International Journal of Information Technologies and Systems Approach
Volume 17 • Issue 1

BitTrace:
A Data-Driven Framework for Traceability of Blockchain Forming in Bitcoin System

Jian Wu, Southwest University, China*
Jianhui Zhang, Southwest University, China
Li Pan, UCSI University, Malaysia

ABSTRACT

Bitcoin is a digital currency system built on the foundation of fairness. However, some malicious miners, driven by their own interests, employ unfair tactics such as selfish mining to compete, which disregard the legitimate miners' investments in computational power and energy consumption. In order to assess the efficiency of the Bitcoin system in real-time and promptly detect malicious miners in the network, this paper proposes a data collection framework called BitTrace, which addresses the issues of low efficiency, lack of timeliness, and data loss in traditional data collection frameworks. BitTrace enables real-time collection and analysis of the blockchain formation process, storing it as structured data. Furthermore, the paper discusses factors that influence the efficiency of data collection and proposes a topological control scheme based on the DPC algorithm to enhance the integrity and efficiency of data collection. Researchers can explore various research areas and applications, such as selfish mining detection and legitimate mining strategy research.

KEYWORDS

BitTrace, Blockchain Formation Process, DPC, Topological Control, Traceability

INTRODUCTION

Bitcoin, the first and most well-known application of blockchain, is a decentralized digital currency that operates on a peer-to-peer network. The Bitcoin blockchain serves as a public ledger that records all transactions of the cryptocurrency (Catalini & Gans, 2016).

Despite the tremendous success of Bitcoin as a digital currency, it has long been criticized for issues such as performance and resource consumption (Chen et al., 2022). Reasonable competition is unavoidable in the Bitcoin network. However, if a significant number of malicious miners emerge and employ selfish mining tactics to gain higher profits (Yang et al., 2020; Wang et al., 2022), it becomes unfair for legitimate miners. Legitimate miners invest substantial computational power and energy into mining, yet they do not receive the deserved rewards (Franzoni et al., 2022). We aim to address these concerns by obtaining real-time data from the Bitcoin network to evaluate its

DOI: 10.4018/IJITSA.339003 *Corresponding Author

efficiency. With this data, we can further analyze cheating behaviors within the network and make reasonable adjustments to miners' mining strategies to improve mining efficiency. We are committed to developing a real-time assessment framework for the Bitcoin network, aiming to detect nodes engaging in unfair mining practices and enhance the mining efficiency of legitimate miners within the network. To achieve this goal, it is crucial to obtain relevant data regarding the formation process of the blockchain in the Bitcoin network.

Traceability, known as the ability to trace the process of transactions, is the most noted fundamental characteristic of blockchain technology. For example, a company can record every step of the drug production process on the blockchain, allowing users to trace each stage of drug creation (Martin et al., 2020; Xu et al., 2020). In reality, the formation process of the blockchain in the Bitcoin network is transparent and non-traceable due to resource and efficiency considerations. For users, it is not necessary to know the formation process of the blockchain. But for researchers, inability or failure to trace the formation process of the blockchain makes it difficult to address certain issues. For instance, miners cannot determine if the received block is stale, and users cannot measure the performance of the Bitcoin system. While some blockchain systems offer API interfaces (Harry et al., 2020) to assist researchers in accessing data, these interfaces often fall short in efficiency, performance, and latency, and thus are unable to meet the requirements for real-time analysis. Some existing work (Wang et al., 2022; Zheng et al., 2022; Wong et al., 2019; Mohanta et al., 2020; Chen et al., 2020; Grossman et al., 2017) has proposed methods and implementations to obtain more relevant data in Bitcoin or other blockchain systems to address these issues. But these studies face various challenges regarding data granularity, non-traceability of the blockchain formation process, and the scalability of the framework. Currently, there is no data collection solution specifically designed for the blockchain formation in Bitcoin system, which makes it challenging to conduct research based on the real data from the Bitcoin network.

In this paper, we introduce BitTrace, a data collection framework for blockchain formation in the Bitcoin network. BitTrace is designed using the principles of microservices and a layered architecture. Each layer is responsible for specific functions, including monitoring, data collection, sending, reception, parsing, and storage. Furthermore, each layer can be scaled according to the data volume, ensuring optimal performance of the framework. The framework offers the following advantages:

1. **Fine-grained:** We have defined the states and events during the process of blockchain formation and employed code instrumentation techniques to monitor the nodes, ensuring the capture of all relevant data related to the formation of blockchain blocks. Fine-grained data guarantees the accuracy of subsequent research results and enables the applicability of this data to a wider range of studies.
2. **Real-time:** Once a state transition or an event occurrence is triggered, the monitoring program promptly processes the relevant data, which allows researchers to obtain real-time and authentic data. Real-time data is crucial for security research, as security protocols can detect network states and identify unauthorized miners in real time.
3. **Traceability and observability:** The data collected by BitTrace is stored using standardized data structures that can describe the process of blockchain formation. This facilitates researchers to trace and observe the entire process.
4. **Scalability:** BitTrace uses the principles of microservices and a layered architecture. It supports scaling modules to efficiently handle more data and incorporating multiple plug-ins to perform different tasks, providing strong scalability and extensibility.

This paper makes the following contributions:

1. **Introduction of BitTrace:** The paper presents the design principles, system framework, and design concepts for each module of BitTrace.
2. **Improved data collection efficiency and comprehensiveness:** The paper discusses the factors that enhance data collection efficiency and comprehensiveness. It proposes a Bitcoin node topology control method based on the density peak clustering (DPC) algorithm, which effectively improves the efficiency and comprehensiveness of data collection in BitTrace.
3. **Geolocation plug-in for identifying malicious miner nodes:** The paper develops a plug-in based on BitTrace to identify the geographical location of malicious miners.

The paper is organised as follows. In Section II, we introduce the related work and discuss their advantages and disadvantages. Section III presents the fundamental concepts used in the article. Section IV describes the implementation of BitTrace and topological control scheme based on the DPC. The article is summarized in Section V, which also provides an outlook for future research.

RELATED WORK

Although the Bitcoin system has many advantages over traditional transaction systems, there are still several problems (Guo & Yu, 2022; Mohanta et al., 2020). To address these problems, many researchers have proposed solutions, such as the Bitcoin Lightning Payment Network (Sarode et al., 2023), which can improve the overall transaction processing capacity, throughput, and smart contracts that provide more efficient, secure, and reliable solutions for transactions and execution in various fields by addressing issues such as trust, intermediaries, and security (Taherdoost, 2023). Many security problems, however, remain to be solved, such as performance monitoring and malicious node detection. Research has shown that the mining strategy employed by miners can affect their earnings (Wu et al., 2019; Liu et al., 2022). As the difficulty of independent mining increases, more and more miners are changing their mining strategies, and some even resort to dishonest mining strategies to gain more stable profits (Motlagh et al., 2021; Yang et al., 2020). The reason for this is that most blockchain systems do not provide enough relevant data to solve the problem in order to gain greater operational efficiency. Therefore, researchers need other data collection methods to obtain operational data and performance metrics of blockchain systems to address these problems.

To analyze the selfish mining behavior in the Bitcoin network, it is necessary to obtain data from the formation process of the Bitcoin network (Wang et al., 2022). Some researchers, such as Harry et al. (2020), have utilized the remote procedure call (RPC) interfaces of blockchain systems to retrieve real data from the network and analyze transaction data in certain blockchain systems. However, the method of collecting data using the RPC interface incurs high time costs and provides limited information. Other researchers, like Zheng et al. (2022), acknowledged the limitations posed by RPC interfaces and adopted a different approach by directly accessing the raw logs file of the blockchain system. They used regular expressions to extract performance through data analysis. However, it is worth noting that this method is not without its own limitations. Firstly, the volume of log data available in blockchain systems is inherently limited, which can impact the comprehensiveness of the analysis. Additionally, the data obtained through the application of regular expressions is context-specific and may not fully represent general experimental data.

