid
LIAP Laboratory, University of El Oued, Algeria
Email: abdelkader-laouid@univ-eloued.dz

Kara Mostefa
LIAP Laboratory, University of El Oued, Algeria
Email: karamostefa@univ-eloued.dz

Mohammed Al-Khalidi
Manchester Metropolitan University, United Kingdom
Email: m.al-khalidi@mmu.ac.uk

Khaled Chait
LIAP Laboratory, University of El Oued, Algeria
Email: chait-khaled@univ-eloued.dz

Mohammad Hammoudeh
King Fahd University of Petroleum & Minerals, KSA
Email: mohammad.hammoudeh@kfupm.edu.sa

Ahmed Aziz
Tashkent state university of Economics, Uzbekistan
Email: a.mohamed@tsue.uz

Abstract—Blockchain relies on storing and verifying a large volume of data across multiple nodes, making efficient data compression techniques crucial. By reducing the size of data, compression techniques enable more data to be stored within the limited space constraints of the blockchain networks. Furthermore, compressed data consumes less bandwidth for transmission and enhances the overall performance of blockchain networks by reducing the time and resources needed for data storage and retrieval. To overcome this issue, this paper presents a new data representation approach to enable efficient storage and management of diverse data types on the blockchain, ensuring scalability, cost-effectiveness, and improved network efficiency. A binary matrix M of size $m \times n$ bits can be converted to two vectors H and V of sizes m' and n', respectively. The compression rate expressed by $(m'+n'+\mid Hash(M) \mid) \times 100/(m \times n)$ increases exponentially, i.e., 2^{λ} with λ depends on m and n); this makes the proposed technique is very effective in data size reduction. With a matrix, for example, $M = 512 \times 512$ bits, we achieve a rate of reduction equal to 96.42%. The original data can be recovered using H, V, and $Hash(M)$. The conversion from M to (H, V) is simple, which optimizes energy consumption for low-power devices. Meanwhile, the challenge of recovering the original data could be exploited in a blockchain process, where the mining consensus could be identified based on the node that recovered a predefined set of vectors. Furthermore, this technique ensures that data integrity checking is available only at the nodes with a massive computation capacity.

Index Terms—Data representation; Blockchain; Variety of resources computation; Distributed computing.

I. Introduction

In recent years, blockchain technology gained considerable interest and popularity not just as a secure and decentralized ledger of transactions that is tamper resistant, but also as a technology for data storage across a variety of applications [1]. Blockchain data storage presents several challenges. As the number of transactions and participants increases, the amount of data that needs to be stored on the blockchain increases exponentially, posing scalability challenges. The size of the blockchain grows over time, making it difficult for new participants to join the network and synchronize with the existing data. Additionally, storing large volumes of data on the blockchain is expensive, especially if the blockchain uses a consensus mechanism that requires every participant to store a complete copy of the blockchain. The issue of storage limitation can lead to slower transaction processing and the need for more expensive hardware.

The block size refers to the maximum amount of data that can be stored in a single block. This size of the block is important in determining a blockchain network's transaction capacity and performance. Large block sizes can increase network node storage requirements and demand higher bandwidth usage. This makes it difficult for resource-constraint nodes to participate in the network, potentially leading to further centralization [2], [3].

Data representation is an aspect of data management which greatly affect the usability and accessibility of the data [4], [5]. It also significantly impacts the memory size required to store and transmit data. Different data representation methods can result in vastly different storage requirements, with some methods being more efficient than others [6]. Data representation also has a direct impact on the block size. Binary data representation is often used in blockchain to represent transaction data as it can be efficiently processed and transmitted across the blockchain network. Each transaction in a blockchain is represented as a set of binary data, which includes information such as the sender and the receiver addresses, the transaction data, and the transaction timestamp. A collection of transactions verified and added to the blockchain network form a block [7], [8].

Another factor that affects block size is the use of compression algorithms [9]. Compression algorithms reduce the data size by removing redundant or unnecessary information. This can result in significant reductions in

block size [10]. Each data representation type requires different compression methods to optimize storage and transmission efficiency. However, compression algorithms can also increase the computational overhead required to process the data, which can affect the overall efficiency of the blockchain network. Therefore, the choice of compression algorithm for a given type of data representation depends on various factors such as the data characteristics, the desired level of compression, the intended use of the compressed data, and the processing power available for compression and decompression [11], [12].

