Fu *et al. J Wireless Com Network* (2020) 2020:239
https://doi.org/10.1186/s13638-020-01855-6

EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

Data processing scheme based on blockchain

Jinhua Fu[1,2*], Mixue Xu[1], Yongzhong Huang[1,4], Xueming Si[1,3] and Chao Yuan[1]

*Correspondence: jinhua@zzuli.edu.cn
[1] State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou 450001, China
Full list of author information is available at the end of the article

Abstract
In the white paper written on Bitcoin, a chain of blocks was proposed by Satoshi Nakamoto. Since then, blockchain has been rapidly developed. Blockchain is not only limited to the field of cryptocurrency but also has been extensively applied to the Internet of Things, supply chain finance, electronic evidence storage, data sharing, and e-government fields. Both the public chain and the alliance chain have been extensively developed. In the data processing field, blockchain has a particularly good application potential. The Square Kilometre Array (SKA) is a proposal consisting of a joint venture of more than ten countries, resulting in the world's largest synthetic aperture radio telescope. In the SKA, the processing scale of the data is large, and it consists of several data processing nodes. The data will be processed in the cloud computing mode. Taking the SKA under consideration, this report proposes a data processing scheme based on blockchain for the anti-counterfeiting, anti-tampering and traceability of data. Furthermore, the authenticity and integrity of the data are assured. The primary aspects include data distribution, data operation and data sharing, which correspond to the data reception, data algorithm processing and result sharing of data operation in the SKA. With this process, the integrity, reliability and authenticity of the data are guaranteed. Additionally, smart contracts, homomorphic hashing, secure containers, aggregate signatures and one-way encrypted channels are implemented to ensure the intelligence, security and high performance of the process.

Keywords, Blockchain, Data sharing, SKA, Cloud computing, Privacy protection

1 Introduction

Blockchain is a distributed ledger technology [1]. Initially, blockchain was used primarily in the field of cryptocurrency, with Bitcoin being the most common. Litecoin [2], Monroe [3] and Zcash [4] are accepted as well. With the introduction of Ethereum in 2013, the applications of blockchain have expanded, in which the combination of smart contracts and blockchains plays an important role. However, the target of Ethereum is primarily the public blockchain. Due to the low transaction rate of Ethereum and insufficient privacy protection, successful application cases currently include issuing TOKEN and simple games, such as CryptoKitties. In Bitcoin, only seven transactions per second can be handled. Although the performance of Ethereum is better than that of Bitcoin, it can only handle 15–20 transactions per second; thus, it is unable to meet practical demand. Additionally, certain practical applications have higher requirements for privacy protection, which Ethereum cannot currently meet. In 2015, the Hyperledger project was launched, in which the IBM-backed Fabric framework was the most recognized. Fabric is aimed at the alliance blockchain, which essentially meets the needs of practical applications in terms of performance, privacy protection and usability.

With the development of the public and alliance chains, the blockchain application field has rapidly developed. Blockchain has been extensively applied to the Internet of Things [5], supply chain finance, digital data storage certificate, data processing and e-government [6] fields. In the field of data processing, blockchain guarantees the authenticity, security and reliability of data [7]. Various studies have introduced the use of blockchain for medical data sharing [8], personal data protection [9] and data distribution [10].

Astronomical data have certain characteristics, such as large amounts of data, real-time requirements [11], complicated calculation processes [12], heterogeneous calculation nodes [13], diverse storage models, various data access patterns [14], high expansibility, etc. High-performance computing, distributed computing, parallel computing, uniform resource management, container technology and telescope observation control system technology are needed [15]. Current-related technologies, such as Apache Hadoop, OpenMP, MPI, etc., all face various problems in processing astronomical data [16]. In the SKA data process, it is necessary to use cloud computing [17]. In distributed data processing, attention should be given to data protection [18]. Therefore, the security of the data-distributed storage [19] and the integrity of the data [20] are particularly important. During data processing, there are extremely high requirements for the synchronization of time [21] and the optimization of algorithm in data merging [22]. Blockchain can play a positive role in ensuring the integrity, security and availability of the data.

In the remainder of this report, Sect. 2 primarily introduces the data distribution scheme based on blockchain, which reflects the generation and collection of data in the SKA. Section 3 introduces the method of data operation, which reflects the combination of collected data and related algorithms. Section 3.3 introduces the process of sharing data, which reflects the sharing problem of the result after the original data is processed by the related algorithms. Section 4 summarizes the conclusions of this report.

2 Preliminaries

In this section, we first define certain notations used in this report. If S is a set, then $|S|$ denotes the number of elements in this set. If b is a real number, then $a \leftarrow b$ indicates that $a = b$. If C is a node and c is an element, then $C \Leftarrow c$ denotes sending c to C. If a and b are two real numbers, then $a||b$ indicates the cascading of a and b.

2.1 Bilinear mapping

$\mathbb{G}_1$ and $\mathbb{G}_2$ are two multiplicative cyclic groups of prime order p, where g_1 is a generator of $\mathbb{G}_1$ and g_2 is a generator of $\mathbb{G}_2$. ψ is a computable isomorphism from $\mathbb{G}_2$ to $\mathbb{G}_1$, with $\psi(g_2) = g_1$. A bilinear pairing can be defined as $\mathcal{G} = (n, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ where $\mathbb{G}_1 = \langle g_1 \rangle$, $\mathbb{G}_2 = \langle g_2 \rangle$ and $\mathbb{G}_T$ are multiplicative groups of order n. Let $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ be defined as a map with the following properties:

- *Bilinear*: $\forall u \in \mathbb{G}_1, v \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_n : e\left(u^a, v^b\right) = e(u, v)^{ab}$.
- *Non-degenerate*: There exists $u \in \mathbb{G}_1$, $v \in \mathbb{G}_2$ such that $e(u, v) \neq \mathcal{O}$, where $\mathcal{O}$ denotes the identity of $\mathbb{G}_T$.
- *Computability*: There is an efficient algorithm to compute $e(u,v)$ for all $u \in \mathbb{G}_1$, $v \in \mathbb{G}_2$.

