Received: 23 March 2022 | Revised: 29 May 2022 | Accepted: 13 July 2022

DOI: 10.1002/cpe.7344

WILEY

RESEARCH ARTICLE

# BAC: A block alliance consensus mechanism for the mine consortium blockchain

Yingsen Wang[1] | Yulan Ma[1] | Yan Qiang[1] | Juanjuan Zhao[1] | Yi Li[1] | Keqin Li[2]

[1]College of Information and Computer Science, Taiyuan University of Technology, Taiyuan, China

[2]State University of New York, Department of Computer Science, New Paltz, New York, USA

**Correspondence**
Yan Qiang, College of Information and Computer Science, Taiyuan University of Technology, Taiyuan, China.
Email: qiangyan@tyut.edu.cn

**Funding information**
National Natural Science Foundation of China, Grant/Award Numbers: 61872261, 61972274

## Abstract

Safety is an important issue in the mining industry and the Internet of Things (IoT) plays an important role to enhance the safety of the underground working environment. The IoT is used to transfer data generated by underground sensors to cloud storage for further processing. However, third-party platforms are often a target for cyber attacks. Serious mining accidents might occur if the data were tampered with. In the overground scenario, the security of trading data is also an important issue. The Mine Consortium Blockchain (MCB) is proposed to solve the above problems. The MCB avoids the risk of centralized storage and enables data security, provenance and transparency by taking advantage of blockchain technology. The MCB platform ensures that only designated participants can process mineral data. Any violation is immutably recorded in the MCB and is easily traced back by other participants. Classical consensus mechanisms as the core technology of the blockchain cannot be directly and appropriately applied to the mining industry. A Block Alliance Consensus (BAC) mechanism, which is suitable for all consortium blockchain scenarios, is proposed to improve the performance of the MCB. In addition, the block structure of the underground sensor data is optimized: the blocks only contain a hash of the sensor data and the data being stored in the cloud. The efficiency of the BAC is demonstrated by simulation experiments where the performance of the BAC consensus mechanism is compared with with the performance of classical consensus mechanisms. The MCB and the BAC consensus mechanism were also implemented on Hyperledger Fabric. Finally the Hyperledger Caliper evaluation tool was used to evaluate the performance of the system.

KEYWORDS

block alliance consensus, block structure, data security, mine consortium blockchain, mining industry

## 1 | INTRODUCTION

Mining is one of the earliest human activities succeeding agriculture and has always played a significant role since the beginning of our humankind civilization.[1] The mining industry has a vital responsibility toward the health, safety, legal, and social requirements of a country's economy and human resources.[2] Robust and efficient IT communication between the overground and underground mining is required to ensure the safety of the mining industry and provide information in real-time.[3,4] Mine Internet of Things (IoT) mainly refers to apply monitoring sensors to underground mining operations and environment (toxic gases, methane, humidity, transport, and machinery) to obtain underground monitoring information to ensure workers safety and production effectiveness.[5] Apart from IT communication in underground mining, overground Internet information mainly contains coal mine transportation and transaction data, mineral right transaction and price, audit logging, and media-related funds.[6,7]

However, there are many significant challenges in implementing these digital technologies in mining industries.[1] The mining industry is attractive to cyber-attacks due to its vital economic factor for a nation.[8] An increase in Internet of Things (IoT) technologies with thousands of connected devices in the mining industry increases the attack range and security vulnerability.[9,10] Access to the mining network and sensor nodes can be easily acquired by enemies and malicious attacks such as stealing, destroying or tampering with messages often occur.[11] Apart from external attacks, data tampering within the mining industry poses a serious threat to underground mining safety and national interests. For example, staff often tamper with sensor data to avoid shutdowns in coal production, which might result in serious accidents, or tamper with coal mine transaction data to obtain high profits.[1] It is evident that research on the security and accuracy of mineral data is becoming increasingly urgent.[12]

The application of blockchain to the mining industry has become a promising technology as it can help in curbing the penetration and disruption of cyber-attacks.[1] Unlike the centralized cloud servers paradigm in current IoT solutions,[13] blockchain is a decentralized, distributed and immutable ledger consists of irrevocable sequence of blocks.[14,15] Current IoT technologies are mostly prone to a single point of failure, if a cloud server breaks down, it may impact the whole network.[16,17] On the contrary, the attackers in blockchain must possess a majority of the mining power of the network in order to conduct a successful attack.[18,19] Furthermore, a decentralized network of blockchain can also easily address some limitations in IoT such as the remarkable cost of maintaining centralized clouds.[20]

The consensus mechanism is one of the most vital parts of the blockchain. It is the core technology but the bottleneck of blockchain. The consensus mechanism enables different nodes to agree on the new block waiting to be added to the blockchain. The inherent encryption characteristics of blockchain ensure those old data blocks that already existed in the blockchain can not be tampered with, while consensus methods ensure the validity of the new data block. The Byzantine problem is always the most difficult problem to solve in the distributed consensus protocol. Blockchain technology has come into public attention due to its use in the Bitcoin system. The Byzantine General Problem is a common challenge that decentralized computer systems must overcome. The analog used for the Byzantine Generals Problem basically goes like this: A group of generals of the Byzantine army camped with their troops around an enemy city. Communicating only by messenger, the generals must agree upon a common battle plan. However, one or more of them may be traitors who will try to confuse the others. The problem is to find an algorithm to ensure that the loyal generals will reach an agreement.[21] The POW consensus mechanism adopted in Bitcoin solves the Byzantine Generals problem through computing power. The probability of a Bitcoin miner mining a block is proportional to the relative hashing power of the miner. However, most of the consensus methods currently employed in blockchain require high-computational power and lack low latency so they are not suitable for practical resource-constrained IoT network.[20] So in this study, we propose a consortium blockchain structure of the mining industry and a data Block Alliance Consensus (BAC) algorithm to ensure the accuracy and reliability of the data and the security of the network.

The main contributions of our study are as follows:

(1) A distributed Mine Consortium Blockchain (MCB) structure is proposed, all the participants record and maintain the ledger together. The MCB is divided into underground and overground scenarios. The MCB-underground could prevent insiders from tampering with the data. The MCB-overground could simplify the trading process and monitor any trading data tampering in the mining industry.
(2) We propose the Block Alliance Consensus (BAC) consensus algorithm suitable for the consortium blockchain. The BAC is divided into BAC-underground and BAC-overground corresponding to the MCB-underground and MCB-overground, respectively. BAC does not rely on high computational power, which is very suitable for resource-constrained industrial consortium blockchain. We use a timeout mechanism and a Credit Incentive to identify specific nodes to add blocks to the block chain. The Block Consistency reduces the time complexity of BAC to $O(N)$ where $N$ is the number of nodes. BAC also avoids forking of the blockchain and accelerates the confirmation of blocks. In Bitcoin, the birth of a new block needs to be confirmed by at least six blocks, while BAC only needs the confirmation of two blocks. BAC applies to all consortium blockchain scenarios.
(3) The MCB introduces Credit Incentive without tokens used in the traditional blockchain. Experiments show that the Credit Incentive can make the BAC perform better and the nodes are motivated to conform to the rules of the MCB.
(4) We first simulate BAC through a blockchain dedicated simulator instead of traditional experiments to obtain more performance on blocks and nodes. Then we add the superior performed BAC to the consensus module, implement our MCB in Linux system and conduct several experiments based on the implementation. Experimental results show the effectiveness and efficiency of our proposed method and system.

The remainder of the paper is organized as follows. Section 2 presents related work. Section 3 introduces our MCB model. The block structure and the BAC consensus mechanism are proposed in Sections 4 and 5, respectively. Section 6 provides the performance evaluation of our BAC consensus mechanism and the MCB platform. The last section concludes our work and outlines future research.

## 2 | RELATED WORK

Blockchain is suited to the rigorous requirements of IoT networks for protecting the stored data from malicious attacks. It also provides a secure platform for all nodes (different devices and sensors) to communicate with each other without a third-party trusted platform or central server.[22] Blockchains can be divided into three categories based on how the user can access the ledger and participate in the consensus: public, private, and consortium.[23,24]

No single node could control the tamper-evident ledger in a public blockchain, but can be shared and supervised by all. The new block can be added to the blockchain as long as it is verified by all in the network.[25] Private blockchains are often implemented in companies for specific applications with permission to the system and dependent on a third-party platform which is opposite to the original idea of blockchain being decentralized.[24,26] While a private blockchain is maintained by a single institution, consortium blockchain is controlled by several companies all of which directly participate in the blockchain system.[25] Each node of the consortium blockchain can participate in and exit the network only after authorization and usually has a corresponding entity organization. The essence of a consortium blockchain is the private blockchain, the only difference between them is the number of governing institutions.[20] The most well-known consortium blockchain is the Hyperledger project which is a cooperation between many well known companies and hosted by the Linux foundation.[27]

There are already some works that have researched how to apply the blockchain to the mining industry. Reference 28 outlined several key applications of blockchain for the mining industry, such as data provenance, supply chain transparency, and the integration with ERP (Enterprise Resource Planning) systems.[29] used private blockchain with the consensus algorithm of the POW and Proof-of-Authority to ensure the reliability of transactions in the common information space of a mining enterprise.[30] presented a blockchain-based machine-to-machine communication architecture, which could be deployed directly to industrial devices.

The BFT originated from the Byzantine Generals Problem (BGP) proposed by Lamport.[21] The BGP is a common challenge that decentralized computer systems must overcome. The analog used for the BGP basically goes like this: A group of generals of the Byzantine army camped with their troops around an enemy city. Communicating only by messenger, the generals must agree upon a joint battle plan. However, one or more of them may be traitors who will try to confuse the others. The purpose is to find an algorithm to ensure that the loyal generals reach an agreement. However, the asynchronous BGP was not considered. The time threshold $t$ in the traditional BFT (BGP) is a fixed constant value. However, there is no concept of the time threshold in asynchronous systems. The approach taken by PBFT[31] is as follows: the $t$ in Practical Byzantine Fault Tolerance (PBFT) proposed after the traditional BFT will increase if the system times out. It ensures that no matter how considerable the system's delay is, PBFT can guarantee that nodes eventually reach a consensus as long as the delay does not increase indefinitely.

A blockchain-based IoT network relies on the consensus mechanism for different nodes to communicate with each other and agree on the validity of any transaction (sensors data or trading proposal of underground and overground). The most well-known consensus algorithm is Proof of Work (POW) used by Bitcoin, the first application of blockchain.[32] POW is a computationally expensive consensus method that requires different nodes in blockchain to solve a mathematical problem using a cryptographic hash function. The node who first solves this problem can append a block of transactions to the blockchain and other nodes could verify the block validity.[18]

Ethereum plans to employ Proof of Stake (PoS) which is the most prevalent consensus mechanism for cryptocurrencies instead of POW as its underlying consensus method.[33] But it currently uses the POW. A node is chosen to keep account based on its proportional stake, that is, its assets in terms of that cryptocurrency. As a result, POS requires much less computational resource than POW. A combination of POW and POS was proposed in Reference 34. The POW is applied for the initial assignment of stake in the early stage. The POS is employed to maintain the long-term security of the network when sufficient tokens are accumulated in the blockchain system. In the POS algorithm, the linear relationship between coinage and time is modified to a function with exponential decay so that the growth rate of the coin age approaches zero over time, thereby preventing currency accumulation.

POW and POS are two classic consensus methods for the application of public blockchain, but they are not prevalent for resource-constrained IoT networks due to their high latency and low throughput. The core problem to be solved by consensus mechanism is how to reach consensus when there are nodes doing evil. PBFT was proposed in Reference 31. The PBFT algorithm provides the fault tolerance if $f \leq (n-1)/3$ where $f$ is the number of faulty nodes and $n$ is the total number of nodes participating in the blockchain network. Thus PBFT requires $n \geq 3f + 1$ nodes.

All the nodes in PBFT should participate in the voting process to verify the new block and the consensus is reached as long as more than two-thirds of all nodes agree on that block. The PBFT can reach consensus quicker than POW and does not require tokens similar to POS.[35] PBFT is well-suited for consortium blockchains like the hyperledger projects for its high throughput, low latency, and low computational overhead—all of which are desirable for IoT applications.[15,36] However, PBFT is only suitable for small IoT networks for its high network overhead. And the faulty nodes can result in consensus failures or even data inconsistencies.[37,38]

## 3 | MINE CONSORTIUM BLOCKCHAIN

The MCB we proposed is applied for both underground and overground scenarios as Figure 1. The main part of the underground BAC is the Wireless Sensor Network. The information overground contains mostly mineral trading proposals. Cli is the client that initiates transaction requests. CA is the default certificate authority. CA manages the identity certificate of each entity (user) in the network. CA is responsible for the registration of all entity's identities in the blockchain, the issuance of digital certificates, and the renewal or revocation of certificates. All nodes that join the consortium blockchain must be registered and obtain a certificate issued by CA. Ledger (L) is a channel's chain and current state data which is maintained by each peer on the channel. The role of the channel is to realize the isolation of the business in the blockchain. A consortium blockchain has multiple channels, each channel represents a business. Members in the channel are organizations within the consortium blockchain, and an organization can join multiple channels. The essence of a channel is a private atomic broadcast channel divided and managed by sorting nodes, with the purpose of isolating the information in the channel so that entities outside the channel cannot access the information inside the channel, thus achieving privacy of the transaction. The channel is divided into the system channel and the application channel. The sorting node manages the application channel through the system channel, and the user's transaction information is delivered through the application channel for general users. WSN is the Wireless Sensor Network. SAGES is the Self-Advancing Goaf Edge Support Systems. It is a facility for the improvement of safety in underground coal mines. Both underground and overground information are published to the blockchain instead of the traditional centralized server. Users or staff can access the MCB blockchain system through their mobile phones to query or publish information related to the MCB.