Some researchers have deployed probing programs within blockchain nodes to monitor the operational data of nodes and the blockchain. Chen et al. (2020) proposed a method to collect and analyze data related to transactions, smart contract creation and invocation, and node transaction logs in Ethereum. By analyzing the data, they identified the presence of risks such as malicious accounts and malicious contract invocations within the network. Grossman et al. (2017) obtained information on internal transactions and storage operations in smart contracts by inserting code and analyzed related

security risks, such as DAO attacks. Their research focused more on Ethereum's smart contracts. To date, the research on code instrumentation in the Bitcoin system has remained relatively limited.

In summary, the aforementioned limitations in current research on data collection during the formation process of the Bitcoin network make it difficult to monitor and trace the formation process of the Bitcoin network in real time since they fail to meet the requirements for fine-grained and comprehensive data collection. Data is crucial for experimental analysis, and obtaining real and comprehensive data is the key to data collection. Security analysis of the Bitcoin network, such as selfish mining, triangle attacks (Wang & Li, 2019), blockchain visualization (Tri et al., 2017), and stale block generation, relies on data from the formation process of the Bitcoin network.

FUNDAMENTAL CONCEPTS

This section explains the blockchain in Bitcoin and the blockchain formation model of Bitcoin and other related fundamental concepts.

Blockchain in Bitcoin

The blockchain is the key technology of the Bitcoin system (Yaga et al., 2018). Only one miner can gain bookkeeping rights in one consensus process and mine the block. Even honest miner nodes will inevitably generate multiple blocks of the same height due to competition for bookkeeping rights. These blocks reach the receiving node in sequence depending on the network location of the miner node that mined the block. Nodes will connect blocks of the same height to their parent blocks, which results in a fork of the blockchain. Therefore, the blockchain maintained by the Bitcoin node is a tree structure with the Genesis block as the root node and blocks of different heights as the nodes of the tree, as shown in Figure 1. We use the optimal node (optimal block) to express the set of the deepest leaf nodes that are the earliest to be connected to the tree. In common sense, the chain is a collection of nodes that starts from a leaf node and traces back to the root node by the parent-child relationship. The chain with the optimal block is usually defined as the main chain of the blockchain. Other chains are defined as side chains of the blockchain. The node with the deepest depth on the side chain is usually defined as the side chain optimal node (i.e., the chain optimal block).

Blockchain Formation Model

For a block of a miner to be added to the blockchain, the miner needs to broadcast it to the network, but it is for all the full nodes to reach an agreement if this block or another block should be added to the blockchain through the consensus protocol. This process is called block synchronization. This paper illustrates the complete process of block synchronization in Figure 2 (Luo & Zhang, 2023). It consists of six steps: block retrieval, block validation, block type confirmation, block chain synchronization, orphan pool processing, and chain verification.

The blockchain-forming process from the perspective of nodes is the process of multiple block synchronization of nodes. The time for nodes in different network locations to receive a block varies greatly, with current research (Yahya et al., 2022) showing that it takes at least 10s for an 11M block to propagate to 90% of the nodes in the network. As a result, different nodes synchronize blocks of the same height at different times, which directly leads to significant differences in the blockchain-forming process from the perspective of different nodes.

The process of blockchain formation from a multi-node perspective involves nodes reaching a consensus on block validation through information exchange, which is essentially the process of achieving PoW consensus in the network (Guo et al., 2021). The Bitcoin network encourages miner nodes to participate in the consensus process of PoW to verify and record Bitcoin transactions in the network. Miner nodes calculate and solve a hash problem in order to earn the right to add a block. The miner who calculates the result first gains the right to add the block and includes all Bitcoin transactions during a specific time period, sequentially linking them to the blockchain. Additionally,

Figure 1. Blockchain Structure

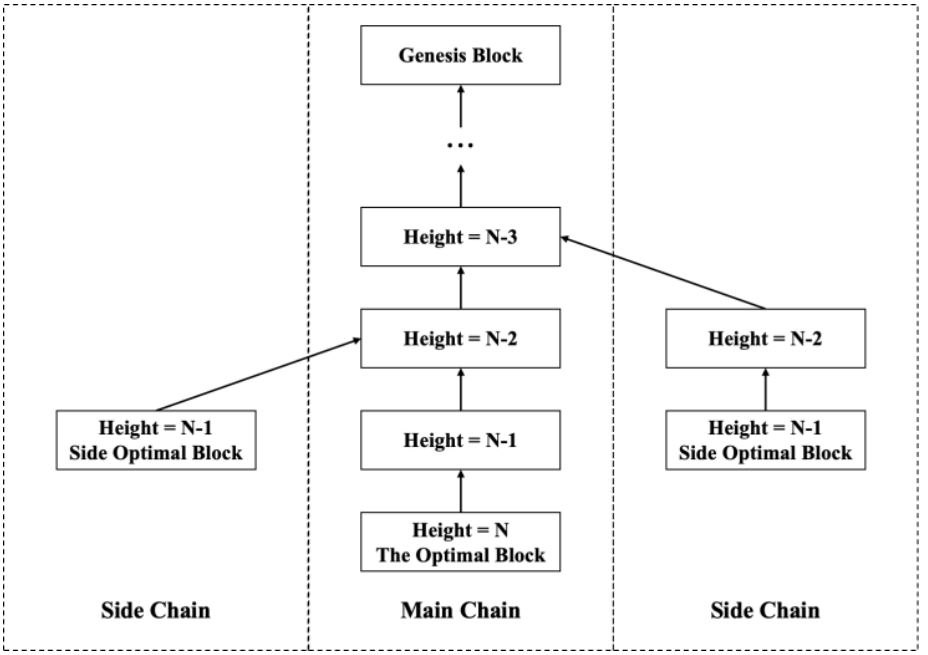
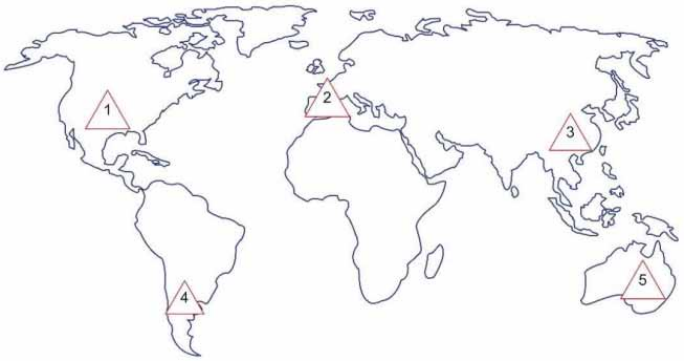


Figure 2. Bitcoin Blockchain Synchronization Flowchart



the miner broadcasts the block to the Bitcoin network to assist other nodes in synchronization. Upon receiving the new block, other nodes will synchronize either all blocks or only the block headers based on their capabilities. Miners who have not earned the right to add the block will stop their current computations after synchronizing the new block, and then choose the new block for the next mining. Consensus on block validation is achieved once all nodes in the network complete block synchronization.

METHODOLOGY

This section provides a detailed account of the BitTrace architecture and the working principles of its components. Various influencing factors are explored in order to enhance data integrity and

efficiency. A method is proposed based on the DPC algorithm to identify high-quality nodes in the Bitcoin network, and results show that this algorithm outperforms other clustering algorithms. Additionally, a method for locating Bitcoin miners is presented to aid in detecting malicious miners in the Bitcoin network.

Design Principles

This section introduces the basic design principles of BitTrace.

1. **Microkernel Architecture:** The microkernel architecture is a scalable architecture that emphasizes functional separation. It separates the core system responsible for common functionalities from the plug-in modules responsible for business logic. This separation allows for rapid and flexible expansion without compromising the stability of the entire system. BitTrace follows the principles of the microkernel architecture by separating the core system from the plug-in modules for business logic. The core system adopts a layered architecture, with different modules implementing core functions such as data collection, analysis, and storage, while ensuring scalability and extensibility.
2. **Generality:** BitTrace is applicable to various types of Bitcoin nodes since it considers the commonalities among different types of nodes. This generality ensures compatibility and adaptability across different node configurations.
3. **Scalability:** Each component of BitTrace is implemented with standardization, allowing for easy expansion and contraction based on varying workload conditions. This scalability enables researchers to adapt BitTrace to different scenarios and requirements.
4. **Extensibility:** The layered architecture of BitTrace enables researchers to accomplish different tasks by designing additional plug-in modules. This scalability allows for the integration of new functionalities and the exploration of diverse research directions.