One way to organize and manipulate data in a structured manner is the matrix representation which is efficient for storing large datasets. Matrix representation helps identify patterns and relationships in the data; therefore, redundancies can be identified and removed easily, reducing the amount of storage space needed [13]. Blocks can be represented in matrices, where each row of the matrix could represent a transaction within the block, while each column represents a feature or variable of the transaction. For example, the columns could include information such as the sender address, recipient address, timestamp, transaction amount, and transaction fee.

This paper proposes a new method of presenting and storing information in blockchain to reduce the size of data. The solution is reached by converting a binary matrix M of size $m \times n$ bits to two vectors H and V of sizes m' and n', respectively. The proposed method represents data in a way where the size is much less than traditional storage. The original matrix can be recovered using H, V, and the hash of the matrix $Hash(M)$. This proposed method of compressing large data helps in efficient data transfer, use, and storage, especially for devices with limited hardware resources.

The rest of the paper is structured as follows. Section II provides a review of related work in the field of reduced data in the blockchain. SectionIII presents the proposed technique and its underlying concepts. Section IV discusses the efficiency analysis and shows the feasibility of the proposed technique. Finally, Section V concludes the paper and discusses future directions for research.

II. Related Work

Blockchain technology can potentially revolutionize various industries and applications, such as secure messaging, and supply chain management [14]. However, it must address several challenges to achieve its full potential, including scalability, privacy, and efficient storage. Those challenges emerge because blockchain-based systems generate vast amounts of data, making storing and transmitting this information challenging. This challenge is particularly significant for resource-constrained devices like those in the Internet of Things (IoT) and smart homes [15].

Various propositions were made to address the aforementioned issues. Some propose solutions that reduce the size of the data, such as summarizing and compressing blocks, storing bytecode off-chain, segmenting the blockchain, and using distributed storage systems to bypass storing liabilities. Others suggested strategies enabling nodes to store only part of the blockchain, such as deleting part of the blockchain or designing a semi-full node. These solutions aim to reduce the storage requirements and increase the throughput of the blockchain network, making it more suitable for data-heavy applications and expanding the number of nodes participating in the network.

Some techniques involve creating summary blocks and compressing them to increase space savings and make it easier for nodes to verify transactions. The proposed method in [9] is applied to the Bitcoin blockchain, and the space-saving is calculated by comparing the original block size with the summarized and compressed block sizes. The experiment results show that the space-saving for summary blocks is 22.318%, while the space-saving for compressed summary blocks is 78.104%.

To help reduce the size and growth of the blockchain, a system for the Ethereum blockchain that moves the bytecode of a contract creation transaction off-chain is proposed [16]. It replaces contract creation transaction data with hashes that identify a file in the InterPlanetary File System (IPFS). The proposed system retains the assurance provided by blockchain while reducing network traffic under certain conditions.

The authors of [17] proposed a segment blockchain approach that allows nodes to only store a copy of one blockchain segment. The proposed system uses Proof-of-Work (PoW) as a membership threshold to limit the number of nodes taken by an adversary. The system can sustain a $(AD/n)^m$ failure probability when the adversary has no more than AD number of nodes and every segment is stored by m number of nodes. However, the proposed segment blockchain system can fail if an adversary stores all copies of a segment, causing a permanent loss of the segment.

Sohan et al. [18] proposed a method using a distributed storage system called IPFS to bypass storage liabilities and increase throughput. Moreover, the proposed method employs a dual-blockchain approach that adds references of the main block to the ledger instead of the original block, preserving the core features of the blockchain.

To tackle the data bloating problem in blockchain systems, Chou et al. [19] presented the BC-Store framework that deploys a data accessing model on an IPFS cluster system to classify hot and cold blockchain data. Hot data is stored in the local cache, whereas cold data is stored in the IPFS cluster, substantially shortening the blockchain's initial synchronization time and saving a considerable amount of data storage.

Researchers have also proposed several lightweight blockchain solutions that can reduce the computational and storage requirements of blockchain consensus proto-

cols. For instance, in [20], an improved PBFT blockchain consensus mechanism based on a reward and punishment strategy was proposed to achieve a lightweight blockchain. Furthermore, a blockchain storage optimization scheme based on the RS erasure code was introduced to reduce the storage overhead of the blockchain.