We utilize the asymmetric encryption technology to guarantee the authenticity and validity of the message (transaction or block) between sender and receiver. The symmetric encryption algorithm uses the same secret key for encryption and decryption, while the asymmetric encryption

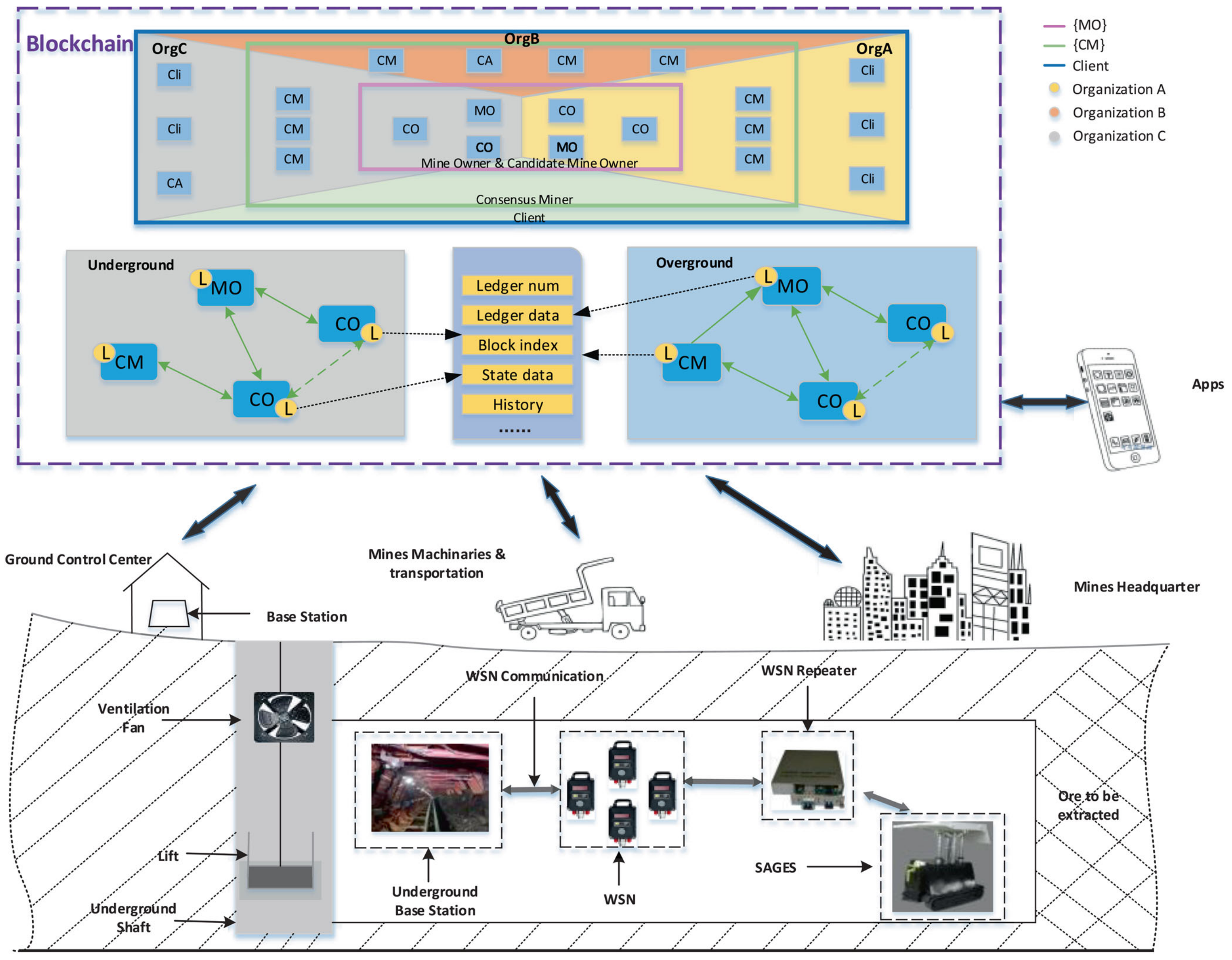


FIGURE 1 The constructure of the Mine Consortium Blockchain

algorithm requires two keys for encryption and decryption: the public and private keys. Only the corresponding private key can decrypt if a node encrypts data with a public key.

The MCB allows an entity to verify the record's validity without relying on a third-party platform like cloud computing or a bank. The nodes of the MCB is divided into three categories: Mine Owner (MO), Candidate Mine Owner (CO), and Consensus Miner (CM). The MCB is deployed in a consortium blockchain, that is, a permissioned blockchain (the Hyperledger in our evaluation part). Each node participating in the blockchain system needs to be licensed whereas nodes without permission can not join. This inherent property of the consortium blockchain initially ensures the security of the system. There is no "mining" or native crypto-currency required for consensus in MCB. The "Org" in Figure 1 can be interpreted as the organization that manages a series of cooperative enterprises. The MCB allows organizations to participate in multiple independent blockchain networks at the same time through channels, which provide effective infrastructure sharing while maintaining data and communication privacy. Sufficient independence between channels can help organizations separate their business from different competitors and unite independent organizations when necessary. Our main work focuses on the consensus mechanism between nodes, which is one of the most important properties along with encryption and smart contract. Two real examples are given in Figure 2. The MO, CO, and CM in our paper represents a computer or server in real life. The "Org" means the organization. All nodes and user accounts belong to an organization. The organization in the blockchain could be a company, an enterprise, or an association in the real world. For example, the departments responsible for data monitoring and processing and the regulatory agencies can all be regarded as organizations in the MCB. Each organization contains various types of nodes (CO, CM, CA etc.), and nodes refer to computers or servers. The client is a program that provides local services to customers. It is generally installed on an ordinary computer. Corresponding to the client is the server. The client accepts the service and the server provides the service. Commonly used clients include web browsers such as those used on the World Wide Web, and email clients when sending and receiving emails. QQ and WeChat can also be regarded as clients.

These two monitoring systems are based on the consortium blockchain. The controller, IP camera, sensor, and transformer station are used to collect and transmit the data to an industrial ethernet switch or server. The switch and server then transmit the data to designated nodes (MO, CO in the MCB) in the monitoring system. How to process data between nodes and how to prevent various types of data from being tampered with is the focus of our paper. The "blockchain" part in the purple wireframe in Figure 1 is the specific form of the node cluster in Figure 2.

Blockchain technology has come into prominence since Sat-oshi Nakamoto published a white paper titled "Bitcoin: A Peer-to-Peer Electronic Cash System." Blockchain is a distributed ledger that integrates distributed storage, peer to peer transmission, consensus mechanism, encryption

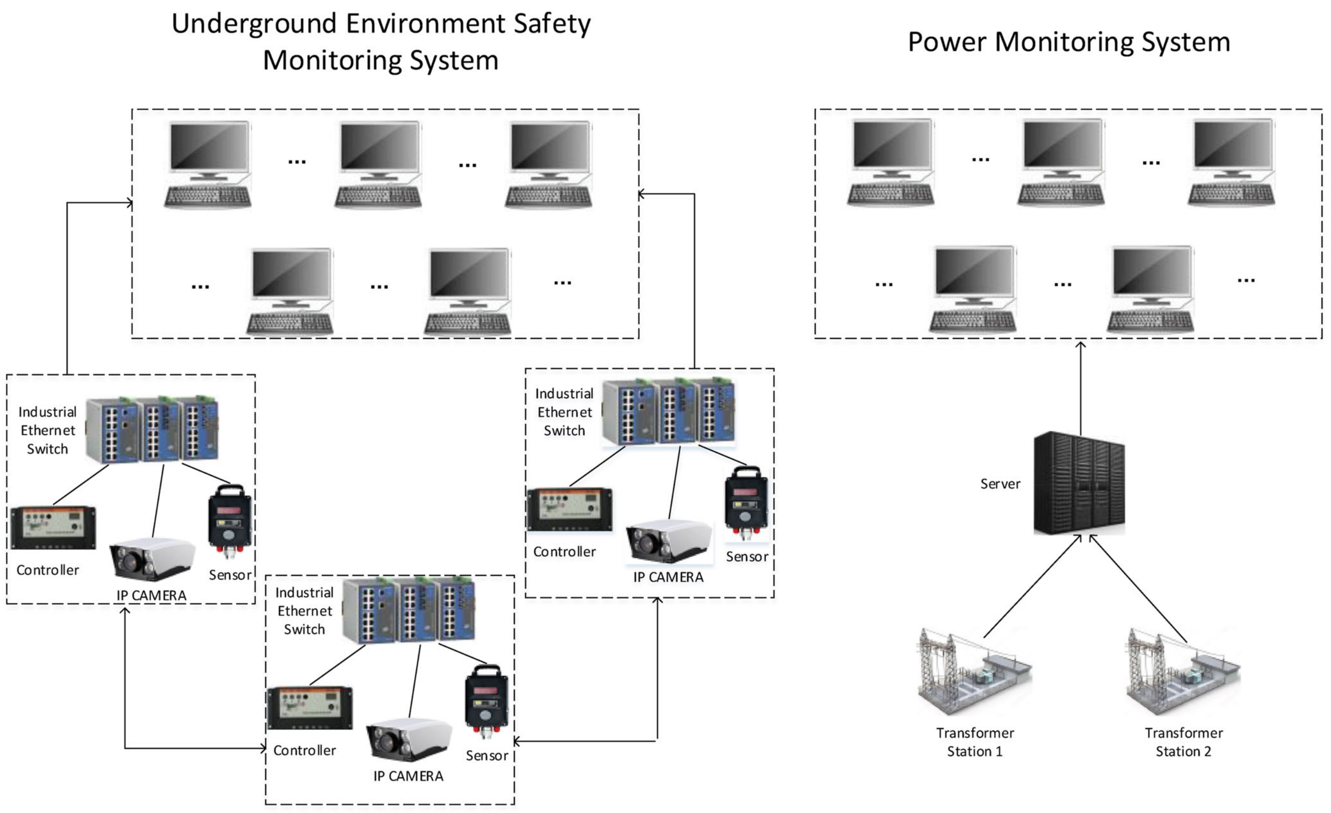


**FIGURE 2** Two real examples of the Mine Consortium Blockchain

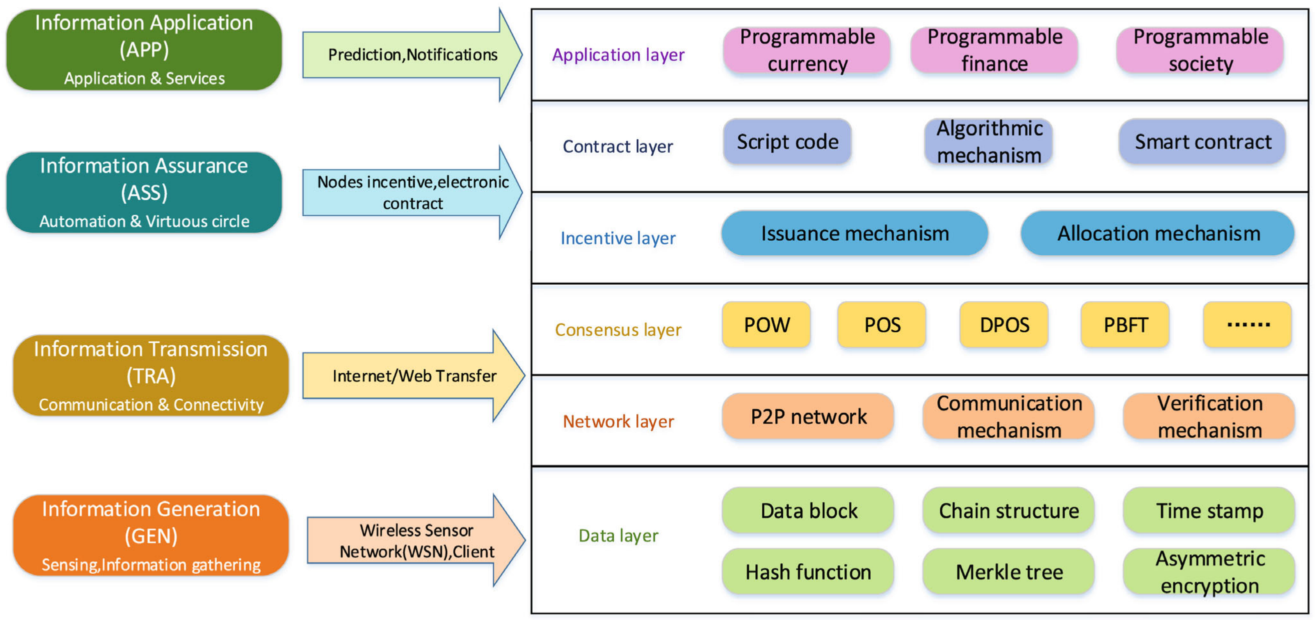


**FIGURE 3** The composition of the Mine Consortium Blockchain

algorithm and smart contract. The Bitcoin scripting language is not Turing-complete which means that Bitcoin transactions can conduct only relatively simple smart contracts. The essence of the blockchain is a decentralized database. The infrastructure of blockchain technology consists of data layer, network layer, consensus layer, incentive layer, contract layer, and application layer, as shown in Figure 3. The left side of the figure shows the traditional IoT building blocks.