Then, e is considered bilinear mapping.

2.2 Aggregate signature

An aggregate signature is a variant signature scheme used to aggregate any number of signatures into one signature. For example, suppose there are n users in the system $\{u_1,u_2,\ldots,u_n\}$, n public keys $\{pk_1, pk_2,\ldots,pk_n\}$, n messages $\{m_1, m_2,\ldots,m_n\}$ and n signatures $\{\sigma_1, \sigma_2,\ldots,\sigma_n\}$ for these messages. The generator of the aggregate signature (here the generator can be arbitrary and does not need to be in $\{u_1,u_2,\ldots,u_n\}$) can aggregate $\{\sigma_1, \sigma_2,\ldots, \sigma_n\}$ to a short signature σ. Importantly, the aggregate signature is verifiable, i.e., given a set of public keys $\{pk_1, pk_2,\ldots,pk_n\}$ and its signatures of the original message set $\{m_1,m_2,\ldots,m_n\}$, it can be verified that the user u_i has created a signature of message m_i. The execution of the aggregate signature is described in detail below.

AS = (Gen, Sign, Verify, AggS, AggV) is a quintuple of the polynomial time algorithm, and the details can be noted as follows:

DS = (Gen, Sign, Verify) is a common signature scheme, which is also known as the benchmark for the aggregate signature.

Aggregation signatures generation (AggS). Based on Gen and Sign, the common signature function and the aggregation of $\{m_1, m_2,\ldots, m_n\}$, $\{u_1, u_2,\ldots, u_n\}$ and $\{\sigma_1, \sigma_2,\ldots, \sigma_n\}$ can be realized, thus aggregating a new signature σ_n.

Aggregation signature verification (AggV) Suppose that each u_i corresponds to a public–private key pair $\{pk_i, sk_i\}$. If AggV($pk_1,\ldots,pk_n$, $m_1,\ldots,m_n$, AggS($pk_1,\ldots, pk_n$, $m_1,\ldots,m_n$, Sign(sk_1,m_1),…,Sign(sk_n,m_n))) = 1, then the output is 1; otherwise, the output is 0.

Furthermore, the aggregate signature can support incremental aggregation; thus, if σ_1 and σ_2 can be aggregated to σ_{12}, then σ_{12} and σ_3 can be aggregated to σ_{123}.

2.3 Homomorphic hash

Homomorphism is the mapping of two algebraic structures in abstract algebra that remain structurally constant. There are two groups, $\mathbb{G}_1$ and $\mathbb{G}_2$, and f is the mapping from $\mathbb{G}_2$ to $\mathbb{G}_1$. If $\forall a, b \in \mathbb{G}_1$, $f(ab) = f(a)f(b)$, then f is called a homomorphism from $\mathbb{G}_2$ to $\mathbb{G}_1$.

The homomorphic hash has long been used in peer-to-peer networks [23], which use correction and network codes together against attack events. In a peer-to-peer network, each peer will obtain the original data block directly from the other peers; thus, hash functions such as SHA1 can be used to directly verify the correctness of the received data block by comparing the hash value of the received data block with the original hash value.

Using the homomorphic hash function mentioned in earlier studies [24], i.e., $h_{\mathbb{G}}(\cdot)$, a set of hash parameters can be obtained as $h_{\mathbb{G}}(\cdot), \mathbb{G} = \left(p, q, g\right)$. The parameter description is shown in Table 1. Each of these elements in g can be represented as $x^{(p-1)/q} \bmod p$, where $x \in \mathbb{Z}_p$ and $x \neq 1$.

Table 1 Parameter description

Parameter	Description
λ_p	Discrete logarithmic secure parameter (1024 bit)
λ_q	Discrete logarithmic secure parameter (257 bit)
p	Random prime number $\lvert p\rvert = \lambda_p$
q	Random prime number $q \mid p-1, \lvert q\rvert = \lambda_q$
β	Data block size (16 kB)
m	Number of subblocks contained in each block $m = \left\lceil \frac{\beta}{\lambda_q - 1} \right\rceil (512bit)$
g	Generate the 1 $\times$ m row vector, randomly selected from the $\mathbb{Z}_p$
$\mathbb{G}$	Hash parameters, including $\mathbb{G} = (p, q, g)$
n	Block number
Seed	Seed of keystream generator
MAXRAND	Maximum possible output of $rand(\cdot)$
MINRAND	Minimum possible output of $rand(\cdot)$

$$h_{\mathbb{G}}(\cdot) : \{0,1\}^k \times \{0,1\}^\beta \rightarrow \{0,1\}^{\lambda_p} \tag{1}$$

$$rand(\cdot) : \{0,1\}^k \times \{0,1\}^t \rightarrow \{0,1\}^t \tag{2}$$

where $rand(\cdot)$ is a pseudo-random function, which can be used as a pseudo-random number generator to initialize the homomorphism hash function parameters in the generating process, generate random numbers in the tag generate process, and determine the random data block in the challenge process, thus creating challenges that can cover the entire data range.

For a block b_i, the hash value can be calculated as follows:

$$h(b_i) = \prod_{k=i}^{m} g_k^{b_{k,i}} \bmod p \tag{3}$$

The hash values of the original block $(b_1, b_2, \ldots, b_n)$ are $h(b_1), h(b_2), \ldots, h(b_n)$.