Privacy and security of transaction aspects are so important in the blockchain. While immutability makes blockchain systems secure, it also raises concerns about privacy. Several research efforts proposed privacy-preserving solutions to protect the sender and receiver's identities and the transaction amount's confidentiality [21]. For example, in [22], the authors proposed a new blockchain ring confidential transaction protocol (RingCT3.0) that can protect the privacy of the sender's identity, the recipient's identity, and the confidentiality of the transaction amount.

III. Specifications of the New Data Representation Technique

In this Section, we present a detailed explanation of the proposed technique, which aims to present the data in a very small way. The node converts the matrix that contains the data according to the proposed method and then sends it. The conversion process is very easy and fast regardless of the size of the matrix; also the converted output is small in size compared to the original data size.

A. Representation of the Proposed model

In this section, we present the specifications of a new data representation technique suitable for data compression on the blockchain. This technique extracts two vectors, horizontal H and vertical V which replace the matrix. Suppose we have a matrix M containing m rows and n columns. The first vector H is a vector containing m values, the first value is the number of "1" in the first row of the matrix, the second value is the number of "1" in the second row, and so on ... until the last value, that is the number of "1" in the last row of the matrix.

Similarly, for the second vector V, it is a vector that contains n values, the first value is the number of "1" in the first column of the matrix, the second value is the number of "1" in the second column, and so on ... until the last value, that is the number of "1" in the last column of the matrix.

Representing the information in this way is insufficient to retrieve it from the cloud because we can find two different matrices A and B with the same vector representation, i.e., $H_A = H_B$ and $V_A = V_B$. Therefore, in the proposed model, we add the hash of the matrix to represent data uniquely, where $Hash_A \neq Hash_B$. The following example illustrates a demonstration in a simple way.

$$A = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

with $H_A = (2, 2, 2, 2)$; $V_A = (2, 1, 2, 3)$.
and

$$B = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

with $H_B = H_A = (2, 2, 2, 2)$; $V_B = V_A = (2, 1, 2, 3)$.

B. Proposed Model: Recuperation

After representing a matrix M as mentioned in Subsection III-A, the cloud can use its available computing power to recover the original matrix based on H_M, V_M, and $Hash_M$. The currently used algorithm in the proposed model is based on searching exhaustively according to the possibilities that exist. When the search starts from a zeros matrix with m rows and n columns, then the existing possibilities are tested based on the number of 1 in the two vectors H_M and V_M. For each matrix possibility M_i is calculated, the cloud calculates its hash and compares it with $Hash_M$. Algorithm 1 gives a general illustration of this procedure.

Algorithm 1 Recover matrix algorithm

Require: $H_M, V_M, Hash_M$
Ensure: $matrixM$

```
1: function RecM
2:     M_0 ← zeros
3:     for for each possible composition of the number of 1 do
4:         form M_i
5:         compute H_i
6:         compute V_i
7:         compute Hash_i
8:         if H_i = H_M and V_i = V_M and Hash_i = Hash_M then
9:             M ← M_i
10:            exit
11:        end if
12:    end for
13:    return M
14: end function
```

Now, we discuss the number of possible matrices (NPM) that the algorithm 1 will extract from H and V.

Lemma 1. If $size(M) = n \times m$ then $NPM_{n,m} = \prod_{i=1}^{m} NPR_i$ where m is the number of rows and NPR_i denotes number of possibilities in the row i.

We give $NPR_i = \alpha \times (\alpha + 1) \times (2 \times \alpha + 4)/12$ where $\alpha = n - x + 1$

Proof. The $RecM$ algorithm 1 will not try all possibilities from 'zeros' to 'ones'; in this case, the number of possibilities is $2^{n \times m}$. This is because the number of possible values for the first line is 2^n if the first line