The data layer contains the basic data and algorithms such as data encryption and timestamp. The network layer contains networking mode, data transmission protocol, etc. The peer-to-peer network is generally used to organize the whole nodes to participate in data verification and accounting. The consensus layer is the core of the blockchain, but it is also the bottleneck of performance such as throughput and latency. It mainly contains the consensus algorithm among distributed nodes, which is not only a way for distributed nodes to verify the transaction, but also a tamper-proof means for implementing the blockchain system. The incentive layer integrates the economic factors into the blockchain, and the smart contract is the chain code in the blockchain to automate the implementation of rules and terms to automatically settle claims. The application layer contains all kinds of application scenarios of the blockchain. In this study, we create a lightweight data block structure and propose the BAC consensus algorithm. We will discuss them in detail below.

## 4 | DATA LAYER—BLOCK STRUCTURE IMPROVEMENT

Every transaction (sensors data and trading data) with its timestamp will be packaged into the block. The blockchain is a linked list of these ordered blocks with timestamps using hash pointers. The hash pointer saves not only the address of the structure in memory but also the hash value of the structure, which is used to detect the contents of the structure. The hash value is calculated by the SHA256 function. Input an arbitrary length string $x$, the hash function will calculate the corresponding fixed-length output $H(x)$. The calculation process is irreversible, and with a slight move in the input, the output will beyond all recognition. The operations involved in the SHA256 are all logical bitwise operations.

Tamper-evident log ensures that transactions in blockchain can be traced. Tamper-evident logging of transactions enables both institutional and personal oversight of how these transactions are being used. The linked list structure of the blockchain is used to realize tamper-evident log: the hash pointer of current block is calculated by taking the hash value of all the contents of the previous block (including the hash pointer of the previous block). Tampering with the contents of any block will cause a domino effect, resulting in a change in the hash value saved in the latest block. So as long as we save the hash value of the latest block, the content of any block can be detected whether it has been tampered with or not. Making use of tamper-evident log, only the MO and CO nodes cluster $M = \{\mathrm{MO}, \mathrm{CO}\}$ in the MCB need to store the intact blockchain information. Other CM nodes cluster $C = \{\mathrm{CM}\}$ only need to save the latest one thousand blocks, thus greatly reducing the storage pressure of the nodes in the mining industry. Both underground and overground blocks are more optimized than traditional blocks. There is no need for Merkle root in underground block headers. The effect of the Merkle tree in Bitcoin is to verify the integrity and security of the transaction data in the block body. The data in

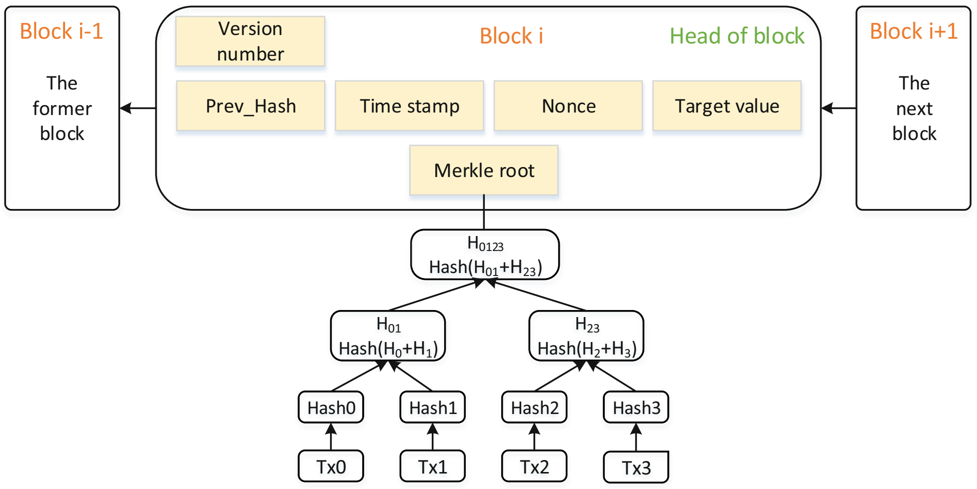


**FIGURE 4** The traditional block structure

underground block body do not contain transaction information. Our purpose is to detect whether the sensor data has been tampered with. The average size of an underground block is about 0.5 MB since the transmission of the underground data has the characteristic of high real-time. The average size of an overground block is about 1 MB.

The traditional block structure is presented in Figure 4. The traditional block structure contains the block header, which stores (i) the hash pointer to the previous block, (ii) the timestamp, (iii) the current target, it is a 256 bit integer encoded in compressed 32-bit form, (iv) the nonce and (v) the Merkle root and the block body, which stores all the transaction information. The blockchain system will set a target value for all miners to maintain the rate of producing a block so that the POW algorithm can run stably and smoothly:

$$\text{Hash} = \text{Hash}(\text{header}). \tag{1}$$

$$\text{Verify}(\text{Hash} < \mathit{Target}). \tag{2}$$

Let the symbol + denote concatenation of strings. The cryptographic problem that a miner $M$ has to solve is: compute a double SHA-256 hash

$$s = \text{SHA256}(\text{SHA256}(n + h + s' + x)), \tag{3}$$

such that $s < x$, where $x$ is the Target, $n$ is a random nonce value, $h$ is the Merkle root of the transactions stored in the block body and $s'$ is the hash pointer to the head of the blockchain at miner $M$. If $s \geq x$ then $n$ is updated and $s$ is recomputed until a solution $s < x$ is found.

The total difficulty of the blockchain is the sum of difficulties of all branches of blockchain, as shown in the following equation:

$$D(l) = D\left(\sum_{i=1}^{n} l_i\right) = \sum_{i=1}^{n} D(l_i), \tag{4}$$

$n$ is now the number of branches in the blockchain. The process of consensus occasionally causes the bifurcation of the blockchain. The blockchain system follows the principle of "longest chain," the highest blockchain is the one with the greatest total difficulty, as shown in the following equation:

$$\Phi(l_1, l_2, \ldots, l_n) = l_m, m = \max_{i \in \{1,2,\ldots,n\}} (D(l_i)). \tag{5}$$

The POW consensus mechanism is adopted in the traditional blockchain. The nonce and target value in the block are two essential parts of the POW consensus algorithm. In this study, we create a new data block structure to facilitate the BAC algorithm we proposed. The new data block structure is shown in Figures 5 and 6. The blocks in BAC consist of both block headers and block bodies. Since there is no difference between the BAC's block body and the block body in the traditional blockchain (both store specific transaction information or data information), Figures 5 and 6 mainly show the improved block headers. There is no target value and nonce in the block header. The underground data is in the form of sensor digital information rather than trading details. And coal mine safety prediction requires the high real-time performance of sensor data. Therefore, the "hash value of the previous block" is adopted instead of "Merkle root" in the underground data block.

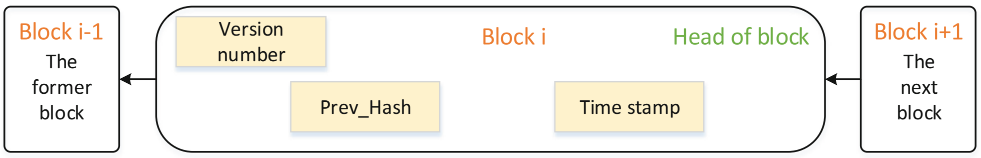


FIGURE 5 The underground block structure

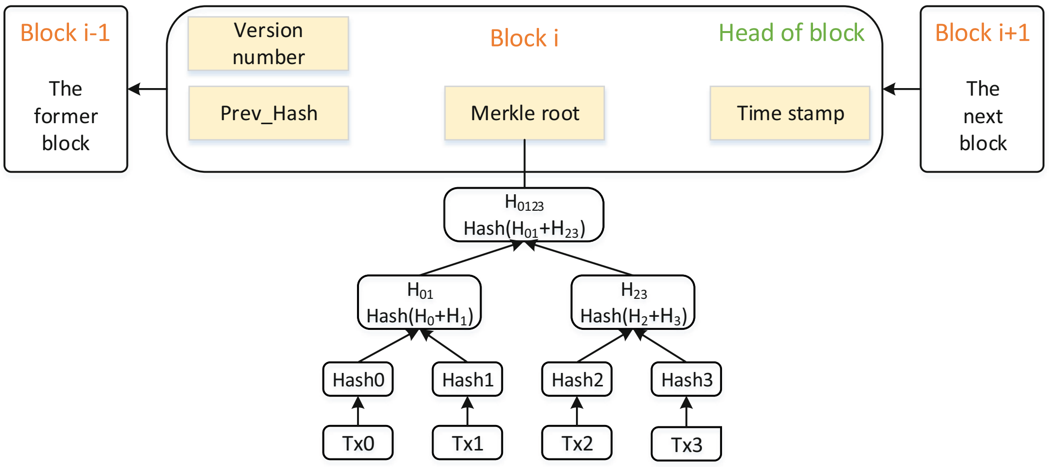


FIGURE 6 The overground block structure

The overground data block still retains the Merkle tree for its data is about transaction information such as digital assets, buyers, sellers, and production like Bitcoin. The Merkle root prevents the transaction information from being tampered with.

## 5 | CONSENSUS LAYER—BAC CONSENSUS MECHANISM

The distributed ledger of blockchain is jointly maintained by all nodes and the block content stored by each node must be consistent. So how to reach a consensus on the block content is an essential research direction in the field of blockchain technology. The traditional POW highly relies on computing power and consumes a lot of energy. PBFT is suitable for private blockchain and consortium blockchain, but the communication complexity of PBFT is critically high.

A blockchain is a deterministic finite state machine driven by transactions. The system state ($s \in S$) is a collection of stored data states. A transaction state machine is a quad $(T, S, g, \delta)$. $T$: a collection of transactions; $S$: a collection of states; $g$: the initial state; $\delta$: $S \times T \rightarrow S$, a state transition function. The ledger in blockchain is a transaction log, which is a set of transactions with abstract data types constructed recursively. The ledger is defined as follows:

$$l = \Gamma(T_t), \tag{6}$$

$$l = l_1 + l_2, \tag{7}$$

$T_t \in 2^T$ denotes a set of transactions; $\Gamma$: $T_t \rightarrow l$ is a function for constructing the blockchain ledger.

Now let $\{T_t, T_t'\} \in 2^T$, $T_t \subseteq T_t'$. If $l = \Gamma(T_t), l' = \Gamma(T_t')$ and $l \prec l'$ (where $\prec$ denotes the prefix relationship) are not satisfied, the data forking occurs. Assuming that there are $Z$ nodes in the blockchain system. $L$ is a collection of logs. $M = |L|$; $N = |S|$; $M_i = |L_i|$. $L_i = \{l | f(l) = s_i, s_i \in S\}$. $f$ is a function for constructing the system state. The probability of data forking occurrence is $P$, as shown in the following equation:[39]

$$P = \sum_{i=1}^{N} \left(\frac{M_i}{M}\right)^Z - \frac{1}{M^{Z-1}}. \tag{8}$$

As can be seen from Equation (14), there are two ways to optimize the consensus mechanism by reducing the probability of data forking. The first is to change the structure of the ledger and construct a new structure with a low forking probability. The second is to reduce the number of nodes $Z$ participating in the consensus mechanism. If the new MO diverges from a previous block of the old MO, a blockchain fork may occur. In traditional blockchains, such as Bitcoin, there is a high probability of forking because there may be two "miners" who have calculated the random value at the same time. In BAC, the designated node (leader) packages the block. Other nodes do not participate in the competition, but only vote and verify the block packaged by the leader node. We have planned to adopt the second way. But only by reducing the number of consensus nodes to achieve the superior effect of the consensus mechanism is lack of practical significance. Inspired by the blockchain sharding technology, we have divided the nodes into three categories. The number of nodes is reduced in a certain category and is a constant. We propose the BAC consensus mechanism for the MCB to ensure the accuracy of the information and the consistency of blocks. Our consensus mechanism consists of three parts: Block Consistency, $M$ Election and Credit Incentive.

## 5.1 | Block consistency

We propose the following Block Consistency algorithm for the MCB:

(I) The MO packages data from underground sensor nodes or transaction messages from overground clients(transaction initiator) into a $block(i)$ and calculates the hash value of it. The $i$ denotes the height of a block in the blockchain. The MO and CO nodes store the whole block (block header and block body) in both underground and overground scenarios. CM nodes only store the block body and block header in underground and overground respectively. In practical applications, some CM also save the information of the entire block in the overground scenario.

(II) The CO verifies the block from the MO and checks whether the data contained in the block is correct. Then the CO calculates the block hash value and compares it with the hash value of the $block(i)$. Each clock source has different clock drifts from others. So even if the time stamps of each sensor are aligned at the initial time, the result of the previous alignment will still deviate after a period of operation. The solution to this problem is to unify the clock source in the hardware. A common approach is to make a pulse generator. Each trigger will correct the clock, so that the accumulated error of clock sources can be eliminated. The time of each client may be different, the server time should be used as the same standard time. The server provides an interface for obtaining the current timestamp. The client directly gets the data from the server to obtain the current time.

(III) For underground data blocks, if the MO collects authentication messages from more than half of $O = \{\text{CO}\}$ then the MO broadcasts $block(i)$ to the other $C$ to achieve distributed storage.