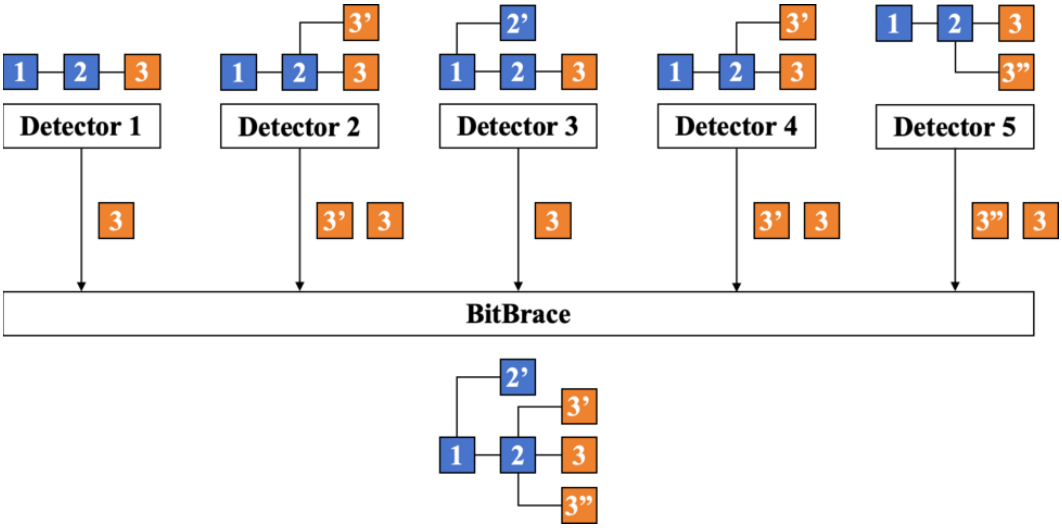
Guided by these design principles, BitTrace provides a framework that is scalable, flexible, and adaptable to different types of Bitcoin nodes, facilitating data collection, analysis, and exploration in various research areas.

Framework Architecture

Figure 3 illustrates the overall architecture of BitTrace. The architecture includes the Detection Nodes (consisting of Code Instrumentation and Exporter), Agent, Storage, and Plug-in. Each component is described as follows.

1. **Detection Nodes.** These nodes are responsible for data collection within the Bitcoin network. They consist of two main components:
    - **Code Instrumentation.** This component applies code instrumentation techniques to the Bitcoin nodes, allowing the detection and analysis of their behavior.
    - **Exporter.** The Exporter component retrieves detailed real-time data by monitoring the instrumented code in the Bitcoin nodes.
2. **Agent.** Deployed outside the Bitcoin network and acting as a data receiver and processor, the Agent communicates with the Detection Nodes and performs the following tasks:
    - **Receiver.** The Receiver component receives the information outputted by the Exporter of the Detection Nodes.
    - **Resolver.** The Resolver component is a cluster of message processors that parse and process the received data in parallel.
3. **Storage.** The Storage component is responsible for receiving and storing the data generated by the Agent. It maintains a structured data structure to store the collected information.

Figure 3. Overall Architecture of BitTrace



4. **Plug-in.** The Plug-in module implements specific business logic based on a microkernel architecture. It can be customized for different application scenarios or research purposes. Examples include plug-ins for handling stale blocks and data visualization.

The communication between components can utilize various standard communication protocols. Employing standardized communication interfaces proves advantageous in software design and facilitates the scalability of components.

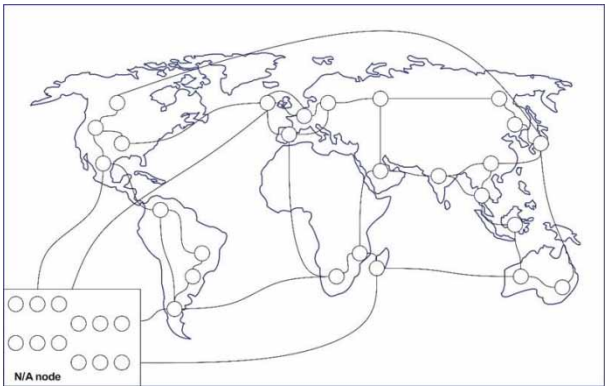
Code Instrumentation

Code instrumentation refers to the practice of adding additional code or instructions to a software program in order to gather information on its behavior, performance, or execution. This technique is commonly used in software development, debugging, profiling, and monitoring processes. In this paper, code instrumentation is applied to create the detection nodes for BitTrace.

As previously mentioned, the block synchronization process in the Bitcoin network falls into three steps: block validation, block type validation, and block synchronization. These three processes can be further broken into six sub-processes, namely block detection, orphan block detection, main block detection, block linking to the main chain, block linking to side chain, and linking actions leading to main-side chain exchange. These behaviors correspond to specific functions in the underlying code of Bitcoin nodes. For example, when the block detection occurs, the underlying code calls the function checkBlockSanity. These mappings are illustrated in Figure 4. By instrumenting the code, we send the required data to our data center when specific behaviors occur in Bitcoin, in addition to invoking the corresponding functions.

In the perspectives of other nodes in the Bitcoin network, a detection node using code instrumentation appears no different from regular honest Bitcoin nodes. Its behavior is not regarded as malicious. Compared to other data collection approaches, instrumentation does not disrupt normal communication in the network. The detection nodes are not identified as malicious nodes, allowing them to operate stably over a long term.

Figure 4. Code Instrumentation



Data Resolving and Storage

This section presents a detailed account of how the Resolver parses the data collected from the detection nodes regarding the blockchain formation process in a standardized manner. First, the following concepts are defined in this context:

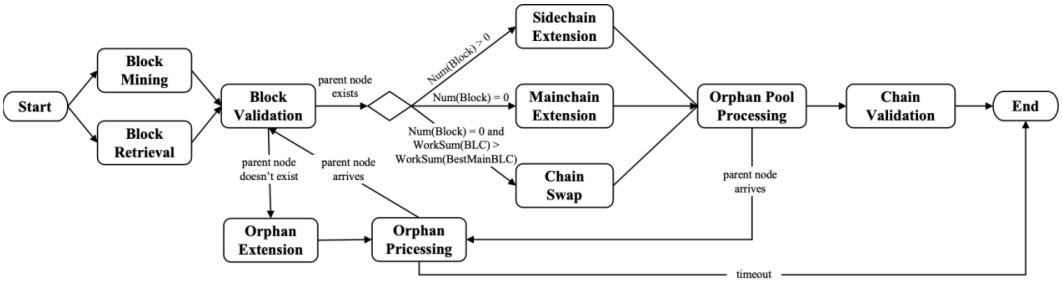
1. **State:** It refers to a certain set of attributes related to the block synchronization of a node at a certain time that we obtain through the Exporter. For example, block state refers to a set of block attributes, blockchain state refers to a set of blockchain attributes, etc.
2. **Snapshot:** It refers to a copy of all the states in a client node at a particular time. This copy allows the user to restore the state of the data in the node at the time when the copy was taken.
3. **Event:** As the fundamental cause of various types of state changes in a node, it refers to a fine-grained behaviour that is occurring or has occurred in a bitcoin node, such as detecting whether a block is complete and regular.
4. **Revision:** As the direct cause of a change in a set of various attributes in a node, it refers to a change brought by the node to the attribute set after an event has occurred in the node.
5. **Timeline:** As a chronology of the changes of states in nodes, which preserves the timing order of Event and Revision, and also contains the timing order of state sets and snapshots.