Given a coding block e_j and a coefficient vector $(c_{j,1}, c_{j,2}, \ldots, c_{j,n})$, the homomorphic hash function $h_{\mathbb{G}}(\cdot)$ can satisfy the equation as follows:

$$h(e_i) = \prod\nolimits_{i=1}^{n} h^{c_{j,i}}(b_i) \tag{4}$$

This feature can be used to verify the integrity of a code block. First, the publisher needs to calculate the homomorphic hash values of each data block in advance. The download downloads these homomorphic hash values. Once the verification block is received, its hash value can be calculated using Eq. (3). Then, Eq. (4) can be used to verify the correctness of the verification block [25].

3 Results and discussion

3.1 Blockchain-based data distribution scheme

Here, we simplify the process of receiving astronomical data in the SKA. The *SOURCE* represents the original astronomical data, and the *Data Receiving Station (DRS)* represents the real astronomical data receiving device. The DRS setting is distributed. Different DRSs are responsible for receiving data within their own respective areas. Considering the limitation of the hardware functions, the DRS is only responsible for data reception, temporary storage and data forwarding; it does not participate in data calculation. All data calculation is completed by the *Data Processing Node (DPN)*, which is connected to the blockchain. The concrete architecture is shown in Fig. 1.

The method of processing data from the *SOURCE* to the *DRS* is relatively simple. It involves processing the data format and setting the storage mode, which is not the focus of this study. Here, the execution process of the *DRS* to the *DPN* is introduced.

Furthermore, we use the idea of distributed storage in an IPFS, as shown in Fig. 2.

Each block contains a list of trading objects, a link to the previous block, and a hash value for the state tree/database.

Additionally, we introduce the method used to import data into a blockchain. Let q be a large prime number. Then, select $P \in \mathbb{G}_1, Q \in \mathbb{G}_2$ to define an additive group $\mathbb{G}_1$ and a multiplicative group $\mathbb{G}_2$ with order q. Thus, a bilinear mapping $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ and hash functions $H : \{0,1\}^* \to \{0,1\}^*$, $H_0 : \{0,1\}^* \to \mathbb{Z}_q^*$, $H_1 : \{0,1\}^* \times \mathbb{G}_1 \to \mathbb{G}_2$, $H_2 : \{0,1\}^* \to \mathbb{G}_1$, $H_{DV} : \{0,1\}^* \to \mathbb{G}_1$ can be obtained. The number of data receiving stations is m, the number of data processing nodes responsible for the ith data receiving station is m_i, and the current view is v.

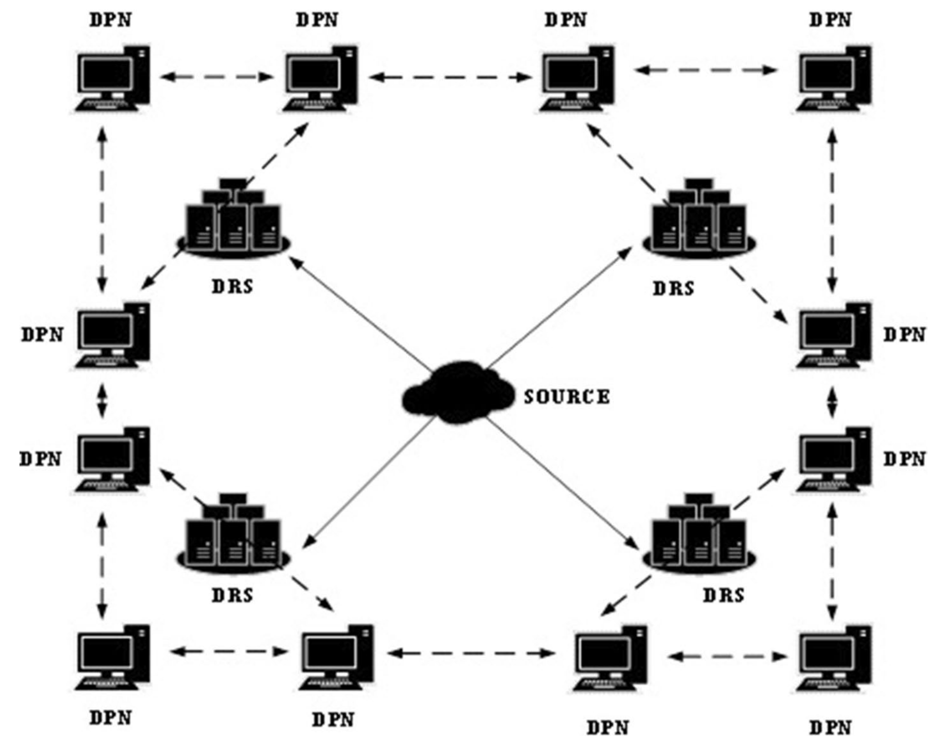

Fig. 1 Data distribution based on blockchain. This image depicts the concrete architecture of the process of receiving astronomical data in a SKA. The SOURCE represents the original astronomical data, and the data receiving station (DRS) represents the real astronomical data receiving device. All data calculations are completed by the data processing node (DPN), which is connected to the blockchain

