International Journal of Information Technologies and Systems Approach
Volume 16 • Issue 2

An Optimised Bitcoin Mining Strategy:
Stale Block Determination Based on Real-Time Data Mining and XGboost

Yizhi Luo, Southwest University, China*
Jianhui Zhang, Southwest University, China

ABSTRACT

Stale blocks are not avoidable in blockchain, such as the Bitcoin network, when proof-of-work is used as the consensus protocol. However, as the economic loss to the miners and the security risk to the network cannot be ignored, research is needed to identify and analyse stale blocks. By analysing the factors influencing the generation of stale blocks, the authors propose a new machine learning model based on XGBoost. They propose a new data collection method for bitcoin nodes to obtain real data for training prediction model. Then, based on the model, they generate optimal mining strategies and analyse the economic benefits. The experimental data and application cases show that the real-time data detection and machine learning model that they propose can accurately identify and predict the generation of stale blocks and generate an economically optimal mining strategy in the Bitcoin network with the presence of stale blocks.

KEYWORDS

Mining Strategy, Real-Time Data Mining, Stale Blocks, XGBoost

INTRODUCTION

The emergence of digital currencies has substantial financial markets. For example, the market value of Bitcoin (Nakamoto, 2008), the first digital cryptocurrency, has now reached trillions of dollars (Böhme et al., 2015). As the core technology of digital cryptocurrencies, blockchain has gained much attention, and there are now many attempted implementations in a wide area of applications (Long et al., 2021; Tang & Zeng, 2021). However, all decentralized blockchain systems face the problem of ownership of bookkeeping rights.

For example, the Bitcoin system uses the proof-of-work (POW) mechanism to encourage miners to perform numerous calculations to compete for bookkeeping rights. The consensus on witnessing the transactions is based on the time spent calculating the results. Thus, competing for bookkeeping rights essentially turns out to be a zero-sum game among miners. In such a game, unfortunately, in competition, honest miners will inevitably produce blocks that are the same height as the optimal blocks but slightly later. These blocks become stale blocks.

DOI: 10.4018/IJITSA.318655 *Corresponding Author

In the Bitcoin network, stale blocks are successfully mined by miners but ultimately fail to be retained on the mainchain because of forks in the blockchain caused by peer nodes competing for transaction bookkeeping rights. In other words, in the mining and competing process, a stale block has the same height as the mainchain block, and a miner produces it after performing hash operations of the same complexity but ultimately fails to be added to the mainchain. As a result, the miner that generates the stale block suffers a great financial loss. Furthermore, the transactions verified with packaged stale blocks must also be repeatedly verified by other nodes. This verification process results in a substantial waste of resources and affects the fairness of mining. When a miner receives a stale block, if it chooses the branch where the stale block is located as the basis for mining, it is likely to suffer a substantial loss in mining revenue because of the cropping of the branch.

Although the generation of stale blocks causes economic losses to miners and affects the network performance, it has been ignored by most researchers, possibly due to the small probability of stale block generation. To the best of our knowledge, there is a lack of research on the prediction of stale block generation.

In this paper, we study the impact of stale blocks on the Bitcoin network. To do this, this paper investigates the operation, data, and state changes of full Bitcoin nodes from real-time data and studies how to provide honest miners with strategies for reducing the impact of stale blocks on revenue. Our main contribution is to provide a machine learning model based on real-time data for stale block generation and prediction, analyze the effects of stale blocks, and generate an optimal mining strategy for honest miners to improve profits when a fork occurs.

Related Work

Analysis and Definition of the Stale Block. Stale blocks are introduced in Bonneau et al. (2015), where stale blocks are described as blocks with valid transactions and complete transaction proofs but are not included in the mainchain, without analyzing the specific reasons for the emergence of stale blocks. The work of Feng and Niu (2019) analyzes the impact of stale blocks and uncle-order blocks generated with selfish mining in Ethereum. Boscovic et al. (2022) presented simulations of the probability of orphan blocks and stale blocks with different block sizes by constructing the NS3 blockchain simulator. The simulations show that block size is an important factor affecting the rate of stale blocks, but it was not verified by real data validation.

Data Collection and Node Instrumentation. Most works in the literature and projects related to the instrumentation of blockchain have been conducted for Ethereum smart contracts using various data of smart contracts. Grossman et al. (2017) investigated how information about internal transactions and storage operations is obtained by instrumentation to detect object callbacks in smart contracts and analyze the related security risks (e.g., DAO attacks). This work is limited to only the Ethernet smart contract environment and is not scalable. In Chen et al. (2020), a similar approach was proposed for Ethernet to collect and analyze data about transactions, smart contract creation and invocation, the transaction log of nodes, and some malicious accounts and malicious contract invocations. Based on this approach, further work by the same authors (Chen et al., 2019) presents a tool for automatically detecting inconsistent behavior of cryptocurrency tokens in Ethernet. There is little work on full bitcoin node instrumentation to collect real-time data about stale block generation.