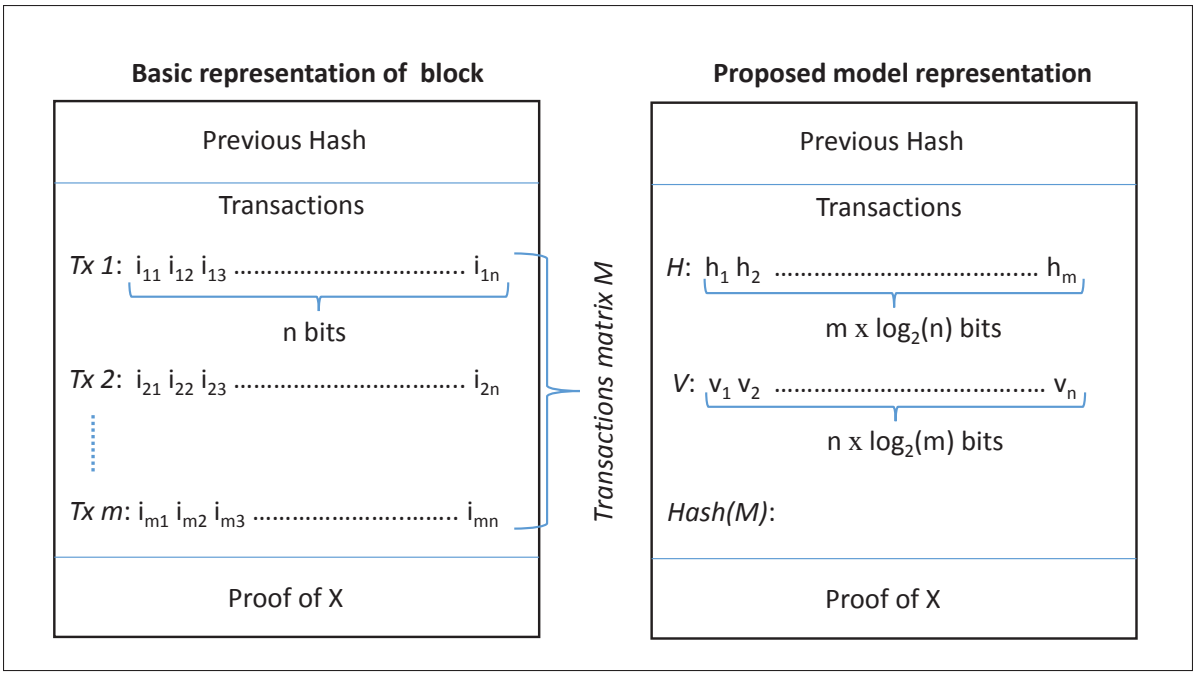

Fig. 1. Simplified block representations: basic and ours, where h_i denotes the number of 1 in row i $(tx : i)$ and v_i denotes the number of 1 in column i

size is equal to n, and if the matrix has m rows, then $NPM = 2^n \times 2^n \ldots 2^n = 2^{n \times m}$.

The $RecM$ algorithm 1 will exploit the number of 'ones' in each line to reduce the total number of possibilities, thus greatly reducing the number of trials.

Suppose the number of "1" is x with $x < n$ where n is the size of a row; the first possibility is to position the x bits of "1" successively (e.g. 11100000000 where $x = 3$ and $n = 11$). The next possibility is to move the last "1" (11010000000; then 11001000000 ...) etc.

Therefore, we have $(n - x + 1)$ possibilities. Next, we move the "1" before the last one and do the same with the last "1" as the first step (10110000000; 10101000000; 10100100000), so there are $(n - x + 1) - 1$ possibilities.

In the third step, we do the same thing with the first "1", we will finally find that NPR is equal to the sum of the series $1+(1+2)+(1+2+3)\cdots+(1+2+3+4+\cdots+\alpha)$ where α is $(n - x + 1)$ and x denotes the number of "1" in the first row.

We have $s_i = \sum_{i=1}^{i} i = i \times (i + 1)/2 = (i^2 + i)/2$.

Therefore, $NPR = \sum_{i=1}^{\alpha} s_i = 1/2 \times (\sum_{i=1}^{\alpha} i^2) + \sum_{i=1}^{\alpha} i$.

$NPR = 1/2 \times (\alpha \times (\alpha+1) \times (2 \times \alpha+1)/6) + \alpha \times (\alpha+1)/2$.

So, $NPR = \alpha \times (\alpha + 1) \times (2 \times \alpha + 4)/12$.

□

For example, consider a matrix M of size 6×6. In an absolutely exhaustive search without using H and V, the number of possibilities in the first line $NPR_1 = 2^6$; so, $NPM = 2^6 \times 2^6 \ldots 2^6 = (2^6)^6 = 2^{36} = 68719476736$.

In our proposal using H and V and according to Lemma 1, if we assume that the number of "1" equals 3, $x = 3$ (the average), this gives $\alpha = 6 - 3 + 1 = 4$ and $NPR = 4 \times (4 + 1) \times (2 \times 4 + 4)/12 = 20$; so $NPM = 20^6 = 64000000$.