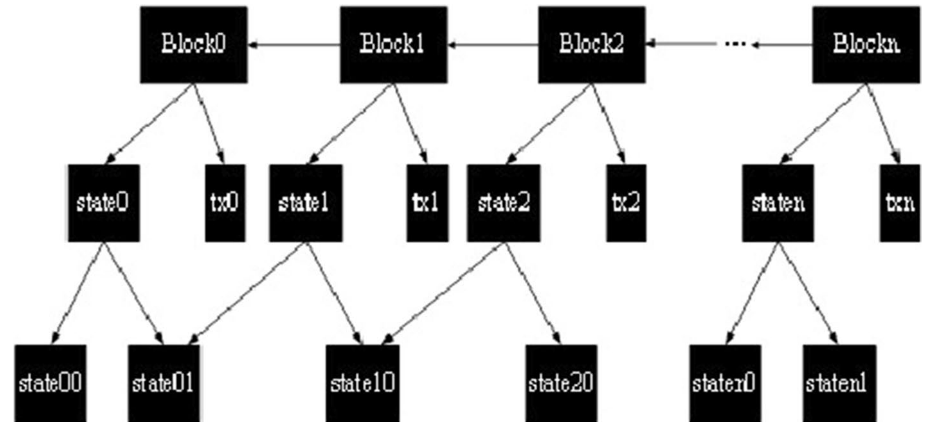

Fig. 2 Distributed storage in IPFS. This image depicts the distributed storage in an IPFS. Each block contains a list of trading objects, link to the previous block, and hash value for the state tree/database

1 Using the current view, calculate $P_v = v \cdot P$. Combined with the existing parameters, the system parameters can be obtained as follows: $Params = \{G_1, G_2, e, q, P, Q, P_v, H_0, H_1, H_2, H_{DV}\}$.
2 The user u_i selects a random value $x_i \in \mathbb{Z}_q^*$ as its secret value and calculates $P_i = x_i \cdot P$, $Q_i = H_1(ID_i||P_i)$, and $D_i = v \cdot Q_i$ to generate the user's private key $S_i = (D_i, x_i)$.

It can be assumed that the public key of the *jth* $(j = 1, 2, \ldots m_i)$ Data Processing Node $\left(DPN_i^j\right)$ of the *ith* $(j = 1, 2, \ldots m_i)$ Data Receiving Station (DRS_i) in the *r*th round is $\{pk_i^1, pk_i^2, \ldots, pk_i^{m_i}\}$. The data produced by the SOURCE is D_i^r. Each DRS consensus for the resulting data can be reached using a static aggregate Practical Byzantine Fault Tolerance (PBFT) [26, 27]. The specific process is shown in Algorithm 1.

Algorithm 1 Method used to import data into a blockchain

```
Input: the number of DRS$m$, round $r$, public keys set $\{v_i^1, v_i^2, \cdots, v_i^{i_n}\}$.
Output: blocks $B_r$
1: $S_r \leftarrow DRS_{r \bmod m}$
2: for $i = 1$ to $m$
3:     $M_i^r \leftarrow DPN_i^{(r+i) \bmod m_i}$
4:     $M_i^r \Leftarrow \left(D_i^r, H\left(D_i^r\right)\right)$
5:     $M_i^r$ selects $r_i^r \in \mathbb{Z}_q^*$ and computes $R_i^r = r_i^r \cdot P$, $h_i^r = H_0\left(ID_i^r \| D_i^r \| P_i^r \| R_i^r\right)$, $T = H_2\left(P_v\right)$
6:     $M_i^r$ computes $V_i^r = D_i^r + h_i^r \cdot r_i^r \cdot T + x_i^r \cdot Q$ and outputs $\sigma_i^r = \left(V_i^r, R_i^r\right)$
7:     $S_r \Leftarrow \left(D_i^r, \sigma_i^r\right)$
8:     $S_r$ computes $V = \sum_{i=1}^{m} V_i, R = \sum_{i=1}^{m} h_i R_i$, $\sigma = (V, R)$, $D \leftarrow D_1^r \| D_2^r \| \cdots \| D_m^r$
9: end for
10: for $i$ =1 to $m$ do
11: $M_i^r \Leftarrow (D, \sigma)$
12:    for $j$=1 to $m_i$ do
13:        $DPN_i^j \Leftarrow (D, \sigma)$
14: end for
15: end for
```

To verify the validity of the aggregate signature σ, Algorithm 1 can be implemented. Using the system parameter *Params*, user's corresponding identity list $ID = \{ID_1, \ldots, ID_n\}$, public keys list $P = \{P_1, \ldots, P_n\}$, messages list $M = \{m_1, \ldots, m_n\}$, signature list $\sigma = \{\sigma_1, \ldots, \sigma_n\}$, computer $Q_i = H_1(ID_i||P_i)$ and $T = H_2(P_v)$, the equation can be verified as follows:

$$e(V,P) = e\left(\sum_{i=1}^{n} Q_i, P_v\right) e(T,R) e\left(Q, \sum_{i=1}^{n} P_i\right) \tag{5}$$

If the equation holds true, then the validation passes; otherwise, the validation fails.

The correctness of this basic framework is given below. Theorems 1 and 2 provide the correctness of the verification process of a single signature and the correctness of the verification process of an aggregate signature, respectively.

Theorem 1 The verification process of a single signature is correct.

Proof: The verification process of the signature $\sigma_i = (V_i, R_i)$ that DRS_i performs for D_i^r can be given as follows:

$$\begin{aligned} e(V_i,P) =& e(D_i + h_i r_i T + x_i Q, P) \\ =& e(D_i,P) e(T, h_i r_i P) e(x_i Q, P) \\ =& e(Q_i,P_v) e(T, h_i R_i) e(Q, P_i) \end{aligned} \tag{6}$$

Theorem 2 The verification process of an aggregation signature is correct.