Briefly, according to the block synchronization process, the Resolver parses the data the Receiver received to generate a snapshot of the node's data at different moments in time. To ensure that the block synchronization process can be restored with high fidelity, the Resolver simultaneously records the events that cause the snapshot revision in a fine granularity.

In practice, a snapshot can only describe the state of a node at a specific time and cannot capture the temporal information or track the process of blockchain formation. The introduction of the concepts of Event, Revision, and Timeline (Bowden et al., 2020) addresses this issue, as shown in Figure 5. The parsing program parses and records fine-grained Events that cause state changes in a node, along with the Revisions executed by the node due to those Events. The parsing program links these Events and Revisions in chronological order and submits them to the Storage for storage.

Based on the aforementioned principles and techniques, this paper adopts a complete data collection solution for tracking and observing the blockchain formation process in the Bitcoin network. The data collected using this framework enables the observation of various states of the blockchain at specific times. Additionally, the data allows the identification of the reasons and processes behind the state changes between Snapshots through Events, Revisions, and the Timeline. By analyzing the

Figure 5. Events and Revisions in Timeline



data within the framework, one can gain insights into the dynamics and evolution of the blockchain, facilitating a deeper understanding of its behavior and development over time (Shahsavari et al., 2020).

In summary, our framework offers the following advantages:

- **Fine-grained data:** The framework can collect fine-grained data from various Bitcoin nodes' operations.
- **Scalability:** The framework allows for efficient processing of more data by increasing modules.
- **Extensibility:** The framework supports the insertion of multiple plug-ins for different tasks.

By leveraging these advantages, our framework enables comprehensive analysis and understanding of the blockchain formation process in the Bitcoin network, facilitating further research and applications in the field.

Data Collection and Stale Blocks Detection

To validate the feasibility of the framework, this paper selects three different types of Bitcoin nodes, namely Bitcoin Core nodes, Bitcoin full nodes, and SPV nodes, to serve as detection nodes. Additionally, the Agent and Storage modules were deployed outside the Bitcoin network to receive, parse, and collect data. Finally, the developed stale block detection plug-in was utilized to analyze the structured data stored in the Storage.

The criteria for identifying stale blocks were as follows:

1. The block is not part of the main chain.
2. The total accumulated work of the chain connected by the block is less than the work of the main chain.
3. The best block height of the chain connected by these blocks is six blocks lower than the best block height.

We collected real data from the Bitcoin network and preserved detailed data on the synchronization

Table 1. Types and Quantity of Data Collection

	Snapshot	State	Event	Revision
Quantity	134,220	536,899	802,757	825,791

process of these blocks, with specific details shown in Table 1. On average, for each block

synchronization, the framework collected 112 relevant data points, totaling approximately 3.2 KB of data.

Optimization for Data Collection

Data analysis generated a total of 20,817 blocks, including 426 stale blocks, 2,327 orphan blocks, and 18,064 normal blocks.

In the previous section, a methodology was proposed to collect real data on the process of blockchain formation from the Bitcoin network. We deployed a single Bitcoin node as a detection node in the experiment to verify the effectiveness of the methodology. However, the results show that deploying only one detection node in a large-scale Bitcoin network resulted in low timeliness of the collected data, as well as compromised data integrity and trustworthiness. In this section, we will investigate how to improve the efficiency of data collection.

Quantity of Linked Nodes

The real data of the Bitcoin network comes from two sources: the data generated by the detection nodes and the nodes directly linked to detection nodes. Theoretically, obtaining more data on blockchain formation facilitates a comprehensive analysis of security vulnerabilities in the network. It is necessary to establish connections with as many nodes as possible, with each node capable of linking up to 150 nodes.

In the experiment, we randomly selected 150 Bitcoin nodes located in the same continent from the global Bitcoin network, and deployed 10 detection nodes respectively linked to 10, 30, 50, ..., 150 nodes. We instructed these 10 detection nodes to collect blockchain data during a specific time period. The experimental results are presented in Table 2.

The experimental results show that the increase in the number of linked nodes is accompanied by the increase in the number of collected blocks. This is because some nodes experience high system load and poor network conditions, which can affect the efficiency of block synchronization. Linking to as many nodes as possible helps mitigate the impact of these factors and contributes to improving the comprehensiveness of data collection.

Quantity of Detection Nodes

Besides the number of linked nodes, the quantity of detection nodes also has a significant impact on data collection. We conducted an experiment that deployed five detection nodes in different geographical locations, each linked to 150 nodes. The variations in data volume turned out to be noticeable among detection nodes located in different geographical locations. This demonstrates the limitations of deploying only one detection node, as comprehensively collecting blockchain formation data from the entire Bitcoin network poses a great challenge. Moreover, the collected data is susceptible to network fluctuations, making it difficult to utilize block delays for identifying malicious nodes.

Topology Structure

In the field of communications, the technique of adjusting the communication ranges and connectivity of network nodes in a wireless sensor network to enhance overall network performance and quality is known as topology control. Topology control can effectively improve energy efficiency, reduce communication energy consumption, and enhance network fault tolerance.

Table 2. Data Collected for Different Numbers of Linked Nodes

Linked Nodes	10	30	50	70	90	110	130	150
Block	11,890	11,820	11,940	12,030	12,149	12,120	12,747	12,854

Bitcoin is known to have no central node. In reality, some nodes have better network conditions and faster interaction speeds, resulting in them being connected to a larger number of nodes than others. Compared to nodes with unstable network environments, these nodes are more sensitive to changes in the Bitcoin network and are more likely to collect comprehensive data. As shown in Figure 6, Node 1 can obtain information about blockchain changes from other nodes with a maximum of two steps. Node 1 is, therefore, more suitable as a detection node compared to Node 2.

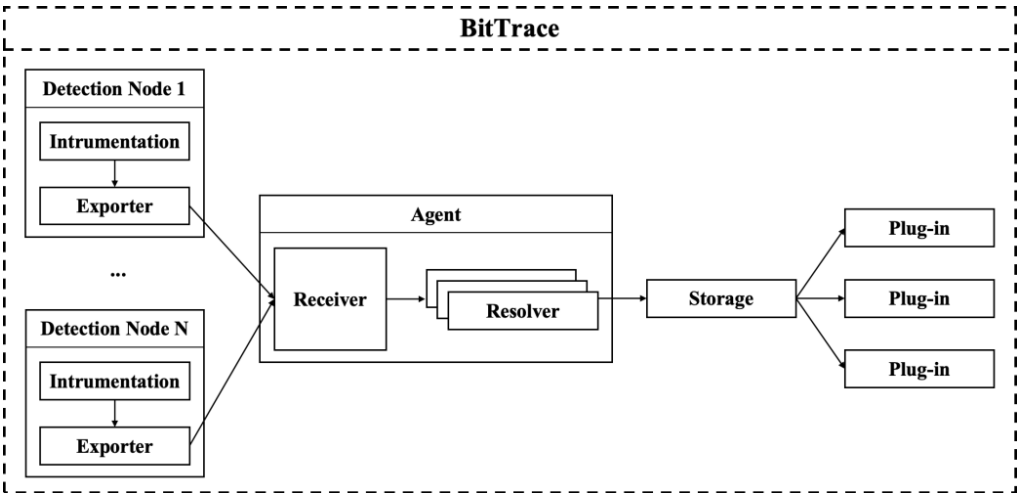
In order to find out these nodes in Bitcoin network, we propose a topology control method for Bitcoin nodes based on the DPC algorithm. The density peak clustering (DPC) algorithm was initially proposed by Rodriguez and Laio (2014). It is a clustering algorithm used to group data points into clusters with high density. The DPC algorithm is intended to achieve effective clustering by computing the local density and distance of data points to identify density and cluster centers. In contrast to some traditional clustering algorithms, the DPC algorithm requires no prior knowledge of the number of clusters and is capable of effectively clustering clusters of arbitrary shapes and densities. In comparative experiments with clustering algorithms like DBSCAN and K-means (Wu et al., 2021), the DPC algorithm demonstrates superior performance and effectiveness.