(IV) For overground data blocks, a CM not only verifies the authentication messages sent by $O$, but also needs to verify the specific trading information in the block. The $block(i)$ will be formally published into the blockchain in the second round of consensus if the verification of $block(i + 1)$ is passed.

Figures 7 and 8 show the BAC-underground and BAC-overground. Both malicious and faulty nodes are considered at each stage of the BAC. Therefore, we added our security and liveness analysis to the following description of the algorithm, instead of listing it as a separate chapter. The details of the BAC consensus mechanism are as follows:

(1) The sensor node transmits data to the MO and $O$ (we take five CO as an example) while overground client broadcasts the transaction information to all nodes in the MCB. For an underground sensor capable of calculating hash value, it could directly propagate the original data together with its hash value to the MO and $O$.

(2) Block-request: MO puts the sensor data and client transaction requests to facilitate the verification of the block by $O$ and the use of other technologies in the mining industry for data prediction. Then it packages relevant information into the current $block(i)$ and sends the $block(i)$ to $O$.

(3) A CO verifies the $block(i)$. Let $block(i) = (header(i), data(i))$ denote an unvalidated block that requests to be connected at height $i$ in the blockchain. Let $block(i)$ denote an validated block at height $i$ in the blockchain. A candidate mine owner CO receives $block(i)$ from the mine owner MO. The CO validates $block(i)$ as follows. The basis of the validation is the $Prev\_Hash$ stored in $header(i)$. The CO calculates the hash $H(II)$ of $BLOCK(i - 1)$ which the CO has already received and validated. If $H(II) \neq Prev\_Hash$ the CO declines to validate $block(i)$ because the CO suspects that the MO tampered with the data in $block(i)$. The CO has already received and validated one or more copies of $BLOCK(i)$ received from other nodes in the CO set. The CO now receives $block(i)$ from the MO. In the case of an underground block the validation test is $hash(block(i)) = hash(BLOCK(i))$. In the case of an overground block the validation test compares the Merkle root in the validated $HEADER(i)$ against the recalculation of the Merkle root of the the data in the unvalidated $block(i)$. If a CO does not receive the block due to network error, it could ask for the $block(i)$ from the MO or other $O$. If $H(II) = Prev\_Hash$, the CO sends the authenticated message $\langle \text{CO-validate}, \text{CO}(j), H(block(i)), \text{CO}(s), \text{CO}(c) \rangle_{\sigma_j}$ or $\langle \text{CO-validate}, \text{CM}(k), H(block(i)), \text{CO}(s), \text{CO}(c) \rangle_{\sigma_k}$. We denote a message $m$ signed by a node

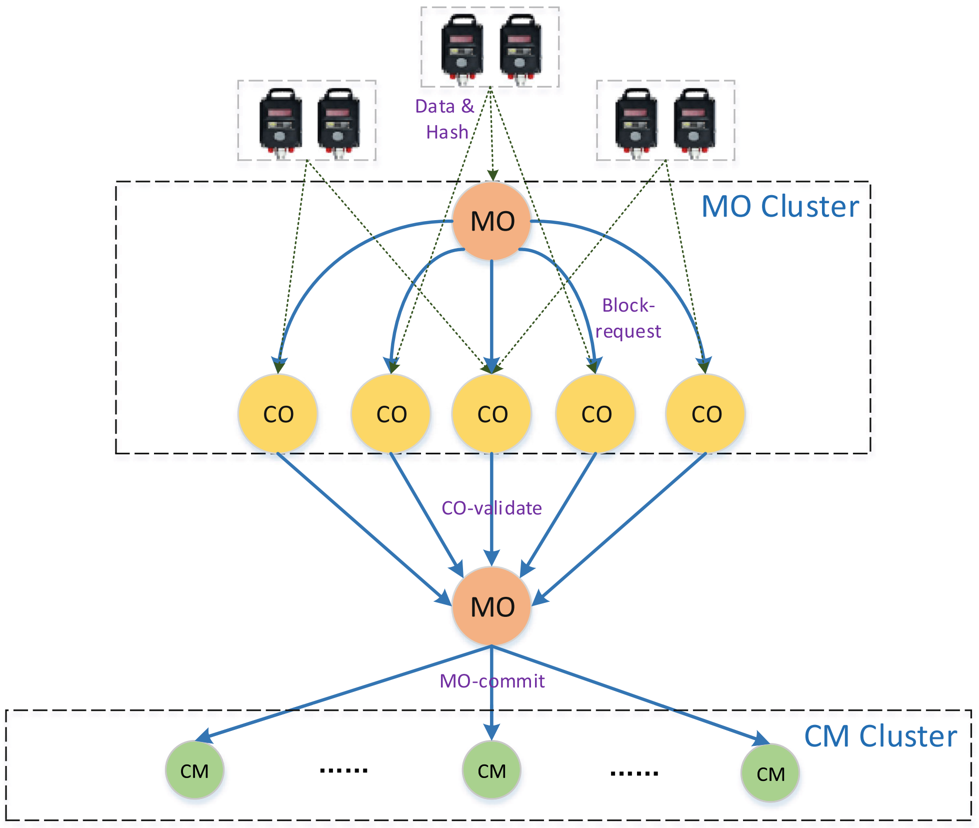


FIGURE 7 The underground-consensus

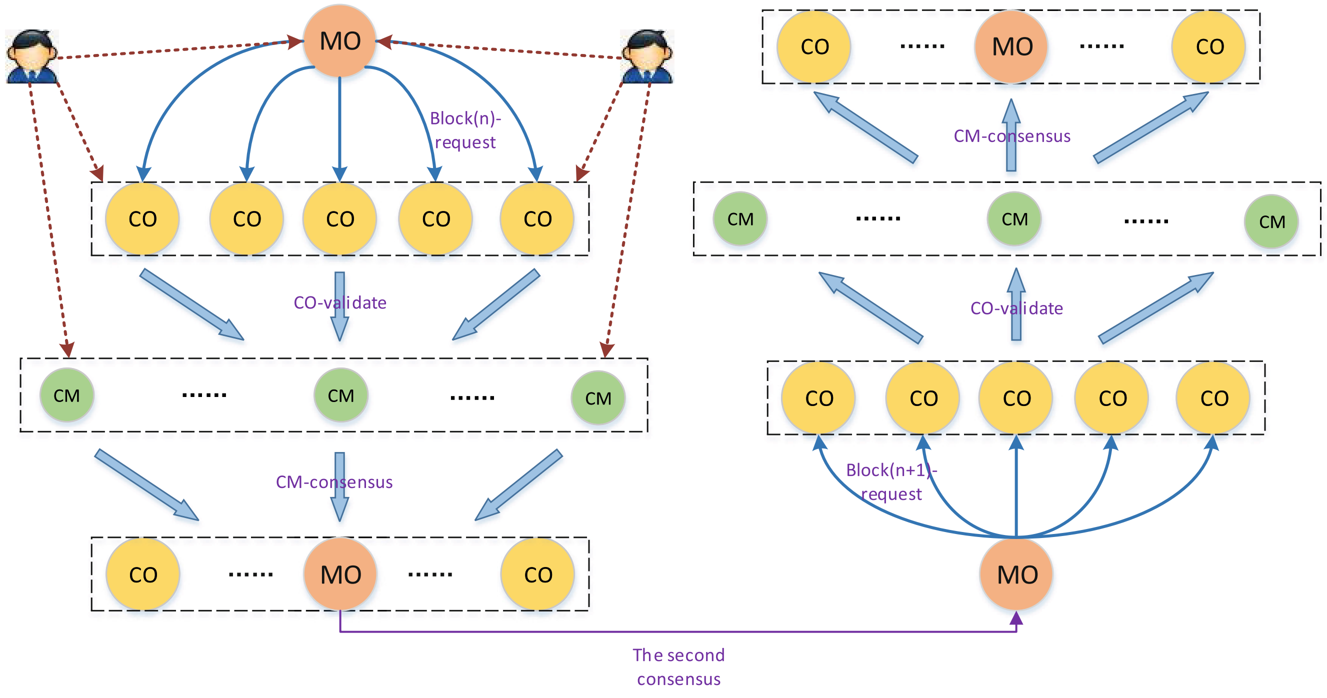


FIGURE 8 The overground-consensus

$i$ as $\langle m \rangle_{\sigma_i}$. A CM($k$) may be elected as the CO during the election phase in overground. CO($c$) indicates whether CO accepts the Block-request. The consensus of $C$ on the new *block*($i$) is divided into underground and overground scenarios.

(4) Underground-consensus: The ⟨MO-commit⟩ is the last step of the underground consensus. $O$ cannot send validate messages directly to $C$, but must return the authenticated messages to the MO, because the BAC must meet the "consistency" principle of atomic broadcasting, that is, as long as the block is legal, the block will be accepted by the whole network.[40] If a CO does not return its message to the MO, the consistency of MO and other $C$'s final state cannot be guaranteed. In addition to the "consistency", the MCB can prevent $O$ from doing evil in CO-validate. Although $O$ are reputable nodes in "Credit Incentive" and all nodes are deployed in the consortium blockchain, $O$ may respond to the MO and $C$ in the opposite way like malicious nodes in BFT.

(4.1) The MO obeys the "majority" rule. The BAC consensus mechanism is proposed based on the consortium blockchain which requires all nodes to be authenticated before taking part in the MCB, so the "majority" rule further ensures the security of the block and prevents nodes from doing evil. The "majority" rule in the MCB is as follows: If MO receives the accept messages from three or more $O$, it sends *block*($i$) and ⟨CO-validate⟩ authentication messages to $C$.

(4.2) The MCB prevents MO from tampering with the data in the second and third steps to ensure the authenticity of the sensor data before they are published into the blockchain, while the inherent characteristics of the blockchain ensures the block cannot be tampered with after it has been published to the blockchain. The MCB also has to prevent information disclosure and the crash of the entire system due to the central server being attacked. So a CM directly stores the block in its local blockchain as long as it receives sufficient CO-validate messages from MO. The ⟨CO-validate⟩ messages set from the MO has the form $\langle \text{MO-commit}, P(CO) \rangle_{\sigma_M}$. We denote a set of $m$ messages as $P(m)$. $C_1$ preserve *block*$(1)$ to *block*$(k)$, $C_2$ preserve *block*$(k+1)$ to *block*$(2k)$ $\ldots$ to achieve the effect of distributed storage. The height of the blockchain keeps growing. The $M$ save the complete blockchain. Each part of the $C$ save the most recent $k$ blocks. The specific value of $k$ needs to be determined in practical applications. It depends on the specific block size and computer configuration, etc.

(5) Overground-consensus: The $M$ (MO and $O$) keep the complete blockchain. Blocks published to the MCB-overground need double rounds of consensus by the whole network. There is no need for $O$ to propagate *block*($i$) to the MO. The clients overground send transaction information to all nodes in the bloc-kchain. In the CO-validate, the existence of the MO will weaken the characteristics of decentralization and is redundant. A CM not only collects ⟨CO-validate⟩ messages, but also needs to verify the authenticity of proposals in the block since the overground data is related to coal mine production and transactions or other essential information. Unlike $O$, there is no need to send a reject message if a CM does not approve the transaction in *block*($i$). A CM sending a $\langle \text{CM-consensus}, H(block(i)), \text{CM}(i), \text{CM}(s) \rangle_{\sigma_i}$ message means it accepts this new block.

(5.1) Block-commit: The MCB completes the first round of consensus in the Block-commit stage. The $M$ collect block authentication messages sent from $C$. A CM may act in a wrong way: it deliberately does not send the authentication message for *block*($i$) or not be able to send the authentication message when it fails. The $M$ obey the "majority" rule in the Block-commit stage: the block can be accepted by $M$ as long as $M$ receive sufficient authentication messages from more than 50% of $C$.

(5.2) Block-on-chain: The current *block*($i$) is accepted by $M$ in Step 5.1 but cannot yet be committed by the whole blockchain system. The pairwise communication between $C$ is avoided in Step 3. However, it causes only the $M$ knowing whether the *block*($i$) has been agreed on by the whole network. $C$ can not acquire the consensus result. So the new *block*($i$) can not be packaged into the blockchain authentically in the first round of consensus. In the second round of consensus, the MO sends the *block*$(i+1)$ which contains the hash value of the *block*($i$) in the block header. A CO checks whether the $H(block(i))$ in the block header of *block*$(i+1)$ is the same as the hash value of its local *block*($i$) as Step 3 described. A CM receiving sufficient CO-validate messages from $O$ proves that the *block*($i$) has been approved by the "majority" of $C$ and can be published to the blockchain. It can be seen that the new block in bitcoin needs the confirmation of six blocks before it can be published to the blockchain while the new block in the MCB only needs the confirmation of two blocks. Algorithm 1 illustrates the details of the Block Consistency. (→: sending authenticated messages; ←: receiving authenticated messages)

## 5.2 | *M* election and credit incentive

The $M$ election protocol provides liveness by allowing the MCB to make progress when the MO or CO fails. Failure or malpractice of MO or CO will result in the election of the faulty node.