C. New Block Size

Among the most important application of the proposed technique is Blockchain. The blockchain is made up of a group of blocks and each block is a set of transactions, then, these transactions can be considered a matrix of data where each line is one transaction or more.

Figure 1 shows the matrix design (Basic representation of block) containing m transactions, each transaction size is n bits; $i_{\alpha,\beta} = \{0, 1\}, \alpha \in [1, m]$ and $\beta \in [1, n]$. Using the proposed method, the block representation changes as shown on the right of Figure 1 (Proposed model representation). Since h_i denotes the number of "1" in row i and the size of row i equals n bits, we need $log_2(n)$ bits to represent each h_i.

$$| H | = m \times log_2(n) \tag{1}$$

and

$$| V | = n \times log_2(m) \tag{2}$$

and

$$| M | = | H | + | V | + | Hash_M | \tag{3}$$

If any party wants to view the transactions, it will be obtained by calculation from $H, V, and Hash_M$.

TABLE I
Limitations and challenges related to blockchain size and consensus.

Limitations and Challenges of Blockchain		
Category	Limitation/Challenge	Description
Size	Limited Scalability	The blockchain size increases with every new transaction, which can limit the number of transactions that can be processed at any given time.
	High Storage Costs	Every node on the network has to store a copy of the entire blockchain, resulting in high storage costs.
Consensus	Slow Transaction Processing	The consensus mechanism used to validate transactions can lead to longer transaction times.
	Energy Consumption	Some consensus mechanisms require a large amount of energy, which can lead to high costs and environmental concerns.

IV. Performance Analysis

Reducing the data size in blockchain can lead to improved performance (including storage utilization, bandwidth and speed) and scalability. By reducing the amount of data that needs to be stored and processed, blockchain applications can become faster, storage efficient, and more scalable. Table I provides a taxonomy of limitations and challenges associated with using blockchain technology, specifically focusing on issues related to size and consensus. The first column of the table identifies the two main categories: Size and Consensus. The second column describes the specific limitations and challenges within each category, while the third column briefly explains each limitation or challenge. Two main limitations are observed in the size category: limited scalability and high storage costs. These limitations arise because the size of the blockchain increases with every new transaction, limiting the number of transactions that can be processed at any given time, where every node on the network has to store a copy of the entire blockchain, resulting in high storage costs.

On the other hand, in the consensus category, two crucial challenges could be identified: slow transaction processing and energy consumption. These challenges arise because the consensus mechanism used to validate transactions can lead to longer transaction times. Some consensus mechanisms require a lot of energy, leading to high costs and environmental concerns.

To this end, we discuss such practical examples to explain further the efficiency of the proposed method and how it works. Suppose that $M = 100 \times 100$, i.e., $n = m = 100$, the largest value that v_i can take is 100, where the column contains one hundred "1", the number "100" is written in binary as 1100100, so the size of each v_i is equal to 7, while the number "128" can be also represented in 7 bits (which means 128 rows or transactions).

We can say that $size(V) = 700$ bits for a matrix of 100×100 and $size(V) = 700$ bits for a matrix of 128×128.

Therefore, it is desirable that the number of rows (column size) are not random so that the proposed method is more effective. In other words, the number of rows (namely transactions) and the number of columns (row size) must be equal to 2^x and 2^y, respectively, the following study shows that (Table II and Figure 2).

Now suppose that $M = 16 \times 16$ and therefore the size of M is equal to 256 bits; since there are 16 columns (respectively 16 rows), the vector H (respectively vector V) contains 16 values, each h_i (respectively v_i) is represented in 4 bits, which gives a size of H (respectively V) equal to $4 \times 16 = 64$ bits. By adding the matrix hash, the total data size is equal to $(4 \times 16) \times 2 + 160 = 288$ bits. We notice here that the application of the proposed method did not help in reducing the data size.

By supposing that $M = 32 \times 32$ and therefore the size of M is equal to 1024 bits; since there are 32 columns (respectively 32 rows), the vector H (respectively vector V) contains 32 values, each h_i (respectively v_i) is represented in 5 bits, which gives a size of H (respectively V) equal to $5 \times 32 = 160$ bits. By adding the matrix hash, the total data size is equal to $(5 \times 32) \times 2 + 160 = 480$ bits. Now, we notice that the application of the proposed method helps in reducing the data size (480 bits vs. 1024 bits).