In the subsequent work, we will employ the DPC algorithm to analyze the clustering centers of all nodes in the Bitcoin network, considering the network's topology and clustering results to generate a certain number of clusters. The detection nodes will then be linked to the cluster center nodes, allowing for the collection of a more comprehensive blockchain-formed data from the Bitcoin network while minimizing the impact of resource consumption and network fluctuations.

The first step is to calculate the local density of each data point in the given dataset, which corresponds to the number of data points in its surrounding neighborhood. Here, we employ the distance measurement of routing distance between nodes in the Bitcoin network. Due to the non-uniqueness of paths between nodes in the Bitcoin network, we need to first calculate the shortest routing distance between any two nodes. Assuming the set of all nodes is represented as $X=\{X_1, X_2, ...\}$, we use Dijkstra's algorithm to compute the shortest routing distance between any two nodes, X_i and X_j.

$$d_{ij} = Dijkstra\left(X_i, X_j\right) \tag{1}$$

Figure 6. Example of the Topology in Bitcoin Network



Once the distances between nodes are determined, the local density of each node, denoted as A, is calculated. Typically, the local density is determined by counting the number of data points within a neighborhood radius. Considering the characteristics of the Bitcoin network, we define a truncation distance, denoted as d_c, as the neighborhood radius. Assuming the set of seed nodes in the Bitcoin network is represented as $S=\{S_1, S_2, ...\}$, they are a set of consistently online and publicly accessible nodes voluntarily provided by network participants, aimed at maintaining the stability of the Bitcoin network. The truncation distance d_c is defined as half of the shortest distance between the seed nodes S_n and S_m, which are the two closest seed nodes to node X_i. If node X_i itself is a seed node, d_c is defined as half of the distance to the nearest seed node from X_i.

$$\rho_i = \sum_j \varphi\left(d_{ij} - d_c\right) \tag{2}$$

$$d_c = \frac{d_{nm}}{2} \tag{3}$$

$$\varphi\left(d\right) = \begin{cases} 1, d < 0 \\ 0, d \geq 0 \end{cases} \tag{4}$$

The magnitude of the local density reflects the data distribution at the location of a data point. In clustering algorithms, data points with higher local density are typically considered as the core or central points of a cluster since they are surrounded by data points of a denser concentration. On the other hand, data points with lower local density may be regarded as outliers or noise points since they have fewer data points in their proximity.

The second step is to calculate the relative distance. The relative distance, denoted as ´ , refers to the minimum distance between a data point and other points of a higher relative density. Its value is the minimum distance between node *i* and all nodes with a higher density than it. If node *i* is the node with the highest density, the value is calculated as the maximum distance between node *i* and other nodes.

$$\delta_i = \begin{cases} \min\limits_{\rho_j < \rho_i} d_{ij}, \rho_i \neq \rho_{max} \\ \max\limits_{j \in C} d_{ij}, \rho_i = \rho_{max} \end{cases} \tag{5}$$

According to the DPC algorithm, the node with the highest local density is considered a density peak, and it is a cluster center. We define its relative distance as the maximum value. We need to find cluster centers among the remaining nodes that satisfy two conditions: a high local density and a large relative distance. We use a decision value ³ to identify these cluster centers.

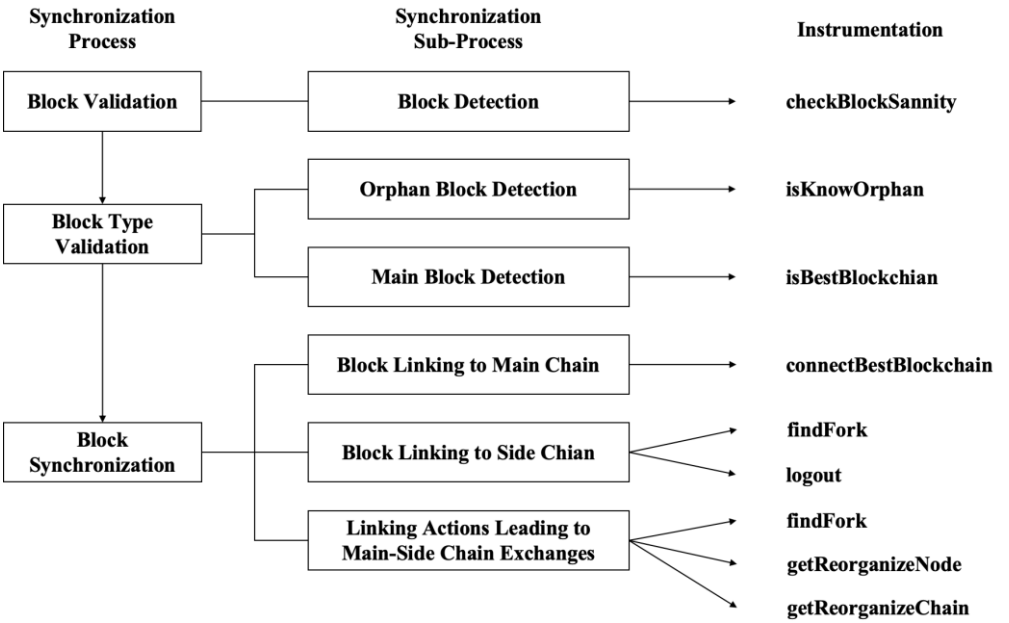
$$³_i = Á_i \times ´_i \tag{6}$$

Provided that a node has both a high local density and a relative distance, it is likely to be a cluster center. The non-central nodes are then assigned to the cluster with the highest density that is closest to them. Finally, the detection nodes are linked to the cluster centers.

Experiments and Analysis

BitcoinNode provided the API for developers to obtain the information of all reachable nodes in the current Bitcoin network. The results indicate that 16,534 nodes are reachable with IP addresses in the current Bitcoin network, with approximately 57% of them labeled as N/A. The owners of these

Figure 7. The Geographical Location of Bitcoin Nodes



nodes have concealed their locations with onion IP or private IP addresses, a measure to protect their privacy and security. Consequently, we are unable to trace the geographical locations of these nodes based on their IP addresses. The geographical location of all reachable nodes is depicted in Figure 7.

Using BitcoinNode API to request the routing information of all reachable nodes is the first step to calculate the routing distance between nodes. The results reveal that we received responses from 75.84% of the nodes. The remaining 24.16% of nodes rejected our requests to prevent potential network attacks. Ultimately, we obtained the routing tables from 12,538 nodes. Using the routing information, we calculated the routing distance d_{ij} between all nodes and used the improved DPC algorithm proposed in this paper to calculate the local density and relative distances for all nodes. The calculation results are shown in Figure 8. The x-axis represents the local density of the nodes, while the y-axis represents the relative distance.

The DPC algorithm determines that nodes of a high local density and a relative distance are more likely to be cluster centers. The results in Figure 8 indicate that nodes in the upper-right corner are more likely to become cluster centers. To further confirm this, we calculate the decision values as shown in Figure 9, where the x-axis represents the number of nodes, and the y-axis represents the decision values.

In order to minimize modifications to the topology structure while maximizing the amount of collected data, we conducted experiments to test the data collection under different decision values. The experimental results indicate that the number of nodes with higher decision value was too small, resulting in incomplete data collection by the detection nodes. On the other hand, connecting the detection nodes to cluster centers with too-low decision values increases modifications to the topology structure instead of contributing to the data volume. We ultimately used a decision value of 75 and selected 20 cluster centers.