### 5.2.1 | The MO election

A CO starts a timer (Δ) when it receives a block from the MO. It stops the timer when the block is under executing, but restarts the timer if at that point it is waiting to validate a new block. Each CO node has a timeout (typically between 150 and 300 ms) in which it expects the heartbeat from

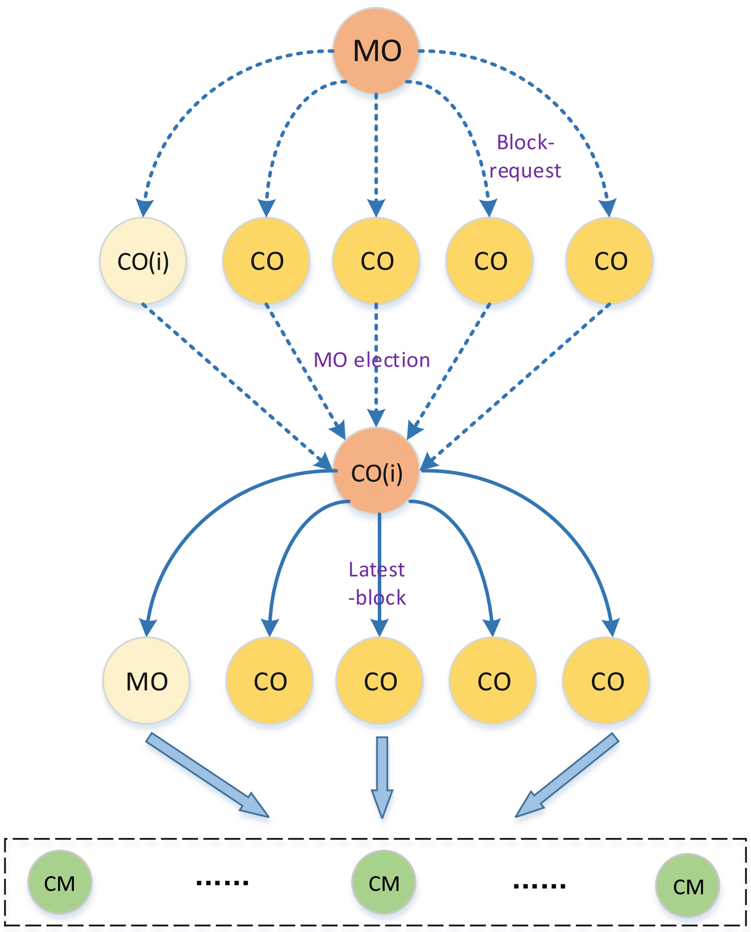


FIGURE 9 The Mine Owner election

the MO. The timeout is reset on receiving the heartbeat. If the timer of *O* expires, the CO(*j*) which gets the highest score will serve as the new MO to package the information into blocks. If there is a CO(*j*′) posing as the new MO, it will not be recognized by other *O* in the third stage of "Block Consistency." The MO may also act maliciously such as tampering with data. *O* and *C* will verify the block body and the block height. The MO has to repackage the block if it received "majority" rejection messages. Figure 9 shows the MO election model.

In the underground scenario, the node should be avoided from being elected as much as possible due to its inflexibility. CO-validate could prevent the MO from tampering with data in Block Consistency. So the election of MO is only due to the off-line (not sending messages) but not doing evil (sending tampered messages). The election of the leading node in PBFT is calculated by the system number and total number of nodes. The BAC selects the CO with the highest credit score (and the smallest number in the case of a tie) serving as the MO.

The new MO broadcasts its most recent *block*(*i*) to *O*, then a CO(*j*′′) verifies the MO's identity and its block, and continues to broadcast if the block is valid. If the most recent *block*(*i*) of the new MO is higher than the *block*(*i*′) of the CO(*j*′′), the CO(*j*′′) needs to request the block from other *O* to achieve synchronization. The CO(*n*) contributing its block will obtain additional credit scores.

In the overground scenario, the consensus process is much more complex. The data involves sensitive transactions so the MO has a greater motivation to tamper with data. In the first round of consensus, *C* will feedback the consensus result of the illegal block to *M*. So the Block-on-chain of this illegal block cannot be completed. MO's credit score will decrease if *block*(*i*) is rejected by majority of *O*. Therefore, the advantage of BAC is that it reduces the communication overhead and avoids the election of the leading node (the MO in BAC) as much as possible. Algorithm 2 illustrates the details of MO election and Credit Incentive.

### 5.2.2 | The CO election

Poor mobility and the high cost of the wired communication make it undesirable in the underground sensor network. On the other hand, wireless communication's transmission distance is too short affected by rock and coal seam. Therefore, the deployment of underground *M* is not as flexible as that of overground, so we only consider the election of *O* overground. *O* send heartbeat[41] messages to the MO.

**Algorithm 1.** Block consistency

```
Input: $T$: a set of transactions; $t$: $t \in T$; $H()$: the hash function; $M = \{MO, CO\}$; $C = \{CM\}$
Output: $out$
1 Underground: $T \to M$ or $H(t) \to M$;
2 Overground: $T \to M, C$ ;
3 while $\underline{MO\ calculates\ H(I) = H(block(i-1))_M}$ do
4     $block(i) \to O$;
5     Ocalculate $H(II) = H(block(i-1))_C$;
6     if $\underline{H(II) \neq H(I)}$ then
7         ⟨CO-validate⟩: $CO(c)_{reject} \to$ MO
8     if $\underline{H(II) == H(I)}$ then
9         ⟨CO-validate⟩: $CO(c)_{accept} \to$ MO
10 while $\underline{\langle CO\text{-}validate\rangle \to MO}$ do
11     MO obey “majority" rule;
12     if $\underline{|P(CO(c)_{accept})| > |P(CO(c)_{reject})|}$ then
13         $block(i)$, ⟨CO-validate⟩ $\to C$
14 while $\underline{C\text{-}underground}$ do
15     verify the $block(i)$;
16     if $\underline{|\langle MO\text{-}commit, P(CO)\rangle| > \frac{|O|}{2}}$ then
17         publish $block(i)$ into blockchain
18     out = success
19 while $\underline{C\text{-}overground}$ do
20     verify the $block(i)$;
21     if $\underline{P(CM) \to M}$ then
22         $M$ obey “majority" rule;
23         if $\underline{CO \leftarrow block(i+1)}$ then
24             verify $H(block(i))$ in $block(i+1)$;
25             if $\underline{|P(CO)| > \frac{|O|}{2}}$ then
26                 publish $block(i)$ into blockchain
27     out = success
28 return out
```

If the timer of MO expires (does not receive the heartbeat from CO), the MO will initiate the replacement of the faulty CO. The MO will send the election request in time because nodes in consortium blockchain are trustworthy. The MO hopes that $O$ operate normally so that it can more likely receive more than 50% of the accept authentication messages of its $block(i)$. According to Credit Incentive, the other $O$ will send $\langle$CO-election,CO($j'$),CM($k$)$\rangle_{\sigma_{j'}}$ message to MO if a faulty node CO($j'$) fails to send heartbeat information. MO selects CM($k$) from $P(\langle$CO-election$\rangle)$ and sends to $O$ and CM($k$). The CM with the number $k$ is upgraded to CO, and CO($j'$) is degraded to CM. The CM with the minimum number is selected to serve as the new CO node if two CM nodes share the highest score. In addition to downtime, the $M$ conduct a score check on $O$ every $x$ blocks specified and initiate an election of the CO whose score is below the minimum level. The specific value of $x$ is determined by the actual applications. Its estimated value is 1000. Figure 10 shows the CO election model.

The election of $M$ sacrifices the availability of the MCB to maintain consistency. All nodes continue to receive transactions from the client but there will be no new blocks generated, that is, the transaction information will be temporarily stopped from being packaged into the blockchain. So the MCB has to avoid the downtime and malicious behavior of the $M$ as much as possible. The more times a node successfully complete the consensus of blocks, the more credit scores will it get. Moreover, they will have a greater chance to serve as the MO or CO through the credit incentive, thus improving the efficiency of the MCB.

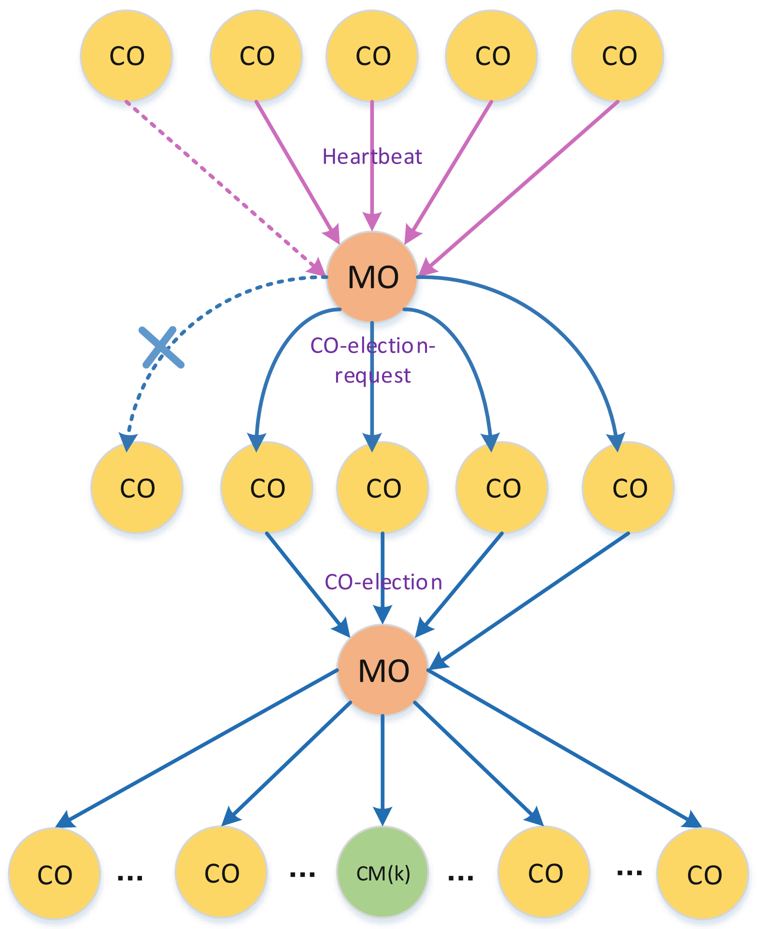


FIGURE 10 The Candidate Mine Owner election

The initial credit score of each node is 100 points. The MO will get five points if the new block is successfully published to the blockchain and be deducted 10 points if the new block which may contain illegal trading is rejected by the majority of nodes. A CO or CM node will get five points if the block authentication message sent by them is consistent with the final state of the corresponding block in the blockchain. Otherwise, no additional credit points will be added. The node contributing blocks will get five points, the upgraded node will get 10 points and the failed MO will deduct 10 points in the process of *M* election. The Credit Incentive makes our BAC mechanism perform better by rewarding nodes. We studied its effect on MCB as block rewards in the simulation part. Algorithm 2 illustrates the details of *O* election and Credit Incentive.

There are three core phases in the PBFT algorithm, namely the "pre-prepare," the "prepare" and the "commit." In the pre-prepare phase, the primary only needs to broadcast the pre-prepare message to other replicas, so the rounds of communications is $N-1$. In the prepare phase, if each replica agrees to the request, it needs to broadcast the prepare message to other replicas, so the rounds of communications is $N(N-1)$, that is, $N^2-N$. As for the commit phase, if each replica reaches the prepared state, it needs to broadcast a commit message to other replicas, so the rounds of communications is also $N^2-N$. BAC is a consensus mechanism applied to the blockchain. It is used in the verification phase of the block. In the blockchain, other nodes need to vote on the block published by the leader node, and the block has only two states of 0 and 1 (the true value is 1, the false value is 0). The block needs to be verified by nodes before it is put on the blockchain. When more than 50% of the nodes agree with the block, the block can then be formally packaged into the blockchain. There are two types of nodes that disagree with the correct block: Byzantine nodes (here by default all Byzantine nodes oppose the block) and crash nodes. If the Byzantine node is disguised, it will not affect the result of the block verification. The classic case of the PBFT consensus algorithm is the problem of kings and generals. There are three possible behaviors for a Byzantine node (send correct instructions, send wrong instructions, and don't send instructions). So the PBFT is fundamentally different from the bool type in the blockchain.

Thus the time complexity of BAC can be calculated. As shown in Figure 8, we are inspired by the blockchain sharding technique. There are $c$ CO nodes and $n$ CM nodes in the BAC, $c$ is a fixed constant value and $c \ll n$ (This raises a new problem, that is, the security and decentralization of

**Algorithm 2.** CO election

```
Input: $\Delta'$: the timer of MO; $s$: credit score
Output: CM($k$), $s$
1  if $\Delta'$ expired then
2      initiate the election of CO($j'$);
3      if $O \leftarrow$ election from MO then
4          $O \rightarrow \langle$CO-election$\rangle$ to MO;
5          MO selects CM($k$) from $P(\langle$CO-election$\rangle)$;
6          CO($j'$) = CM($k$);
7          $s$(CO($j'$)) = $s$(CO($j'$)) − 10;
8          $s$(CM($k$)) = $s$(CM($k$))+10;
9  if $x$ blocks committed then
10     check on $O$;
11     if $s$(CO($j'$)) < min $s$($O$) then
12         initiate the election of CO($j'$)
13 if MO's latest block committed then
14     $s$(MO($k$)) = $s$(MO($k$)) + 5
15 return CM($k$) and $s$
```

the network will decline. It is also the direction we are currently researching). The rounds of communication are $c$ in the Block-request stage. The rounds of communication are $cn$ in the CO-validate stage. The rounds of communication are $(c + 1)n$ in the CM-consensus stage. So the total rounds of communication are

$$T = 2[c + cn + (c + 1)n] = C_1 n + C_2, \tag{9}$$

where $C_1 = 4c + 2, C_2 = 2c$. In the same way, it can be calculated that the time complexity of a simpler BAC-underground is also $O(N)$.