Proof:

$$\begin{aligned} e(V,P) &= e\left(\sum_{i=1}^{n} V_i, P\right) = e\left(\sum_{i=1}^{n} D_i + x_i Q + h_i r_i T, P\right) \\ &= e\left(\sum_{i=1}^{n} D_i, P\right) e\left(\sum_{i=1}^{n} h_i r_i T, P\right) e\left(\sum_{i=1}^{n} x_i Q, P\right) \\ &= e\left(\sum_{i=1}^{n} Q_i, P_v\right) \prod_{i=1}^{n} e(T, h_i R_i) \prod_{i=1}^{n} e(Q, P_i) \\ &= e\left(\sum_{i=1}^{n} Q_i, P_v\right) e(T,R) e\left(Q, \sum_{i=1}^{n} P_i\right) \end{aligned} \tag{7}$$

3.2 Blockchain-based data operation scheme

The Science Data Processor (SDP) [28] is the SKA Data processing module. The main data are taken from the Central Signal Processor (CSP) module [29], the metadata are taken from the Telescope Manager (TM) module, and the Signal and Data Transport (SaDT) module is responsible for the data transmission. Multiple regional data processing centres will be built. The primary functions of the SDP can be given as follows:

- Extract data from the CSP and TM modules

- Treat source data as data products that can be used for scientific research
- Archive and store data products
- Provide access to data products
- Control and feedback information to the TM module for a timely challenge observation

In the SKA SDP, the two most important computational tasks are FFT [30] and gridding [31]. These two algorithms account for an important part of the total computation, and their efficient implementation provides considerable assistance in the design of the SKA SDP.

As depicted in Fig. 3, in the data calculation scheme based on the blockchain, the Data Supply Node (DSN) and the Algorithm Supply Node (ASN) are separated, and all of the data and algorithms enter the Secure Container [32] through a one-way encrypted channel under the control of the Smart Contract (SM) to perform calculations. Providers and the provided time of the data and algorithms are recorded on the blockchain through the SM. It can be assumed that there are w Data Supply Nodes and one Algorithm Supply Node. Before entering the Secure Container, all of the data D_i $(i = 1, 2, \dots, w)$ and algorithms A are signed by the private key(sk_i) of the DSN$_i$ and the private key(sk_a) of the ASN. Furthermore, the data and algorithms are first encrypted by the public key SC_{pk} of the Secure Container and then decrypted and verified after entering the Secure Container by the public key (pk_i) of the DSN$_i$, the public key(pk_a) of the ASN and the private key of the Secure Container. This specific process is shown in Algorithm 2 and Algorithm 3.

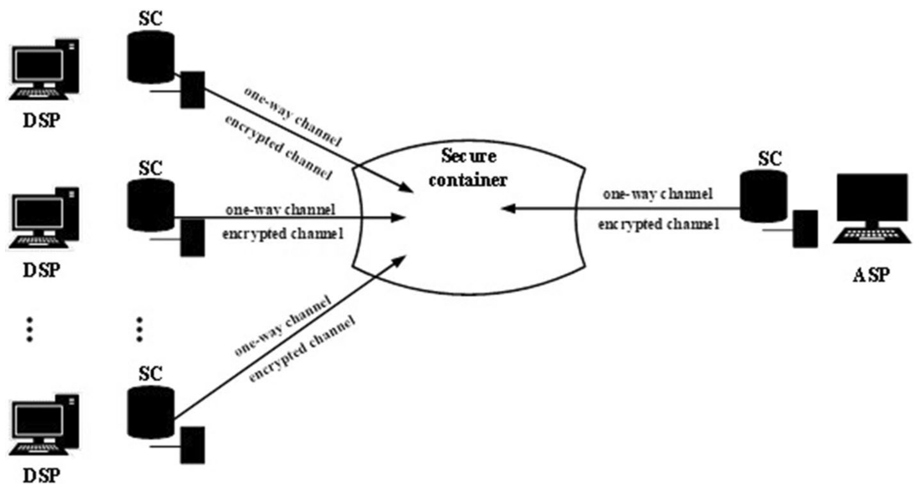

Fig. 3 Data Operation Based on Blockchain. In the blockchain data calculation scheme, the Data Supply Node (DSN) and the Algorithm Supply Node (ASN) are separated, and all of the data and algorithms enter the Secure Container through a one-way encrypted channel under the control of the Smart Contract (SM) to perform calculations

Algorithm 2 Data Operation Based on Blockchain (1)

Input: hash function $H:\{0,1\}^* \to \{0,1\}^*$, public key of security container SC_{pk}, private keys set $\{sk_1, sk_2, \cdots, sk_w\}$ of DSNs, private key sk_a of ASN

Output: block b

```
1: for $i = 1$ to $w$
2: $D_i^{'} \leftarrow Enc_{SC_{pk}}\left(Sig_{sk_i}\left(D_i\right)\right)$
3: $SC \Leftarrow D_i^{'}$
4: $b_i \leftarrow H\left(D_i\right) || time_i$
5: end for
6: $A^{'} \leftarrow Enc_{SC_{pk}}\left(Sig_{sk_a}\left(A\right)\right)$
7: $SC \Leftarrow A^{'}$
8: $b_a \leftarrow H\left(A\right) || time_a$
9: $b \leftarrow b_1 || b_2 || \cdots || b_w || b_a$
```

As described in Algorithm 2, each DSN signs the data with its own private key and then encrypts the data with the public key of the security container. The processed data are sent to the security container. Then, the sub-block $H(D_i)||time_i$ is calculated. Then, the ASN signs the algorithm with its own private key and encrypts the data with the private key of the security container. The processed data are sent to the security container. The sub-block $H(A)||time_a$ is then calculated. At last, the final block $b_1||b_2||\cdots||b_w||b_a$ is calculated.