Finally, by modifying the routing tables of the detection nodes, we establish individual links between the detection nodes and the cluster centers of their respective continents to observe the amount of data collected during the blockchain formation process. Figure 10 shows the distribution of detection nodes deployed. Additionally, this experiment compares the scenarios of not linking to

Figure 8. Local Density and Relative Distances

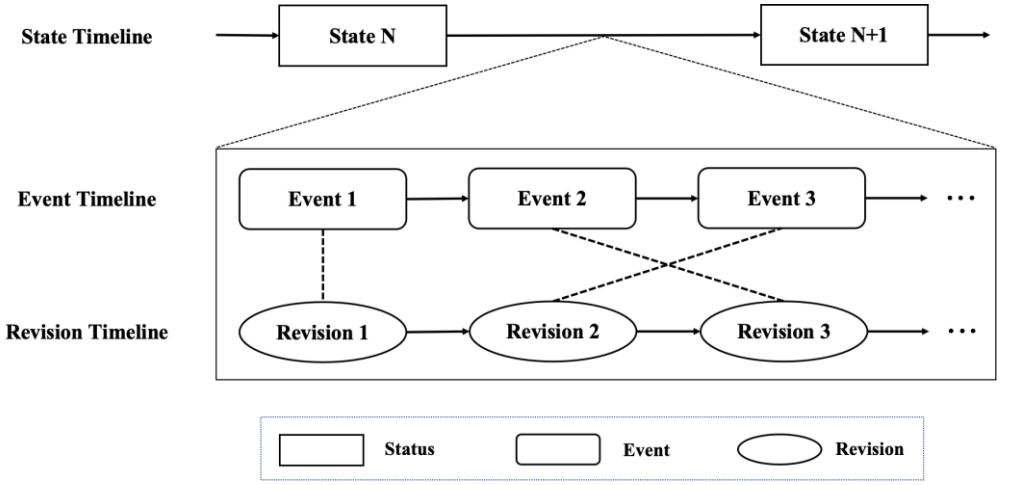
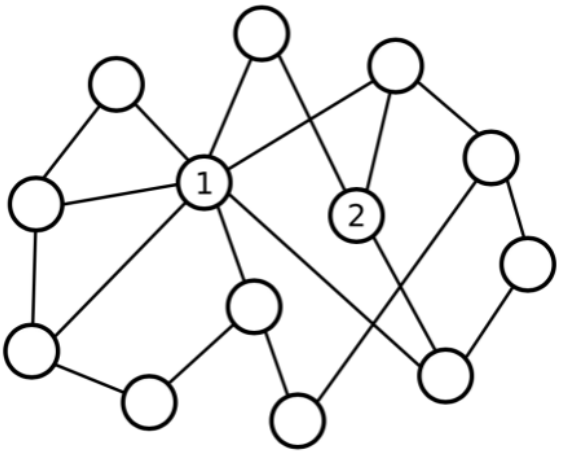


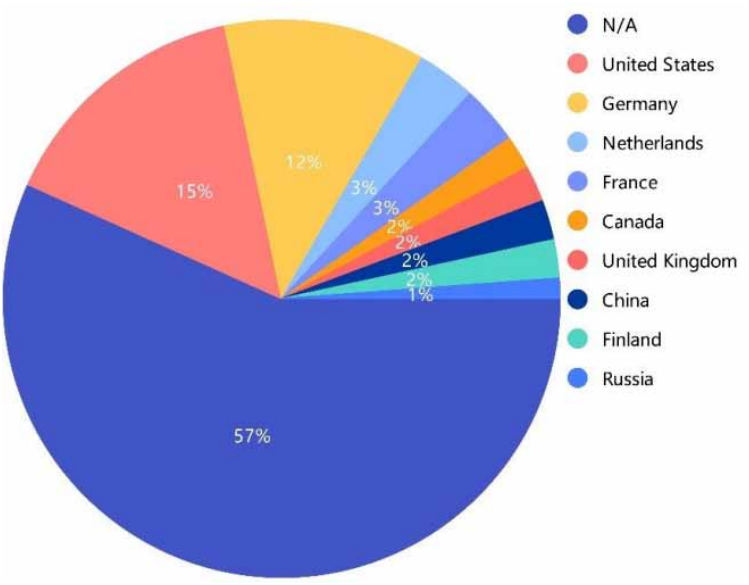
Figure 9. Decision Value and Number of Nodes



cluster centers, using the DBSCAN algorithm, and using the K-means algorithm for cluster analysis. The experimental results demonstrate that linking the detection nodes to the cluster centers enables the collection of more comprehensive blockchain formation data. Compared to the other two clustering algorithms, the DPC algorithm identifies more orphan blocks and stale blocks, and dramatically reduces the time for detection nodes to receive block data, thus improving data collection performance.

We collected real data from the Bitcoin network between October 1, 2022, and December 31, 2022, encompassing a total of 13,348 blocks ranging from height 756,390 to 769,738. To standardize the data collection process, this study categorizes the data collected by detection nodes into four types: Snapshot, representing the snapshots of the blockchain's state at different time points during synchronization; Revision, representing a collection of events that cause state changes; State, representing a collection of blockchain states; and Event, representing a collection of blockchain events. Combining these four types of data, we can describe the blockchain evolution process from the perspective of each detection node. It is important to note that the data collected by different

Figure 10. The Distribution of Detection Nodes



detection nodes may vary due to their respective network environments. The data collected in our experimental environment are presented in Table 3.

By analyzing the data from different detection nodes, it becomes evident that the quantities of normal blocks, orphan blocks, and stale blocks differ from the perspective of each detection node. The results are shown in Table 4. Further processing of the data is required for it to be used in data analysis.

Table 3. Types and Quantities of Data Collected

	Event	Snapshot	Revision	State
Detector 1	27	13,551	66,408	27,003
Detector 2	38	13,659	66,394	26,987
Detector 3	46	13,530	66,702	27,014
Detector 4	62	13,546	66,512	27,028
Detector 5	43	13,524	66,687	27,011

Table 4. Types and Quantities of Data Collected

	Normal Block	Orphan Block	Stale Block
Detector 1	13,348	726	131
Detector 2	13,348	754	142
Detector 3	13,348	731	138
Detector 4	13,348	718	140
Detector 5	13,348	734	135

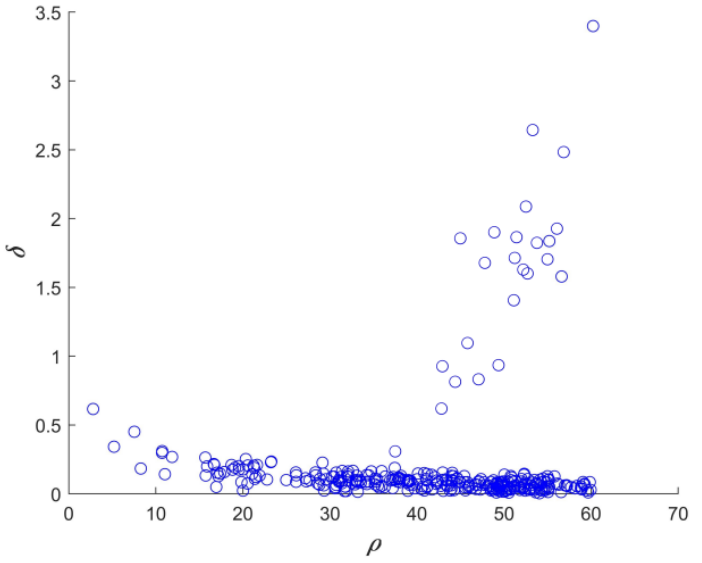
Data cleaning serves as the initial step in multi-node data fusion. The process includes addressing missing data, addressing anomalies, and handling outliers:

1. **Addressing Missing Data:** In the case of non-temporal missing data, completion is performed to address non-temporal missing data using identical data collected from various detection nodes. On the other hand, temporal data is treated as an anomaly during the data cleaning process.
2. **Addressing Anomalies:** Regarding temporal anomalies in the data, this study employs the method proposed by Bowden et al. (2020) for their treatment, and involves uniformly resampling the data between adjacent reliable timestamps.
3. **Handling Outliers:** To mitigate the potential influence of noise, this study employs a method that calculates Cook's distance for the block-related data. Outliers, defined as those exceeding four times the average Cook's distance, are excluded from the analysis. This process aids in identifying and flagging data points that exhibit significant errors, such as erroneous timestamps.