## 5.3 | Safety and liveness

Safety is a fundamental property of distributed algorithms. It means that nothing bad will happen, all honest nodes will eventually deliver consistent results. We first consider the case where MO is faulty. In the underground scenario, the block published by the MO will be deemed invalid in "CO-validate" if MO maliciously tampers with data. In the overground scenario, C will feedback the consensus result of the invalid block published by MO to M in "CM-consensus." The BAC is based on the consortium blockchain, that is, most nodes behave honestly in the MCB. Therefore, we conduct a score check on $O$ every $x$ blocks specified by Credit Incentive to ensure the safety of the whole system. In addition, the "majority" principle based on the consortium blockchain guarantees the validity of blocks published by MO.

Liveness means that all good things must happen, the consensus process must be completed within a certain time, and all honest nodes must deliver consistent results. The "Block Consistency" of BAC reduces the communication overhead and avoids the election of the MO as much as possible. The main reason for the liveness of consensus algorithm is the possible transmission delay between nodes, which makes them unable to produce consistent results. We set the timeout mechanism, the $M$ Election protocol replaces faulty nodes. The elected new node broadcasts the highest block and other nodes compare the height of their own blocks against it. This ensures that consistent results are delivered within a certain time.

## 6 | PERFORMANCE EVALUATION

Code is available at https://github.com/xiaoW-a/Mine-Consortium-Blockchain-MCB.

## 6.1 | Simulation

We use the VIBES blockchain simulator to compare the performance of the BAC with that of classical blockchain algorithms. In addition to VIBES, no tool exists which can analyze the performance and security properties of a generic blockchain network.[42] VIBES differentiates itself from other works in its ability to simulate blockchain systems beyond bitcoin and its support for large-scale simulations with thousands of nodes. All computations are done on a Lenovo computer with Windows Ultimate 64-bit, Intel i7-8550 CPU @ 1.80 GHz and 8.0 GB 2133 MHz LPDDR3, Java JDK Version 11.0.10, Scala Version 2.13.5 and Akka Version 2.6.14.

VIBES uses Unsolicited Block Push (if a node knows none of its peers has a block it can directly push it to all of them) for block broadcasting. The Master Actor (the MO in MCB) builds the core of VIBES and controls the execution of events in the blockchain network. And VIBES's current nodes' classification, functions, network, and metrics are detailed enough to demonstrate three mechanisms' performance below.[43] We compare the BAC algorithm proposed in this paper with the classic POW algorithm in public blockchain and the classical PBFT algorithm in the consortium blockchian.

### 6.1.1 | The simulation of nodes and blocks

The duration of the simulation is set to 4 h. The simulation experiment of each method has been replicated 50 times under the condition of the same number of nodes. Then we calculate the average of these 50 data points of each method and compare them. We compare and analyze the "Blockchain length," "Average block time," "Total number of stale blocks" and "Last block received by" of POW, PBFT, and BAC, respectively. The confidence interval is presented in Table 1. A confidence interval displays the probability that a parameter will fall between a pair of values around the mean. Taking the indicator as the blockchain length and 80, 100, 120, 140, 160, and 180 nodes of these four mechanisms as the example. The error range of the other three indicators is similar to Table 1.

Although the overground BAC is more complete than underground and can represent the performance of the BAC, we have added the simulation of the "Blockchain length" and "Total number of stale blocks" of the underground BAC to investigate the block structure optimization we proposed above. Figure 11 shows the overall picture of nodes and blocks after the end of the simulation.

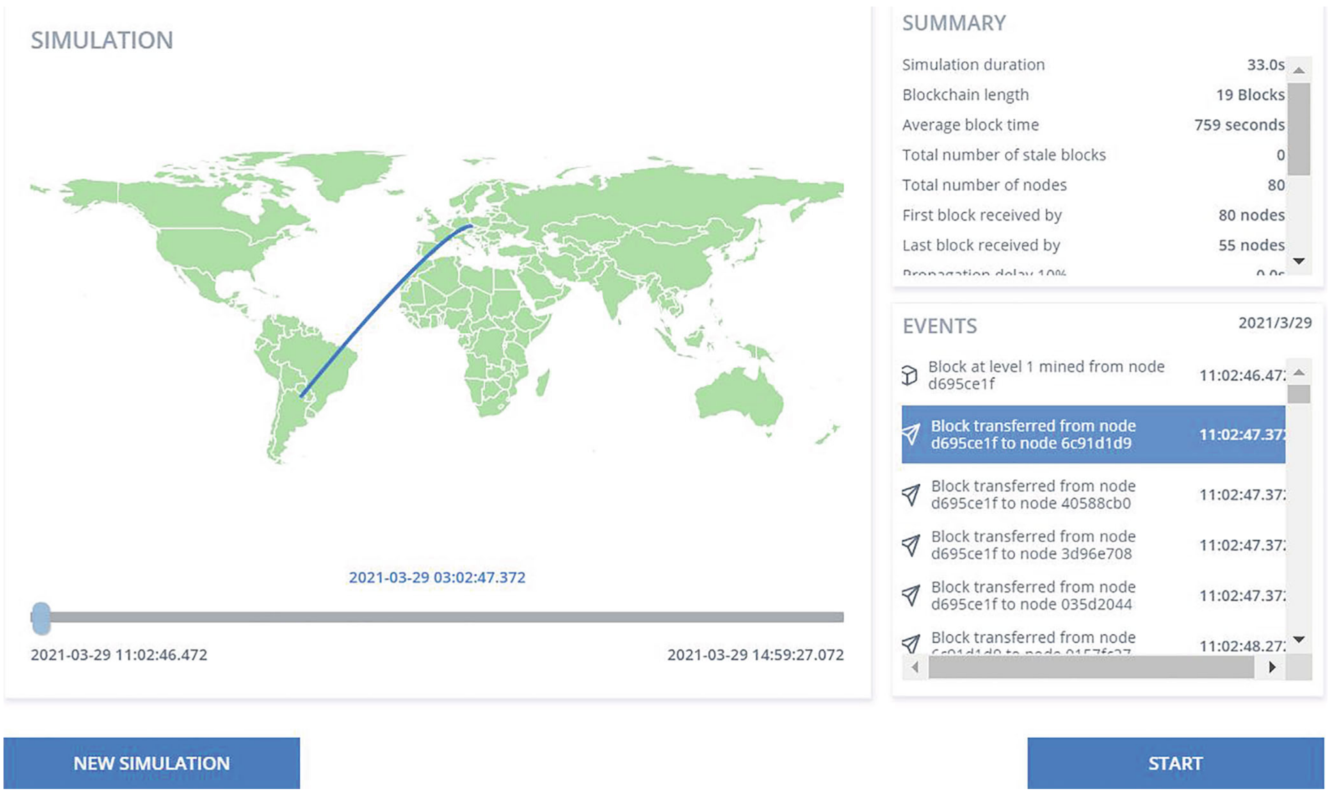
It can be seen from Figure 11 that "d695ce1f" is the MO, "6c91 d1d9," "40588cb0," "3d96e708," and "035d2044" are in *O*, and the rest (not visible in Figure 11) are in *C*. The "SUMMARY" in Figure 11 shows the four metrics about the block, and Table 2 shows the meaning of these four metrics.

(1) Figure 12 shows the performance comparison of the three algorithms. **"Blockchain length"**: The blockchain is getting longer and longer as the number of nodes increases in the POW algorithm because the duration of the POW simulation experiment is not long enough to activate the difficulty adjustment mechanism. The experimental result shows the length of the longest blockchain and blocks that are not in the main branch are viewed as stale blocks. **"Average block time"**: An increase in the number of nodes results in a decrease in block time with the same block difficulty. We realize the simulation of the POW by simulating the Bitcoin. A miner produces a block on average every 10 min in Bitcoin. To make the simulation results comparable, we adjust the block time for 100 s so that we can set parameters of other algorithms conveniently and VIBES will automatically adjust the block difficulty. The range of simulation results is relatively large, which is related to VIBES simulation time, propagation delay (900 ms, which is the system default) and other factors. **"Total number of stale blocks"**: Although producing a block in POW is extremely inefficient, the blockchain using POW is highly secure because all the nodes rely on computing power to solve the same mathematical problem. Its "Total number of stale blocks" is the highest because of the blockchain forking. The blocks that are not in the main branch are viewed as stale blocks.
**"Last block received by"**: A node propagates a block to its neighbours by the inventory block propagation mechanism in Bitcoin. The nodes with the longest blockchain can identify which blocks are needed by other nodes that are not on the main branch. They will identify the first batch

**TABLE 1** The blockchain length of different consensus mechanisms

| | Blockchain length | | | | |
|---|---|---|---|---|---|
| Mechanism | 80 (Nodes) | 100 (Nodes) | 120 (Nodes) | 140 (Nodes) | 160 (Nodes) |
| *BAC-UG* | 580 ± 0.68 | 570 ± 0.66 | 560 ± 0.72 | 550 ± 0.83 | 530 ± 0.74 |
| *BAC* | 500 ± 0.97 | 490 ± 0.76 | 480 ± 0.82 | 470 ± 0.72 | 460 ± 0.69 |
| *PBFT* | 380 ± 0.92 | 310 ± 0.96 | 290 ± 1.78 | 250 ± 0.84 | 210 ± 1.84 |
| *POW* | 30 ± 0.89 | 40 ± 0.78 | 50 ± 0.82 | 70 ± 0.93 | 100 ± 0.87 |

*Notes*: Values are presented as the confidence interval. The confidence level is 95%.



SUMMARY

| | |
|---|---|
| Simulation duration | 33.0s |
| Blockchain length | 19 Blocks |
| Average block time | 759 seconds |
| Total number of stale blocks | 0 |
| Total number of nodes | 80 |
| First block received by | 80 nodes |
| Last block received by | 55 nodes |

EVENTS 2021/3/29

| | |
|---|---|
| Block at level 1 mined from node d695ce1f | 11:02:46.47 |
| Block transferred from node d695ce1f to node 6c91d1d9 | 11:02:47.37 |
| Block transferred from node d695ce1f to node 40588cb0 | 11:02:47.37 |
| Block transferred from node d695ce1f to node 3d96e708 | 11:02:47.37 |
| Block transferred from node d695ce1f to node 035d2044 | 11:02:47.37 |
| Block transferred from node | 11:02:48.27 |

**FIGURE 11** The overall picture of nodes and blocks

**TABLE 2** The meaning of four metrics

| **Blockchain length** | **Total number of blocks in the blockchain at the end of the simulation** |
|---|---|
| Average block time | Average time to produce a block in seconds |
| Total number of stale blocks | A stale block is connected to the blockchain, but it is not in the main branch. |
| Last block received by | How many nodes successfully receive the last block, which shows the overall efficiency of the blockchain system |

of 500 blocks available for sharing, and propagate the hash values of these blocks through the **inventory** messages. Nodes lacking these blocks can request full block information through their **getdata** messages. The value of this metrics is the largest among the three since the block propagation mechanism in public blockchain highly depends on the "Neighbour Discovery Interval" and bandwidth. To make the simulation results comparable, the number of neighbour nodes in POW is set the same as that of BAC and PBFT. The block time above represents the average value of it, and the figure shows the detailed block time of a simulation.

(2) **"Blockchain length"**: The PBFT does not rely on computing power. There is a leading node packaging transactions into a block and propagating it to others. The throughput of PBFT is much higher than POW. However, its optimal time complexity is $O(N^2)$ which means its communication overhead is much larger. With the increase of the number of nodes, the delay of the block in each stage verified by the node becomes longer, and the length of the blockchain drops sharply with more than 100 nodes. **"Average block time"**: There is no block time set by the simulation in PBFT. The block time only depends on node behaviour and processing efficiency. The increases in the number of nodes lead to a gradual increase in "Average block time." The increase is relatively smooth if there are a few nodes. But the block time increases sharply with more than 100 nodes. **"Total number of stale blocks"**: The behaviour of Byzantine nodes and huge communication overhead impact the consensus of blocks seriously. In particular, this metrics becomes much larger if the primary node fails. The behavior of Byzantine nodes in PBFT is highly random. If the primary fails, the new primary may republish blocks from the start of a previous block, or link the originally packaged blocks to the old blockchain after a certain block. In this case there will be a lot of stale blocks. In our simulation experiment, when there are more than 100 nodes, the instability greatly increases, such as 120 nodes and 160 nodes. **"Last block received by"**: There are great differences among nodes to deal with transactions. The check-point mechanism in PBFT determines the transactions" watermark of a node and compresses logs. The "watermark" means that at the exact moment when PBFT reaches