Algorithm 3 Data Operation Based on Blockchain (2)

Input: hash function $H:\{0,1\}^* \to \{0,1\}^*$, private key of security container SC_{sk}, public keys set $\{pk_1, pk_2, \cdots, pk_w\}$ of DSNs, public key pk_a of ASN

Output: block b

```
1: for $i = 1$ to w
2: $D_i \leftarrow Ver_{pk_i}\left(Dec_{SC_{sk}}\left(D_i^{'}\right)\right)$
3: $SC \Leftarrow D_i$
4: $b_i^{'} \leftarrow H\left(D_i^{'}\right) || time_i$
5: end for
6: $A \leftarrow Ver_{pk_a}\left(Dec_{SC_{sk}}\left(A^{'}\right)\right)$
7: $SC \Leftarrow A$
8: $b_a^{'} \leftarrow H\left(A^{'}\right) || time_a$
9: $b^{'} \leftarrow b_1^{'} || b_2^{'} || \cdots || b_w^{'} || b_a^{'}$
```

As described in Algorithm 3, the security container verifies each $D_i^{'}$ with its private key and the public key of each DSN. The processed data are then sent to the security container. Next, the sub-block $H\left(D_i^{'}\right)||time_i$ is calculated. Then, $A^{'}$ is verified with its private key and the public key of the ASN. The processed data are sent to the security container. The sub-block $H\left(A^{'}\right)||time_a$ is calculated. At last, the final block $b_1'||b_2'||\cdots||b_w'||b_a'$ is calculated.

3.3 Blockchain-based data sharing scheme

The Data Requirement Nodes, which are represented by the public keys $\{pk_1, pk_2, \ldots, pk_r\}$ of the calculation result, are determined in advance through the smart contract. Under intrusive surveillance, the calculated result *Re* is shared to the nodes represented by these public keys. The shared results, targets and shared time are recorded on the blockchain through the smart contracts. The concrete architecture is shown in Fig. 4.

As shown in Fig. 4, the allocation of data is allocated by the data container to each data consumer. In order to ensure the security of data, data allocation adopts the way of single channel. The data allocation rules are determined by the smart contract of the system.

Before recording on the blockchain, it is necessary to verify the target, and the target verifies the calculated results. If the verification passes, then it is signed. If more than 2/3 of the target's signature is obtained, then the block formed will be recorded on the blockchain. The simple architecture is shown in Fig. 5.

It is assumed that there are r Data Requirement Nodes (DRNs). The calculated results *Re* are encrypted by the public key pk_i of DRN_i $(i = 1, 2, \ldots, r)$ and signed by the private key SC_{sk} of the SC to obtain Re_i. The cascading of the hash value of Re_i and the time forms the block b_i. The homomorphic hash h is used by pk_i. Then, b_i $(i = 1, 2, \ldots, r)$ forms the final block b. At last, the homomorphic hash is verified. If the verification passes, then the calculation results Re_i will be sent to the DRN_i, which will be decrypted by the private key sk_i of the DRN_i and the public key SC_{pk} of the secure container. This specific process is shown in Algorithm 4.

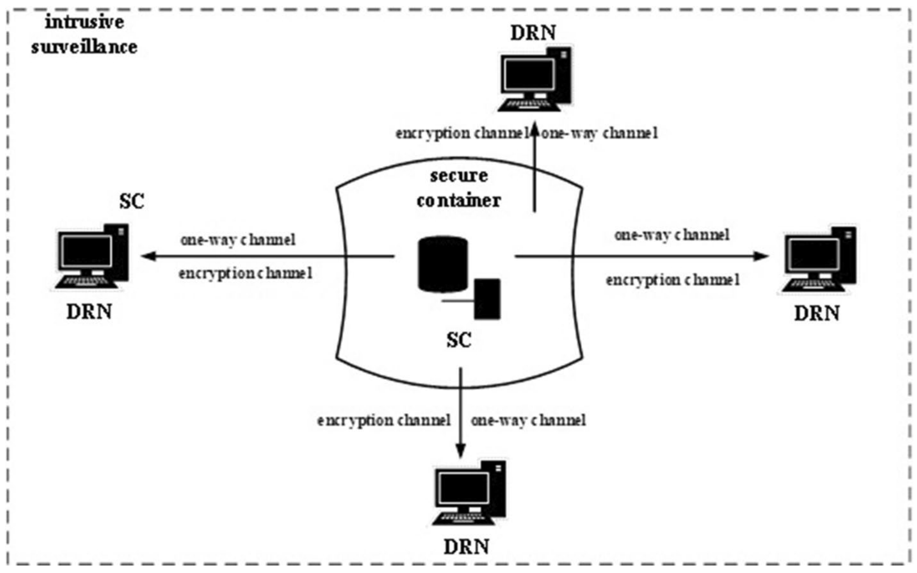

Fig. 4 Data Sharing Based on Blockchain. This image depicts data sharing architecture based on blockchain