After going through the data cleaning process, the resulting data is subjected to multi-node data fusion, yielding the final data on blockchain formation. Figure 11 illustrates an example of the data fusion process. The specific process of multi-node data fusion is outlined as follows:

1. **Individual Node Blockchain Formation:** Each detection node's blockchain formation process is reconstructed by considering the data collected and recorded by that specific node. The specific approach entailed constructing a comprehensive block synchronization process for an individual detection node by assembling the collected data in chronological order, following the synchronization procedure of the blockchain (Chen et al., 2020).
2. **Multi-Node Blockchain Formation:** The blockchain formation processes from multiple nodes are combined and integrated. This step involves harmonizing the data and information obtained from different nodes, aligning the block height of the blockchain and ensuring consistency across the multiple blockchain formations.

Figure 11. An Example of Multi-Node Data Fusion



Finally, we validated the feasibility of our method through a controlled experiment. The "Default" represents the results obtained from the probing node without any modifications to the network topology. The DPC algorithm utilized the method proposed in this study. Additionally, we conducted experiments under the same conditions with two other clustering algorithms, DBSCAN and K-means. Detailed data are shown in Table 5.

The experimental results demonstrate the advantages of our method. Our method yielded an additional 8.3% of data collected compared to the Default method, generating a more comprehensive construction of the blockchain formation process. Furthermore, due to the linkage with cluster centers, the time difference in data collection for DPC was 37% less than that of the Default method.

In comparison to DBSCAN and K-means, DPC achieved the ability to collect more data and reduce the time difference using fewer cluster centers. This implies that we can minimize modifications to the network topology and achieve better performance. Our analysis shows the K-means algorithm is better suited for handling spherical clusters, while the DBSCAN algorithm assumes clusters as dense regions. However, the real Bitcoin network is a decentralized network, and the shape and density of clusters formed in the network cannot be determined. Moreover, the K-means algorithm requires prior specification of the number of cluster centers (in this experiment, k=30), while the DBSCAN algorithm requires specifying density thresholds and neighborhood radius (in this experiment, MinPt=20, eps=2). These preset parameters are difficult to determine in the Bitcoin network and can also affect the quality of clustering.

Geolocating Bitcoin Nodes

Based on the data obtained by traversing the Bitcoin network topology using BitcoinNode, the approximate location map of all nodes in the Bitcoin network is shown in Figure 12. In the map, a circle represents a set of neighboring mining nodes in terms of geographical location. Additionally, 57% of the nodes use either onion IP addresses or private IP addresses, which we refer to as N/A nodes. All N/A nodes are marked in the bottom left corner of the map. It is worth noting that N/A nodes engage in active communication within the Bitcoin network although their addresses cannot be determined.

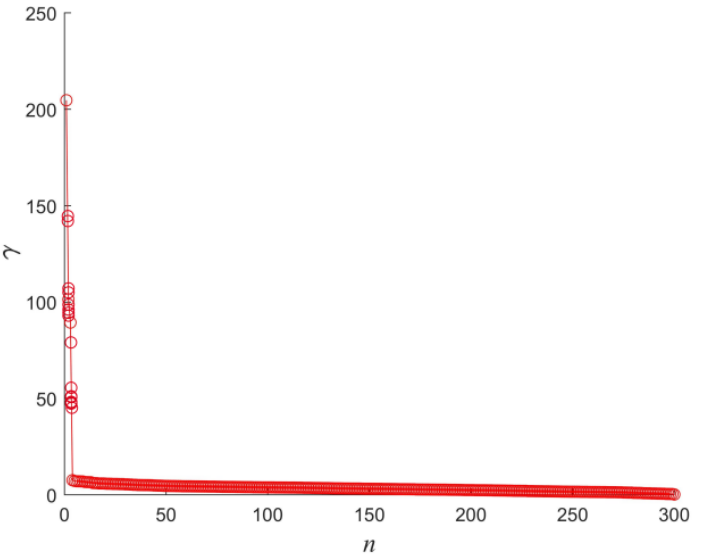
A detection node was deployed in each of the three continents that showed the highest number of nodes in the Bitcoin network, namely, the Americas, Europe, and Asia. This deployment allows for data collection from the densely populated nodes in the Bitcoin network, reducing the impact of network fluctuations. Furthermore, it enables the analysis of block propagation delays among the three nodes, which helps determine the location of malicious nodes. The results from Boscovic et al. (2018) demonstrate a positive correlation between block size and propagation delay, with similar propagation times for the same block among different nodes. In the current Bitcoin network environment, the median time for a block to propagate throughout the entire network is as follows.

$$T = 2 + 0.08 \times S \tag{7}$$

Table 5. Comparison With Default, DBSCAN, and K-means

	Cluster Centers	Normal Block	Orphan Block	Stale Block	Time Difference
Default	0	12,320	702	110	27s
DPC	20	13,348	754	143	10s
DBSCAN	30	13,348	731	140	13s
K-means	35	13,348	719	129	20s

Figure 12. Location Map



Given the similar propagation time of the same block among different nodes in the Bitcoin network, we denote the propagation time as T_B. The time at which the detection node receives a block as T_D can be expressed as the sum of the time at which the previous node releases the block as T_M and the propagation time of the block in the network.

$$T_D = T_M + h \times T_B \tag{8}$$

The symbol h represents the number of steps for block propagation, with each step signifying a block synchronization between every two nodes. By analyzing the time at which the three detection nodes received a block, we can calculate the steps h_1, h_2, and h_3 between the mining node that released the block and each of the three detection nodes.

$$\begin{cases} h_1 \le \dfrac{T_D - T_M}{T_B} \\ h_2 \le \dfrac{T_D' - T_M}{T_B} \\ h_3 \le \dfrac{T_D'' - T_M}{T_B} \end{cases} \tag{9}$$

Based on the topology structure of the Bitcoin network, we finally obtain sets of nodes N_1, N_2, and N_3, with steps h_1, h_2, and h_3 between the nodes and each of the three detection nodes respectively. From the intersection of these three sets, we derive the target node set NT, which consists of the mining nodes we are seeking.

$$N_T = \cap\left\{N_1, N_2, N_3\right\} \tag{10}$$

CONCLUSION

This paper introduces BitTrace, a framework for collecting blockchain formation data in the Bitcoin network. BitTrace enables the collection of fine-grained and real-time data for research areas such as evaluating the state and performance of the Bitcoin network, detecting malicious miner nodes, and adjusting mining strategies. The paper provides a detailed description of the design principles, system architecture, and underlying principles of BitTrace. To improve data completeness and efficiency, we propose a high-quality node identification method based on the DPC algorithm, which outperforms other clustering algorithms. The research findings demonstrate that increasing the number of monitoring nodes and connecting them to higher-quality nodes can enhance data collection efficiency. Finally, a geolocation plug-in for identifying malicious miner nodes is proposed based on the design of BitTrace.

BitTrace, serving as a data collection framework for Bitcoin system, effectively addresses the existing research gap in this domain. The comprehensive framework design and detailed description of the blockchain formation process offer valuable insights and serve as a reference for future researchers in this field. Furthermore, the versatile BitTrace allows for application across domains and scenarios, further enhancing its potential impact and practicality. Significant progress has been made in optimizing Bitcoin mining strategies (Luo & Zhang, 2023) and detecting selfish miners with BitTrace. However, it is worth noting that BitTrace still has some limitations. For example, BitTrace's scalability advantages are weakened by the limited availability of plug-ins. In the future, our work can be extended in several ways:

- Development of more plug-in modules. Our microkernel-based framework is highly scalable and enables more functional plug-ins that implement other business logic. For example, plug-ins that analyse the running status of different clients in the Bitcoin network, and plug-ins that visually analyse the blockchain formation process.
- Framework performance optimization. The framework may become more efficient and less consuming after collecting a large amount of real-time data. In future work, we will optimise data storage in the framework so as to speed up the plug-in data calls.

REFERENCES

Boscovic, D., Chawla, N., & Tapp, D. (2018). *Block propagation applied to Nakamoto networks*. Academic Press.