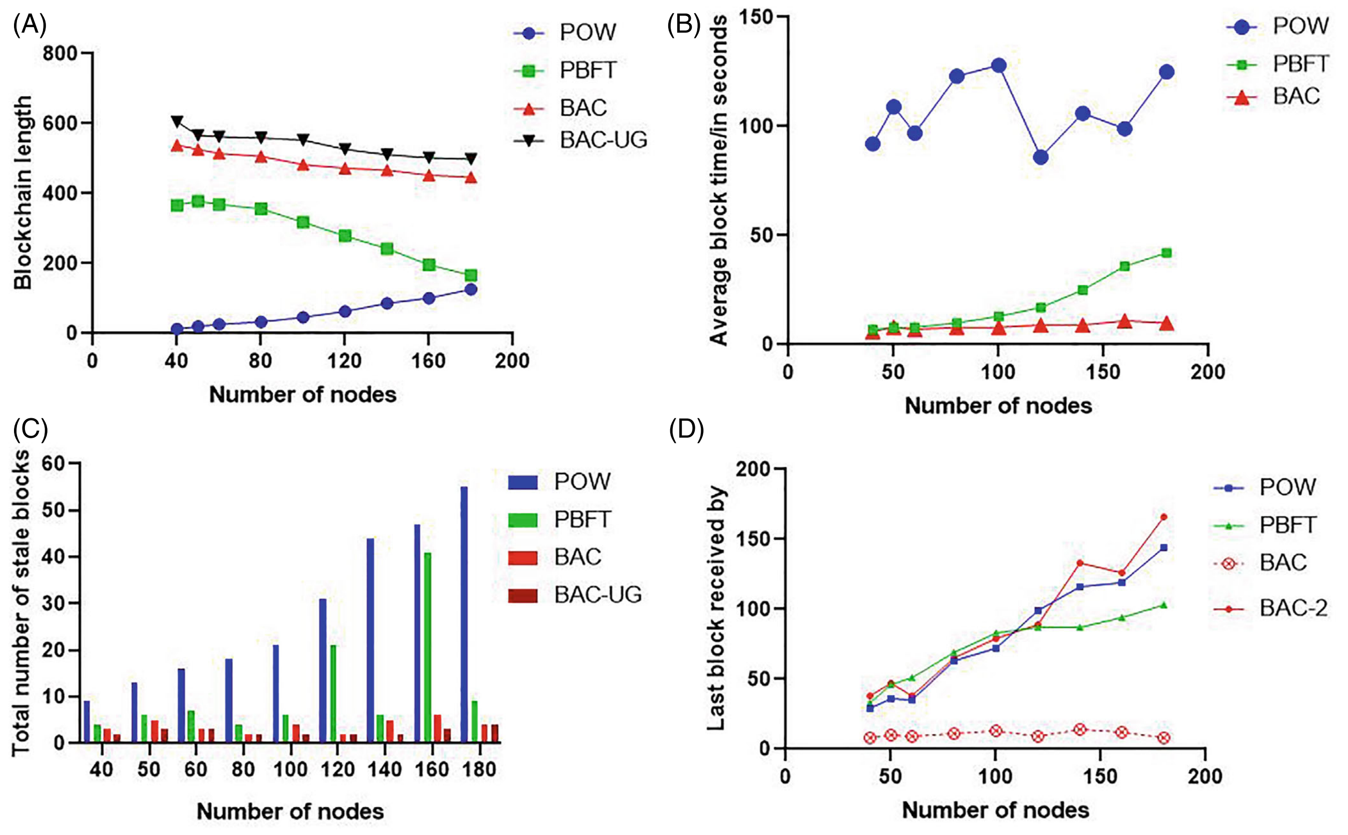


FIGURE 12 Performance of three algorithms

consensus, the block height of each block in the blockchain needs to stay within the same interval, which is controlled by the low water level $h$ and the high water level $H$. This ensures that the difference in block height between nodes is kept within a specific range when the performance of different nodes varies. The stable checkpoint is the largest request sequence number that most nodes $(2f + 1)$ have completed by consensus. Suppose the current system stable checkpoint is 1000. Then 1000 is the low water mark $h$, and the high water level $H = h + L$, where $L$ is a settable value. Its performance is similar to BAC if the number of nodes less than 100. When the number of nodes is greater than 100, the number of nodes receiving the last block is much less than POW and BAC due to the sharp performance degradation of PBFT.

(3) **"Blockchain length"**: The "blockchain length" of BAC is the highest among the three. There is also no need for computing power in BAC and its time complexity is $O(N)$. Although generating a new block needs the confirmation of at least two blocks in BAC, the communication overhead is greatly reduced compared with PBFT. **"Average block time"**: A block in BAC needs to be confirmed by two rounds of consensus. However, the "Average block time" of BAC is still less than PBFT because of its lower communication overhead and time complexity. The advantage is more obvious especially when there are more than 100 nodes. **"Total number of stale blocks"**: A cluster $O$ in BAC preverify the blocks issued by the MO. There are few stale blocks in BAC when the MO works. The existence of a small number of stale blocks in BAC is due to the transaction delay in VIBES. **"Last block received by"**: This metrics is not comparable to POW and PBFT due to the double rounds of consensus. So we replace the last block in BAC with the penultimate block, as "BAC-2" in Figure 12. The communication between $C$ in the last stage is the same as PBFT. Therefore, this metrics of BAC-2 is similar to that of PBFT when the number of nodes less than 100. BAC-UG (BAC-underground) gets higher blockchain length than BAC (BAC-overground). The block in BAC-UG does not need the second round of consensus. Therefore, the number of stale blocks of BAC-UG is less than that of BAC.

### 6.1.2 | Pending transactions

We evaluate the performance of algorithms in Section 6.1.1 by simulating blocks. In Section 6.1.2, we simulate specific transactions for further evaluation. Figure 13 shows the pending transactions of these three algorithms. Every node in the blockchain has its transaction pool size minus the number of transactions that already included in this block at the time of the block creation. The PBFT gets the maximum number of pending

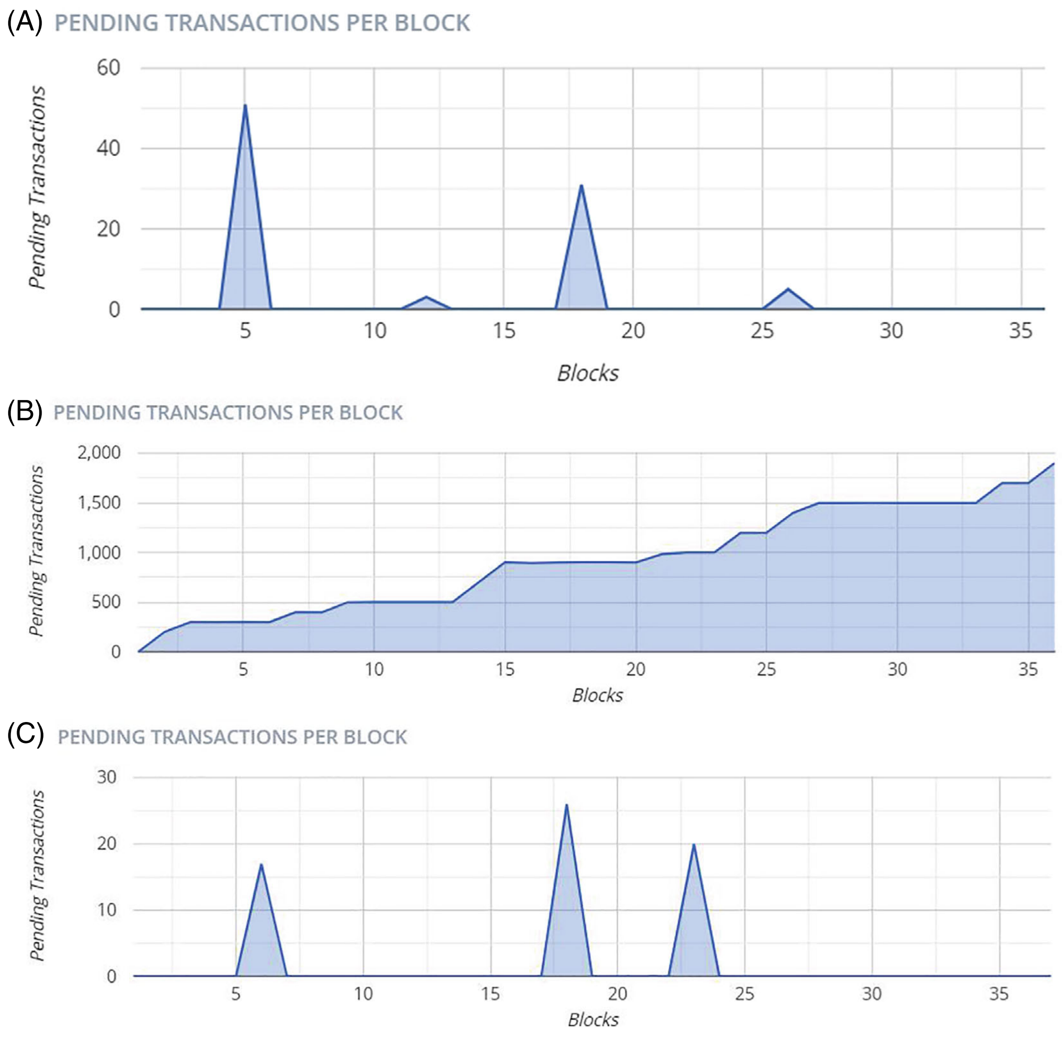


**FIGURE 13** Pending transactions of three algorithms

transactions due to its high communication complexity and poor scalability. The processed transactions have reached the upper limit when there are only a few blocks. Then we calculate the number of pending transactions of POW as 330 and BAC as 180.

### 6.1.3 | Credit incentive

Finally, we simulate the credit incentive in BAC. There are two types of incentive in Bitcoin. One is the block reward through mining. The other is the transaction fee. A user who sends a transaction has to pay a certain amount of bitcoin to the miner as a reward. The more rewards he sets, the greater chance that the transaction would be packaged into the block by the miner.

There are also two kinds of incentives in BAC. One is the credit reward given by the blockchain system according to the effectiveness of the final block state. The other is the reward of CM or *O* giving the required blocks to other nodes. As is shown in Figure 14, a node uses its credits to attach a fee to a transaction and thus persuade the leader to include the transaction in the next block to be mined.

## 6.2 | Hyperledger fabric

We have studied the superiority of BAC through simulation in Section 6.1. In this part, we deploy the MCB platform on Hyperledger Fabric and test the performance of MCB using the Hyperledger Caliper.

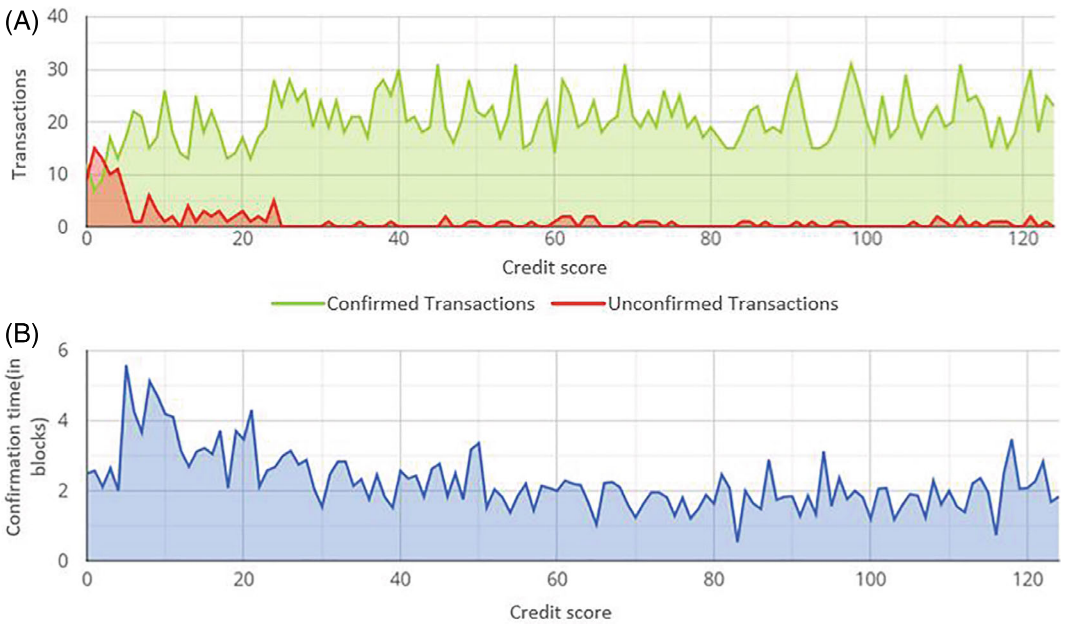


**FIGURE 14** Credit incentive

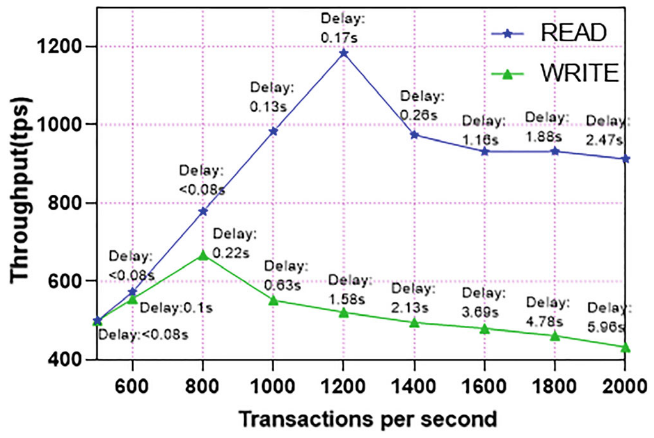


**FIGURE 15** Throughput of READ and WRITE from/to the Mine Consortium Blockchain

Bitcoin and Ethereum are two classic platforms of blockcha-in. However, they are public blockchains and as such they are not suitable for our MCB. We implement the MCB on Hyperledger Fabric. Hyperledger is the first open-source project for the consortium blockchain. It is a distributed ledger platform for enterprises. Hyperledger introduces authority management and is designed to support pluggability and scalability. The consensus mechanism in Hyperledger is pluggable, so we add BAC to the consensus module of Hyperledger. The MCB is implemented in a desktop running 64-bit Ubuntu 16.04.6 LTS with 1.6-GHz Intel Core i5 Quad-CPU and 6G RAM.