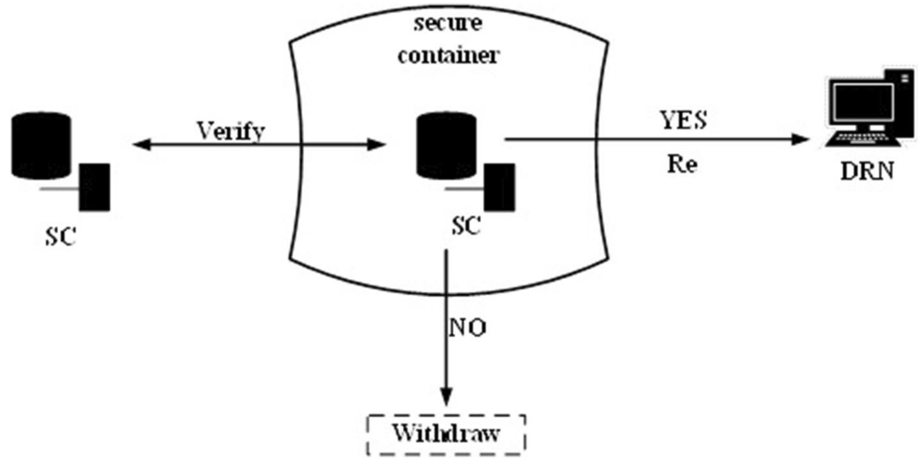

Fig. 5 Validation of Smart Contract. This image depicts the architecture of a smart contract signature. If more than 2/3 of the target's signature is obtained, then the block formed will be recorded on the blockchain

Algorithm 4 Data Sharing Based on Blockchain

```
Input: hash function $H:\{0,1\}^* \to \{0,1\}^*$, public keys set of the target $\{pk_1, pk_2, \cdots, pk_r\}$, public
       key and private key of the security container $SC_{pk}, SC_{sk}$
Output: block $b$
1: for $i = 1$ to $r$
2: $\mathrm{Re}_i \leftarrow Enc_{SC_{sk}}\left(Enc_{pk_i}(\mathrm{Re})\right)$
3: $b_i \leftarrow H(\mathrm{Re}_i) \| time$
4: $h_i \leftarrow h(pk_i)$
5: end for
6: $b \leftarrow b_1 \| b_2 \| \cdots \| b_r$
7: $h \leftarrow \prod_{i=1}^{r} h(pk_i)$
8:if $h = h\left(\sum_{i=1}^{r} pk_i\right)$
9:     for $i = 1$ to $r$
10:        $\mathrm{Re} \leftarrow Dec_{sk_i}\left(Dec_{SC_{pk}}(\mathrm{Re}_i)\right)$
11:        $pk_i \Leftarrow \mathrm{Re}$
12:    end for
13: end if
```

As described in Algorithm 4, each calculated result is encrypted by the private key of the security container and the public key of each target to obtain Re_i. Then, cascading the hash value of Re_i and the time forms the sub-block b_i. h_i is obtained by pk_i using the homomorphic hash h. Then, $b \leftarrow b_1||b_2|| \cdots ||b_r$ and $h \leftarrow \prod_{i=1}^{r} h(pk_i)$ are computed.

4 Conclusion

This study discusses the data storage, data operation and data sharing methods for large amounts of data processing. Using the blockchain data structure combined with intelligent contracts, homomorphic hashes, secure containers, aggregate signatures and one-way encrypted channels, the authenticity, integrity and reliability of data for the collection, calculation and results sharing of astronomical data is ensured. Combined with the SKA project, this scheme can be applied to astronomical data processing. This method provides innovative ideas for the application of blockchain in the fields of large data volume, rapid data generation, high complexity data processing and high value data processing results.

Abbreviations
IoT: Internet of Things; SKA: Square Kilometre Array; DRS: Data Receiving Station; DPN: Data Processing Node; PBFT: Practical Byzantine Fault Tolerance; SDP: Science Data Processor; CSP: Central Signal Processor; TM: Telescope Manager; SaDT: Signal and Data Transport; DSN: Data Supply Node; ASN: Algorithm Supply Node; SM: Smart Contract; DRNs: Data Requirement Nodes.

Acknowledgements
We gratefully acknowledge the anonymous reviewers for taking the time to review our manuscript.

Authors' contributions
All authors contributed to the idea development, study design, theory, result analysis, and article writing. All authors read and approved the final manuscript.

Funding
This research is supported by the Innovative Research Groups of the National Natural Science Foundation of China (Grant No. 61521003), Intergovernmental Special Programme of the National Key Research and Development Programme (2016YFE0100300, 2016YFE0100600), National Scientific Fund Programme for Young Scholar (61672470) and Science and Technology Project of Henan Province (182102210617, 202102210351).

Availability of data and materials
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Author details
[1] State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou 450001, China. [2] School of Computer and Communication Engineering, Zhengzhou University of Light Industry, Zhengzhou 450002, China. [3] School of Computer Science, Fudan University, Shanghai 201203, China. [4] School of Computer Science and Information Security, Guilin University of Electronic Technology, Guilin 541004, China.

Received: 21 April 2020 Accepted: 6 November 2020
Published online: 23 November 2020