With $M = 64 \times 64$, the size of M is equal to 4096 bits; since there are 64 columns (respectively 64 rows), the vector H (respectively vector V) contains 64 values, each h_i (respectively v_i) is represented in 6 bits, which gives a size of H (respectively V) equal to $6 \times 64 = 384$ bits. By adding the matrix hash, the total data size equals $(6 \times 64) \times 2 + 160 = 928$ bits. In this example, we see the rate of reducing size is increasing, 53.12 with 32 bits and 77.34 with 64 bits.

If $M = 128 \times 128$, the size of M is equal to 16384 bits; since there are 128 columns (and 128 rows), the vector H (and vector V) contains 128 values, each h_i (respectively v_i) is represented in 7 bits, which gives a size of H (also V) equal to $7 \times 128 = 896$ bits. By adding the matrix

TABLE II
A comparative summary of basic and our reduced block sizes

square matrix of x bits	16	32	64	128	256	512
basic size (bs)	256	1024	4096	16384	65536	262144
reduced size (rs) with SHA1 (160 bits)	$(16\times4)\times2+160=288$	$(32\times5)\times2+160=480$	$(64\times6)\times2+160=928$	$(128\times7)\times2+160=1952$	$(256\times8)\times2+160=4256$	$(512\times9)\times2+160=9376$
rate (bs/rs)	0.88	2.13	4.41	8.89	15.39	27.96
reducing rate	/	53.12 %	77.34 %	88.08 %	93.50 %	96.42 %

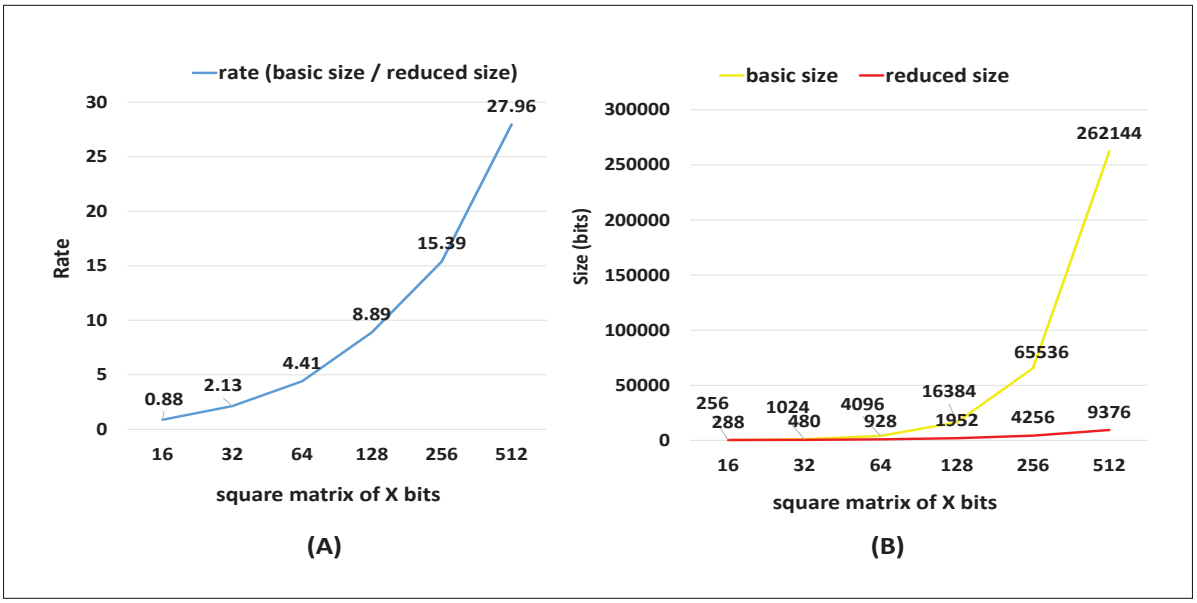

Fig. 2. Original and reduced block sizes comparison

hash, the total data size equals $(7\times128)\times2+160=1952$ bits. The reduction rate now achieves 88.08%.

With a matrix M of 256×256 bits, we achieve a rate of reduction equal to 93.50%; in the last test where $M=512\times512$ bits, we achieve a rate of reduction equal to 96.42%.