Since the proposed consortium blockchain platform is expected to serve a large number of clients accessing data simultaneously, it is necessary to evaluate the performance and scalability of the platform. Bitcoin and Ethereum can only achieve limited throughput (e.g., 7 transactions per second [tps] can be processed in Bitcoin whereas Ethereum reaches around 15 tps). The Hyperledger project launched after Bitcoin and Ethereum by the Linux foundation has demonstrated promising technology advancements of both performance and scalability. The MCB deployed on the Hyperledger Fabric is a permissioned blockchain. The additional permission control ensures that a majority of nodes are trusted, theoretically resulting in higher throughput. We use Hyperledger Caliper for our performance evaluation. Hyperledger Caliper is a blockchain performance evaluation framework that allows users to test different blockchain schemes through predefined application cases and obtain a set of performance test results such as success Rate, throughput and Latency. To integrate with our existing Hyperledger Fabric profile management system, we have programmed our adaptors using Fabric Client SDK (NodeJS version) to interact with the MCB. On top of the adaptation layer is a benchmark layer implementing predefined use-cases in the form of YAML configuration files.

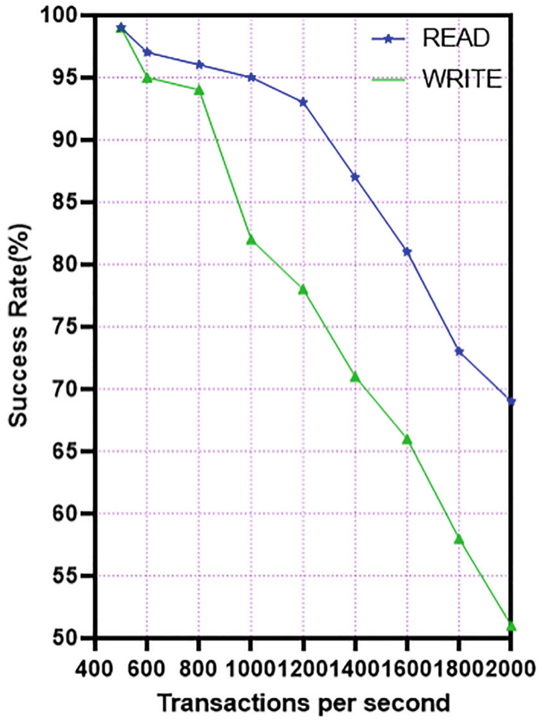


**FIGURE 16** Success rate of READ and WRITE from/to the Mine Consortium Blockchain

Figures 15 and 16 illustrates our MCB performance under different number of workloads from 500 to 2000 tps. There are 1000 clients generating proposals to our MCB system. As can be seen in the figure, the throughput of READ can reach highest to 1183 tps at 1200-tps workload whereas WRITE only reach 668 tps at 800-tps workload with 93% success rate and with less than 0.5-s latency. The ordering service is the most important part of the consensus mechanism in Hyperledger. All transactions have to be ordered through the ordering service to reach consensus. Once the transaction is written to a block, its location in the ledger can be ensured. The MO in BAC also needs to sort transaction proposals. However, the ordering service has also become the performance bottleneck of blockchain networks. WRITE transactions require more processes to chronologically order the transactions, create a new block, and broadcast it to all nodes in MCB; that is why WRITE transactions get lower throughput, lower success rate, and higher latency.

## 7 | CONCLUSION AND THE ROAD AHEAD

In this paper, we propose the MCB based on blockchain technology for the mining industry. The MCB performs efficiently by removing centralized third-party platforms. Therefore, the consensus mechanism between distributed nodes is particularly important. The BAC consensus algorithm was applied in a blockchain to improve the performance of the MCB and to allow the system to continue to work correctly when software errors and/or Byzantine nodes are present. BAC is suitable for all consortium blockchain scenarios by reducing time complexity, greatly reducing the probability of blockchain forks, and accelerating the confirmation of blocks in the blockchain. We also improve the block structure over the traditional blockchain technology. We use a dedicated blockchain simulator to obtain various aspects of the performance of blocks and nodes. The simulation results show that the proposed BAC algorithm has a much better performance on blocks and nodes. Following the guidelines from the design concept including system architecture and ledger models, a blockchain-based MCB platform is implemented on top of the Hyperledger Fabric. The feasibility and effectiveness of the MCB are successfully demonstrated.

There is still much work to do on improving our system and the BAC consensus mechanism. For future work, we will deploy our MCB in Ethereum, a public blockchain. In this regard, the MCB will not be trustworthy as some nodes might be malicious. Thus, more methods need to be implemented in the BAC to avoid the performance loss caused by the ledger forking. We consider eliminating the MO for its heavy pressure and high centralization. CM will not be trusted and we plan to implement an effective fault tolerance method in CM-consensus. Another effort is to use the smart contract to implement our credit incentive. Additionally, effective cryptography methods such as threshold encryption should be carried out to finalise the complete and secure MCB system.

## ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (No. 61872261 No. 61972274). Our deepest gratitude goes to the anonymous reviewers for their careful work and thoughtful suggestions that have helped improve this paper substantially.

## CONFLICT OF INTEREST

The authors declared that they have no conflicts of interest to this work. We declare that we do not have any commercial or associative interest that represents a conflict of interest in connection with the work submitted.

## DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author, upon reasonable request. The data that supports the findings of this study are available at https://github.com/xiaoW-a/Mine-Consortium-Blockchain-MCB. Fully documented templates are available in the elsarticle package on CTAN (http://www.ctan.org/tex-archive/macros/latex/contrib/elsarticle). And some detailed code (e.g., smart contract) are available from Yingsen Wang (e-mail: wangyingsen0318@link.tyut.edu.cn).

## ORCID

*Yan Qiang* https://orcid.org/0000-0003-3038-4255

*Juanjuan Zhao* https://orcid.org/0000-0001-8134-9076

## REFERENCES

1. Ankit S, Dheeraj K, Jürgen H. Iot based information and communication system for enhancing underground mines safety and productivity: genesis, taxonomy and open issues. *Ad Hoc Netw*. 2018;78(Sep):115-129.
2. Dheeraj K. Application of modern tools and techniques for mine safety & disaster management. *J Inst Eng*. 2016;97(1):77-85.
3. Yarkan S, Guzelgoz S, Arslan H, Murphy RR. Underground mine communications: a survey. *IEEE Commun Surv Tutor*. 2009;11(3):125-142.
4. Murphy J, Parkinson H. Underground mine communications. *Proc IEEE*. 1978;66(1):26-50.
5. Pan K, Li X. Reliability evaluation of coal mine internet of things. Proceedings of the 2014 International Conference on Identification, Information and Knowledge in the Internet of Things; 2015.
6. Nan L, Chen W. Modeling China's interprovincial coal transportation under low carbon transition. *Appl Energy*. 2018;222:267-279.
7. Xiao C, Florescu I, Zhou J. A comparison of pricing models for mineral rights: copper mine in china. *Resour Policy*. 2020;65:101546.
8. Huq N. Cyber threats to the mining industry. *Trend Microbiol*. 2016.
9. Weber RH. Internet of things – new security and privacy challenges. *Comput Law Secur Rev Int J Technol Pract*. 2010;26(1):23-30.
10. Almohri H, Watson LT, Evans D. An attack-resilient architecture for the internet of things. *IEEE Trans Inf Forens Secur*. 2020;15:3940-3954.
11. Han G, Wu J, Wang H, Guizani M, Zhang W. A multi-charger cooperative energy provision algorithm based on density clustering in the industrial internet of things. *IEEE Internet Things J*. 2019;6(5):9165-9174.
12. Bi Y, Shan H, Shen XS, Ning W, Hai Z. A multi-hop broadcast protocol for emergency message dissemination in urban vehicular ad hoc networks. *IEEE Trans Intell Transp Syst*. 2015;17(3):1-15.
13. Mistry I, Tanwar S, Tyagi S, Kumar N. Blockchain for 5g-enabled iot for industrial automation: a systematic review, solutions, and challenges. *Mech Syst Signal Process*. 2020;135(Jan.):106382.1-106382.21.
14. Gao W, Hatcher WG, Yu W. A survey of blockchain: techniques, applications, and challenges. Proceedings of the 2018 27th International Conference on Computer Communication and Networks (ICCCN); 2018:1-11.
15. Xie S, Zheng Z, Chen W, Wu J, Imran M. Blockchain for cloud exchange: a survey. *Comput Electr Eng*. 2020;81:106526.
16. Kshetri N. Can blockchain strengthen the internet of things? *IT Prof*. 2017;19(4):68-72.
17. Che Z, Wang Y, Zhao J, Qiang Y, Liu J. A distributed energy trading authentication mechanism based on a consortium blockchain. *Energies*. 2019;12(15):2878.
18. Salimitari M, Chatterjee M, Yuksel M, Pasiliao E. Profit maximization for bitcoin pool mining: a prospect theoretic approach. Proceedings of the IEEE International Conference on Collaboration & Internet Computing; 2017.
19. Zhao W, Lv J, Yao X, Zhao J, Wei C. Consortium blockchain-based microgrid market transaction research. *Energies*. 2019;12(20):3812.
20. Salimitari M, Chatterjee M, Fallah Y. A survey on consensus methods in blockchain for resource-constrained iot networks. *IoT*. 2020;11:100212.
21. Lamport L, Shostak R, Pease M. The byzantine generals problem. *Concurrency: The Works of Leslie Lamport*; 2019:203-226.
22. Novo O. Blockchain meets iot: an architecture for scalable access management in iot. *IEEE Internet Things J*. 2018;5(2):1184-1195.
23. Wu M, Wang K, Cai X, Guo S, Rong C. A comprehensive survey of blockchain: from theory to iot applications and beyond. *IEEE IoT J*. 2019;99:1.
24. Xie J, Tang H, Huang T, et al. A survey of blockchain technology applied to smart cities: research issues and challenges. *IEEE Commun Surv Tutor*. 2019;21(3):2794-2830.
25. Lin IC, Liao TC. A survey of blockchain security issues and challenges. *Int J Netw Secur*. 2017;19(5):653-659.
26. Xiao Y, Zhang N, Lou W, Hou YT. A survey of distributed consensus protocols for blockchain networks. *IEEE Commun Surv Tutor*. 2020;99:1.
27. Androulaki E, Manevich Y, Muralidharan S, Murthy C, Laventman G. Hyperledger fabric: a distributed operating system for permissioned blockchains. Proceedings of the 13th EuroSys Conference; 2018.
28. Mann S, Potdar V, Gajavilli RS, Chandan A. Blockchain technology for supply chain traceability, transparency and data provenance. Proceedings of the 2018 International Conference on Blockchain Technology and Application; 2018:22-26.
29. Evsutin O, Meshcheryakov Y. The use of the blockchain technology and digital watermarking to provide data authenticity on a mining enterprise. *Sensors*. 2020;20(12):3443.

30. Garrocho C, Klippel E, Machado A, Ferreira CM, Cavalcanti CF, Oliveira RA. Blockchain-based machine-to-machine communication in the industry 4.0 applied at the industrial mining environment. *Anais do X Simpósio Brasileiro de Engenharia de Sistemas Computacionais*. SBC; 2020:127-134.
31. Castro M, Liskov B. Practical byzantine fault tolerance. OSDI, Volucella; Vol. 99, 1999:173-186.
32. Nakamoto S. Bitcoin: a peer-to-peer electronic cash system, consulted.
33. Wood G. Ethereum: a secure decentralised generalised transaction ledger.
34. Xu X, Pautasso C, Zhu L, Gramoli V, Chen S. The blockchain as a software connector. Proceedings of the 2016 13th Working IEEE/IFIP Conference on Software Architecture (WICSA); 2016.
35. Debus J. *Consensus methods in blockchain systems, Frankfurt school of finance & management, blockchain center*. Technical report.
36. Red VA. Practical comparison of distributed ledger technologies for iot. Conference on disruptive technologies in sensors and sensor systems. SPIE Defense + Security; 2017.
37. Buterin V. A next-generation smart contract and decentralized application platform.
38. Dhillon V, Metcalf D, Hooper M. The hyperledger project. *Blockchain Enabled Applications*. Springer; 2017:139-149.
39. Wu Y, Song P, Wang F. Hybrid consensus algorithm optimization: a mathematical method based on POS and PBFT and its application in blockchain. *Math Probl Eng*. 2020;2020(11):1-13.
40. Miller A, Xia Y, Croman K, Shi E, Song D. The honey badger of BFT protocols. Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security; 2016:31-42.
41. Huang D, Ma X, Zhang S. Performance analysis of the raft consensus algorithm for private blockchains. *IEEE Trans Syst Man Cybern Syst*. 2019;50(1):172-181.
42. Stoykov L, Zhang K, Jacobsen HA. Vibes: fast blockchain simulations for large-scale peer-to-peer networks. Proceedings of the 18th ACM/IFIP/USENIX Middleware Conference: Posters and Demos; 2017:19-20.
43. Faria C, Correia M. Blocksim: blockchain simulator. Proceedings of the 2019 IEEE International Conference on Blockchain (Blockchain); 2019:439-446; IEEE.

**How to cite this article:** Wang Y, Ma Y, Qiang Y, Zhao J, Li Y, Li K. BAC: A block alliance consensus mechanism for the mine consortium blockchain. *Concurrency Computat Pract Exper*. 2022;34(27):e7344. doi: 10.1002/cpe.7344