Bowden, R., Keeler, H. P., Krzesinski, A. E., & Taylor, P. G. (2020). Modeling and analysis of block arrival times in the bitcoin blockchain. *Stochastic Models*, *36*(4), 602–637. doi:10.1080/15326349.2020.1786404

Catalini, C., & Gans, J. S. (2016). Some simple economics of the blockchain. *Communications of the ACM*, *63*(7), 80–90. doi:10.1145/3359552

Chen, T., Li, Z., Zhu, Y., Chen, J., Luo, X., Lui, J. C.-S., Lin, X., & Zhang, X. (2020a). Understanding Ethereum via graph analysis. *ACM Transactions on Internet Technology, 20*(2), 18, 1-18.

Chen, T., Zhu, Y., Li, Z., Chen, J., Li, X., Luo, X., Lin, X., & Zhang, X. (2020b). Understanding Ethereum via graph analysis. *ACM Transactions on Internet Technology*, *20*(2), 1–32. doi:10.1145/3381036

Chen, Y., Chen, H., Zhang, Y., Han, M., Siddula, M., & Cai, Z. (2022). A survey on blockchain systems: Attacks, defenses, and privacy preservation. *High-Confidence Computing*, *2*(2), 100048. doi:10.1016/j.hcc.2021.100048

Franzoni, F., Salleras, X., & Daza, V. (2022). AToM: Active topology monitoring for the bitcoin peer-to-peer network. *Peer-to-Peer Networking and Applications*, *15*(1), 408–425. doi:10.1007/s12083-021-01201-7

Grossman, S., Abraham, I., Golan-Gueta, G., Michalevsky, Y., Rinetzky, N., Sagiv, S., & Zohar, Y. (2017). Online detection of effectively callback free objects with applications to smart contracts. *Proceedings of the ACM on Programming Languages*, *2*(POPL), 1–28. doi:10.1145/3158136

Guo, H., & Yu, X. (2022). A survey on blockchain technology and its security. *Blockchain: Research and Applications*, *3*(2), 100067.

Guo, S. T., Wang, R. J., & Zhang, F. L. (2021). Survey on the principles and applications of blockchain technology. *Computer Software and Computer Application, 48*(2).

Harry, K., Malte, M., Kevin, L., Steven, G., Martin, P., Alishah, C., & Arvind, N. (2020). BlockSci: Design and applications of a blockchain analysis platform. *29th USENIX Security Symposium (USENIX Security 20)*, 2721–2738.

Liu, Y., Liu, J., Salles, M. A. V., Zhang, Z., Li, T., Hu, B., & Lu, R. et al. (2022). Building blocks of sharding blockchain systems: Concepts, approaches, and open problems. *Computer Science Review*, *46*, 100513. doi:10.1016/j.cosrev.2022.100513

Luo, Y., & Zhang, J. (2023). An optimised bitcoin mining strategy: Stale block determination based on real-time data mining and XGboost. *International Journal of Information Technologies and Systems Approach*, *16*(2), 1–19. doi:10.4018/IJITSA.318655

Martin, W., Friedhelm, V., & Axel, K. (2020). Tracing manufacturing processes using blockchain-based token compositions. *Digital Communications and Networks*, *6*(2), 167–176. doi:10.1016/j.dcan.2019.01.007

Mohanta, B. K., Jena, D., Ramasubbareddy, S., Daneshmand, M., & Gandomi, A. H. (2020). Addressing security and privacy issues of IoT using blockchain technology. *IEEE Internet of Things Journal*, *8*(2), 881–888. doi:10.1109/JIOT.2020.3008906

Motlagh, S. G., Misic, J. V., & Mišić, V. B. (2021). The impact of selfish mining on bitcoin network performance. *IEEE Transactions on Network Science and Engineering*, *8*(1), 724–735. doi:10.1109/TNSE.2021.3050034

Rodriguez, A., & Laio, A. (2014). Clustering by fast search and find of density peaks. *Science*, *344*(6191), 1492–1496. doi:10.1126/science.1242072 PMID:24970081

Sarode, R. P., Singh, D. G., Watanobe, Y., & Bhalla, S. (2023). High-volume transaction processing in bitcoin lightning network on blockchains. *International Journal on Computer Science and Engineering*, *26*(4), 445–458.

Shahsavari, Y., Zhang, K., & Talhi, C. (2020). A theoretical model for block propagation analysis in bitcoin network. *IEEE Transactions on Engineering Management*, *69*(4), 1459–1476. doi:10.1109/TEM.2020.2989170

Taherdoost, H. (2023). Smart contracts in blockchain technology: A critical review. *Information (Basel)*, *14*(2), 117. doi:10.3390/info14020117

Tri, S., Ideva, G., & Siska, A. (2017). Study on blockchain visualization. *JOIV: International Journal on Informatics Visualization*, *1*(3), 76–82. doi:10.30630/joiv.1.3.23

Wang, Y., Wang, Z., Zhao, M., Han, X., Zhou, H., Wang, X., & Koe, A. S. V. (2022). BSM-ether: Bribery selfish mining in blockchain-based healthcare systems. *Information Sciences*, *601*, 1–17. doi:10.1016/j.ins.2022.04.008

Wang, Y. Y., & Li, G. Q. (2019). Detect triangle attack on blockchain by trace analysis. *2019 IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, 316–321. doi:10.1109/QRS-C.2019.00066

Wong, D. R., Bhattacharya, S., & Butte, A. J. (2019). Prototype of running clinical trials in an untrustworthy environment using blockchain. *Nature Communications*, *10*(1), 1–8. doi:10.1038/s41467-019-08874-y PMID:30796226

Wu, C., Yan, B., Yu, R., Huang, Z., Yu, B., Yu, Y., Chen, N., & Zhou, X. (2021). Improvement of K-Means algorithm for accelerated big data clustering. *International Journal of Information Technologies and Systems Approach*, *14*(2), 99–119. doi:10.4018/IJITSA.2021070107

Wu, D., Liu, X., Yan, X., Peng, R., & Li, G. (2019). Equilibrium analysis of bitcoin block withholding attack: A generalized model. *Reliability Engineering & System Safety*, *185*, 318–328. doi:10.1016/j.ress.2018.12.026

Xu, H., Zhang, L., Onireti, O., Fang, Y., Buchanan, W. J., & Imran, M. A. (2020). BeepTrace: Blockchain-enabled privacy-preserving contact tracing for COVID-19 pandemic and beyond. *IEEE Internet of Things Journal*, *8*(5), 3915–3929. doi:10.1109/JIOT.2020.3025953 PMID:37974935

Yaga, D., Mell, P., Roby, N., & Scarfone, K. (2018). Blockchain technology overview. *ArXiv, abs/1906.11078*.

Yahya, S., Kaiwen, Z., & Chamseddine, T. (2022). A theoretical model for block propagation analysis in bitcoin network. *IEEE Transactions on Engineering Management*, *69*(4), 1459–1476. doi:10.1109/TEM.2020.2989170

Yang, G., Wang, Y., Wang, Z., Tian, Y., Yu, X., & Li, S. (2020). IPBSM: An optimal bribery selfish mining in the presence of intelligent and pure attackers. *International Journal of Intelligent Systems*, *35*(11), 1735–1748. doi:10.1002/int.22270

Zheng, P., Xu, Q., Luo, X., Zheng, Z., Zheng, W., Chen, X., Zhou, Z., Yan, Y., & Zhang, H. (2022). Aeolus: Distributed execution of permissioned blockchain transactions via state sharding. *IEEE Transactions on Industrial Informatics*, *18*(12), 9227–9238. doi:10.1109/TII.2022.3164433

Jian Wu, Master's Degree, graduated from South China Normal University in 2017. His research interests include Blockchain System Development.

Jianhui Zhang, Software Engineer, Master's Degree, graduated from Southwest University in 2022. Worked in Beijing ByteDance Co., Ltd. His research interests include Blockchain System Development.