References
1. S. Nakamoto, Bitcoin: a peer-to-peer electronic cash system. Consulted (2008)
2. M. Padmavathi, R.M. Suresh, Secure P2P intelligent network transaction using litecoin. Mob. Netw. Appl. **24**, 318–326 (2018)
3. N. van Saberhagen, Cryptonote v 2.0. HYPERLINK (2013), https://cryptonote.org/whitepaper.pdf
4. E.B. Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, M. Virza, Zerocash: Decentralized anonymous payments from bitcoin, in *IEEE Symposium on Security and Privacy (SP)* (IEEE, 2014), pp. 459–474
5. X. Xu, X. Zhang, H. Gao, Y. Xue, L. Qi, W. Dou, BeCome: blockchain-enabled computation offloading for IoT in mobile edge computing. IEEE Trans. Ind. Inform. (2019). https://doi.org/10.1109/TII.2019.2936869
6. Y. Huang, Y. Chai, Y. Liu, J. Shen, Architecture of next-generation e-commerce platform. Tsinghua Sci. Technol. **24**(1), 18–29 (2019)
7. G. Li, S. Peng, C. Wang, J. Niu, Y. Yuan, An energy-efficient data collection scheme using denoising autoencoder in wireless sensor networks. Tsinghua Sci. Technol. **24**(1), 86–96 (2019)
8. Q. Xia, E.B. Sifah, K.O. Asamoah et al., MeDShare: trust-less medical data sharing among cloud service providers via blockchain. IEEE Access **5**(99), 14757–14767 (2017)
9. G. Zyskind, O. Nathan, A. Sandy Pentland, Decentralizing privacy: using blockchain to protect personal data, in *IEEE Security and Privacy Workshops* (IEEE Computer Society, 2015), pp. 180–184
10. J. Kishigami, S. Fujimura, H. Watanabe, et al., The blockchain-based digital content distribution system, in *IEEE Fifth International Conference on Big Data and Cloud Computing*. (IEEE Computer Society, 2015), pp. 187–190
11. L. Liu, X. Chen, Z. Lu, L. Wang, X. Wen, Mobile-edge computing framework with data compression for wireless network in energy internet. Tsinghua Sci. Technol. **24**(3), 271–280 (2019)
12. A. Ramlatchan, M. Yang, Q. Liu, M. Li, J. Wang, Y. Li, A survey of matrix completion methods for recommendation systems. Big Data Min. Anal. **1**(4), 308–323 (2018)
13. X. Xu, C. He, Z. Xu, L. Qi, S. Wan, M.Z. Bhuiyan, Joint optimization of offloading utility and privacy for edge computing enabled IoT. IEEE Internet Things J. (2019). https://doi.org/10.1109/JIOT.2019.2944007

14. L. Hanwen, K. Huaizhen, Y. Chao, Q. Lianyong, Link prediction in paper citation network to construct paper correlated graph. EURASIP J. Wirel. Commun. Netw. (2019). https://doi.org/10.1186/s13638-019-1561-7
15. R.J. Hanisch, G.H. Jacoby, Astronomical data analysis software and systems X. Publ. Astron. Soc. Pac. **113**(784), 772–773 (2001)
16. X. Chi, C. Yan, H. Wang, W. Rafique, L. Qi, Amplified LSH-based recommender systems with privacy protection. Concurr. Comput. Pract. Exp (2020). https://doi.org/10.1002/CPE.5681
17. L. Qi, W. Dou, Y. Zhou, J. Yu, C. Hu, A context-aware service evaluation approach over big data for cloud applications. IEEE Trans. Cloud Comput. (2015). https://doi.org/10.1109/TCC.2015.2511764
18. W. Gong, L. Qi, Y. Xu, Privacy-aware multidimensional mobile service quality prediction and recommendation in distributed fog environment. Wirel. Commun. Mob. Comput. (2018). https://doi.org/10.1155/2018/3075849
19. L. Qi, W. Dou, W. Wang, G. Li, H. Yu, S. Wan, Dynamic mobile crowdsourcing selection for electricity load forecasting. IEEE Access **6**, 46926–46937 (2018)
20. X. Xu, S. Fu, L. Qi, X. Zhang, Q. Liu, Q. He, S. Li, An IoT-oriented data placement method with privacy preservation in cloud environment. J. Netw. Comput. Appl. **124**, 148–157 (2018)
21. C. Zhang, M. Yang, J. Lv, W. Yang, An improved hybrid collaborative filtering algorithm based on tags and time factor. Big Data Min. Anal. **1**(2), 128–136 (2018)
22. Y. Liu, S. Wang, M.S. Khan, J. He, A novel deep hybrid recommender system based on auto-encoder with neural collaborative filtering. Big Data Min. Anal. **1**(3), 211–221 (2018)
23. L. Qi, X. Zhang, W. Dou, Q. Ni, A distributed locality-sensitive hashing based approach for cloud service recommendation from multi-source data. IEEE J. Sel. Areas Commun. **35**(11), 2616–2624 (2017)
24. M. Krohn, M.J. Freedman, D. Mazieres, On-the-fly verification of rateless erasure codes for efficient content distribution, in *Proceedings of IEEE Symposium on Security and Privacy* (IEEE, Lee Badger, 2004), pp. 226–240
25. H. Shi, W. Liu, T. Cao, An improved method of data integrity verification based on homomorphic hashing in cloud storage. J. Hohai Univ. **43**(3), 278–282 (2015)
26. Y. Chao, X.U. Mi-Xue, S.I. Xue-Ming, Optimization scheme of consensus algorithm based on aggregation signature. Comput. Sci. (2018)
27. O.T.D.C. Miguel, Practical byzantine fault tolerance. ACM Trans. Comput. Syst. **20**(4), 398–461 (2002)
28. P.C. Broekema, R.V. Van Nieuwpoort, H.E. Bal, The square kilometre array science data processor. Preliminary compute platform design. J. Instrum. **10**(7), C07004–C07004 (2016)
29. L. Fiorin, E. Vermij, J. Van Lunteren, et al. An energy-efficient custom architecture for the SKA1-low central signal processor (2015), pp. 1–8.
30. K. Moreland, E. Angel, The FFT on a GPU, in *ACM Siggraph/eurographics conference on graphics hardware* (Eurographics Association, 2003), pp. 112–119
31. E. Suter, H.A. Friis, E.H. Vefring, et al. A novel method for multi-resolution earth model gridding, in *SPE reservoir simulation conference*, 20–22 February, Montgomery, Texas, USA (2017)
32. C.P. Cahill, J. Martin, M.W. Pagano, et al. Client-based authentication technology: user-centric authentication using secure containers, in *Proceedings of the 7th ACM Workshop on Digital Identity Management* (ACM, 2011), pp. 83–92

Publisher's Note