For M of 512×512 bits, Table III shows a comparison with Yu et al. [23] technique.

TABLE III
A comparative summary of reduced size techniques

technique	rate of reducing
Yu et al. [23]	35.5 %
Ours	96.42 %

In this proposal, data integrity verification in blockchain will not be done as classical verification. The verification relies on reconstructing the original data from vectors and hashes, which can require extensive computational resources. This process involves reconstructing the data in its entirety and thus can be computationally intensive and time-consuming. This can be considered inconvenient compared with others. However, in several domains, blockchain verification is a process that is rarely requested. Furthermore, with the huge computation of the mainframe machines, the reconstruction process could be done by these machines, and the verification is done by any network node. The main advantages and disadvantages of the proposed reduced size and computation-based verification technique are summarized in Table IV. The proposed technique takes care of the size aspect, and as for the security aspects, any lightweight encryption method can be chosen to encrypt H and V.

V. Conclusion

This manuscript presented a new approach to representing big data in a way where the size is much less than known traditional storage techniques. We consider that our proposal is a method of compressing big data in the blockchain, which helps in its transfer, use, and ease of storage, especially for devices with low power. Given the large data size of the blockchain, we claim it appropriate to exploit the proposed model for this pioneering storage technology. The provided explanation showed that our technique is easy to apply, as it does not require complex operations to compress data, which reduces the amount of energy consumed to do so. After analyzing the new method, the obtained values proved its

TABLE IV
The main Advantages and Inconveniences of the proposed technique

Advantages	Inconveniences
• Reduced size: The technique is based on extracting two vectors that replace the matrix, which can significantly reduce the size of the stored information. • Efficient storage: The technique efficiently stores information using the horizontal and vertical vectors, reducing storage costs. • Unique representation: The addition of the hash of the matrix in the proposed model allows for a unique representation of data, which helps to ensure its integrity and prevents data tampering.	• Computationally intensive: The integrity verification is based on the original data reconstruction, which requires a huge computation. • Limited to binary data: The technique is limited to binary data as it relies on the number of "1"s in each row and matrix column. • Possible hash collisions: While adding the hash of the matrix allows for a unique representation of data, there is still a possibility of hash collisions, potentially resulting in data corruption.

effectiveness, as we saw that the percentage of reducing the size is very large, especially when the original size of data is considerable; for a matrix $M = 512 \times 512$ bits, we achieved a rate of reduction equal to 96.42%. In other words, the effectiveness of the proposed method increases as the data size to be represented increases. In future work, we intend to exploit the proposed technique to create a consensus system based on the original data recovery. This is done by diffusing the computed matrix (H, V, and $Hash_M$) problem in the network, after that, the nodes compete to obtain the original matrix M. Furthermore, we propose to define a central system that stores the original data where any integrity check should be passed by this system. Overall, the proposed technique opens many advantages and solutions.

References

[1] M. I. Khalid, I. Ehsan, A. K. Al-Ani, J. Iqbal, S. Hussain, S. S. Ullah et al., "A comprehensive survey on blockchain-based decentralized storage networks," IEEE Access, 2023.

[2] J. W. Heo, G. S. Ramachandran, A. Dorri, and R. Jurdak, "Blockchain storage optimisation with multi-level distributed caching," IEEE Transactions on Network and Service Management, vol. 19, no. 4, pp. 3724–3736, 2022.

[3] K. Zhou, C. Wang, X. Wang, S. Chen, and H. Cheng, "A novel scheme to improve the scalability of bitcoin combining ipfs with block compression," IEEE Transactions on Network and Service Management, 2022.

[4] T. Alsbouí, M. Hammoudeh, Z. Bandar, and A. Nisbet, "An overview and classification of approaches to information extraction in wireless sensor networks," in Proceedings of the 5th International Conference on Sensor Technologies and Applications (SENSORCOMM'11), vol. 255, 2011.

[5] M. Hammoudeh, R. Newman, C. Dennett, S. Mount, and O. Aldabbas, "Map as a service: a framework for visualising and maximising information return from multi-modal wireless sensor networks," Sensors, vol. 15, no. 9, pp. 22 970–23 003, 2015.

[6] A. Voisard and B. David, "A database perspective on geospatial data modeling," IEEE Transactions on Knowledge and Data Engineering, vol. 14, no. 2, pp. 226–243, 2002.

[7] A. Hafid, A. S. Hafid, and M. Samih, "Scaling blockchains: A comprehensive survey," IEEE Access, vol. 8, pp. 125 244–125 262, 2020.

[8] Y. Rajendra, S. Sahu, V. Subramanian, and S. K. Shukla, "Storage efficient blockchain models for constrained applications," Cluster Computing, pp. 1–19, 2022.

[9] U. Nadiya, K. Mutijarsa, and C. Y. Rizqi, "Block summarization and compression in bitcoin blockchain," in 2018 International Symposium on Electronics and Smart Devices (ISESD). IEEE, 2018, pp. 1–4.

[10] M. Shafay, R. W. Ahmad, K. Salah, I. Yaqoob, R. Jayaraman, and M. Omar, "Blockchain for deep learning: review and open challenges," Cluster Computing, vol. 26, no. 1, pp. 197–221, 2023.

[11] Z. Guo, Z. Gao, Q. Liu, C. Chakraborty, Q. Hua, K. Yu, and S. Wan, "Rns-based adaptive compression scheme for the block data in the blockchain for iiot," IEEE Transactions on Industrial Informatics, vol. 18, no. 12, pp. 9239–9249, 2022.

[12] S. Lu, Q. Xia, X. Tang, X. Zhang, Y. Lu, and J. She, "A reliable data compression scheme in sensor-cloud systems based on edge computing," IEEE Access, vol. 9, pp. 49 007–49 015, 2021.

[13] L. Kuang, F. Hao, L. T. Yang, M. Lin, C. Luo, and G. Min, "A tensor-based approach for big data representation and dimensionality reduction," IEEE transactions on emerging topics in computing, vol. 2, no. 3, pp. 280–291, 2014.

[14] G. Epiphaniou, P. Pillai, M. Bottarelli, H. Al-Khateeb, M. Hammoudeh, and C. Maple, "Electronic regulation of data sharing and processing using smart ledger technologies for supply-chain security," IEEE Transactions on Engineering Management, pp. 1–15, 2020.

[15] M. Walshe, G. Epiphaniou, H. Al-Khateeb, M. Hammoudeh, V. Katos, and A. Dehghantanha, "Non-interactive zero knowledge proofs for the authentication of iot devices in reduced connectivity environments," Ad Hoc Networks, vol. 95, p. 101988, 2019.

[16] R. Norvill, B. B. F. Pontiveros, R. State, and A. Cullen, "Ipfs for reduction of chain size in ethereum," in 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). IEEE, 2018, pp. 1121–1128.

[17] Y. Xu and Y. Huang, "Segment blockchain: A size reduced storage mechanism for blockchain," IEEE Access, vol. 8, pp. 17 434–17 441, 2020.

[18] M. S. H. Sohan, M. Mahmud, M. B. Sikder, F. S. Hossain, and M. R. Hasan, "Increasing throughput and reducing storage bloating problem using ipfs and dual-blockchain method," in 2021 2nd International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST). IEEE, 2021, pp. 732–736.

[19] I.-T. Chou, H.-H. Su, Y.-L. Hsueh, and C.-W. Hsueh, "Bc-store: A scalable design for blockchain storage," in Proceedings of the 2nd International Electronics Communication Conference, 2020, pp. 33–38.

[20] C. Li, J. Zhang, X. Yang, and L. Youlong, "Lightweight blockchain consensus mechanism and storage optimization for resource-constrained iot devices," Information Processing & Management, vol. 58, no. 4, p. 102602, 2021.

[21] S. Moffat, M. Hammoudeh, and R. Hegarty, "A survey on ciphertext-policy attribute-based encryption (cp-abe) approaches to data security on mobile devices and its application to iot," in Proceedings of the International Conference on Future Networks and Distributed Systems, 2017.

[22] T. H. Yuen, S.-f. Sun, J. K. Liu, M. H. Au, M. F. Esgin, Q. Zhang, and D. Gu, "Ringct 3.0 for blockchain confidential transaction: Shorter size and stronger security," in Financial Cryptography and Data Security: 24th International Conference, FC 2020, Kota Kinabalu, Malaysia, February 10–14, 2020 Revised Selected Papers 24. Springer, 2020, pp. 464–483.

[23] B. Yu, X. Li, H. Zhao, and T. Zhou, "A scalable blockchain

network model with transmission paths and neighbor node subareas," Computing, pp. 1–25, 2021.