Mining strategy. Mining strategies have been heavily studied, mostly on mining strategies related to malicious attacks such as selfish mining. Göbel et al. (2016) studies selfish mining strategies in the presence of propagation delays and present an approach for modeling and analyzing how to gain more revenue through selfish mining in the presence of propagation delays. There, they detected selfish mining behaviour using the generation rate of stale blocks. However, there is little research on how to apply machine learning techniques to analyze mining schemes. For example, Wang et al. (2021) show how a Markov chain model is constructed for mining and analyzing more efficient mining schemes by machine learning methods of reinforced learning (RL). However, the environment for strategy

selection in the work is unrealistic as there is no real-time data from real networks used in the model. Therefore, the model cannot be used to solve problems such as stale blocks or double-spend attacks.

This paper is the first study to model and generate predictions for the stale block generation process compared to other literature. Meanwhile, this paper proposes a new method for collecting data from the Bitcoin blockchain. Finally, this paper is the first to propose a better mining strategy to reduce the impact of stale blocks on miner nodes.

Contribution and Organization

As stated earlier, the problem of stale blocks and their impact on miners' revenue in the Bitcoin network has been largely ignored. This paper proposes to fill this gap by presenting a method for using machine learning techniques for stale block generation identification and prediction and an approach to improve mining strategy.

The model for stale block generation and identification is presented next. This paper parses the code of the full Bitcoin node to extract the operations and calls for an abstract presentation of the standard steps of block synchronization. This allows us to establish a behavioral model of block synchronization performed by the full nodes. With this model, this paper analyzes the synchronization process and determines the conditions for identifying stale blocks.

This paper then implements a collection method by instrumenting the source code of the full node to create detection nodes that can output more detailed data. By deploying the detection nodes into the Bitcoin network, this paper implements a real-time monitor to monitor the block synchronization process. Then, this paper implements a data parser to parse real-time data into analyzable data. Based on the analysis of the monitoring data, this paper uses the monitoring data as the input to the data parser for analysis of the full life-cycle of the blocks received by the full nodes. The output of the data parser is the block type and the state data of the node when it accepts the block.

The rest of this article is organized as follows. We next present the work on the prediction model. This paper measures the likelihood of a block being stale based on the parser's output using four parameters: network difficulty, decentralization degree, network congestion degree, and aggregation effect. We then construct and tune the stale block prediction model by the XGBoost method. For the mining strategy in the presence of stale blocks, this paper then compares the profits of different mining strategies under the influence of stale blocks. Finally, this paper provides a prediction-based mining strategy for honest miners to hedge the risk of choosing stale blocks as the basis for mining, which can effectively increase profits. We then present application case studies and experiments to validate our models and implementations, and finally provide the discussions, conclusions, and future work.

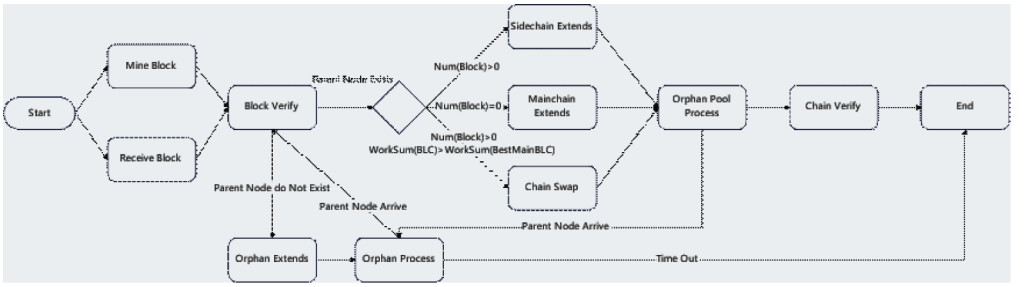
FUNDAMENTAL CONCEPTS

In the Bitcoin network, miners mine transactions and form them into a block and are willing for them to be added to the blockchain. Such a block comprises two parts. They are the block header and the block body. The block header encapsulates information, including the current block hash, current difficulty target (Bits), and a random number (Nonce) of the current block POW process. The block body contains the transactions of the current block and the Merkel tree for verifying the transactions. For a miner's block to be added to the blockchain, the miner must broadcast it to the network. All full nodes must comply with the consensus protocol to determine which block should be appended to the blockchain. This processor is called block synchronization.

Block Synchronization

This paper illustrates the complete process of block synchronization in Figure 1, which contains six steps: block acquisition, block verification, block type confirmation, block synchronization, orphan pool processing, and chain verification. After the full node obtains a new block, it generates

Figure 1. Bitcoin Block Synchronization Flow Chart



a BlockNode that represents the current block based on the block header data. After generating the BlockNode, the full node needs to check whether the block is standard. Then, the full node determines whether the block is an orphan block, a mainchain or a sidechain block and processes it according to its type, workload, and height. Finally, full nodes perform primary chain extension, side chain extension, or chain swap operations.

A block can become stale when more than one valid block with the same height from different miners is broadcast and verified during the block synchronization of the network. Our work in this paper analyzes the block synchronization process and devises full node instrumentation to obtain the data about the communication needed for predicting and identifying stale blocks.

Generation of the Stale Block

In the Bitcoin network, full nodes generate stale blocks in two cases. In the first case, the sidechain blocks become stale blocks. Upon the advent of a certain fork, the mainchain experiences a greater burden than any corresponding sidechains. In the event that the main chain exceeds the six blocks of the side chain, the side chain is no longer capable of replacing the main chain and all sidechain blocks become stale blocks (Figure 2).

In the other case, the original mainchain blocks become stale blocks. After the emergence of the fork and a period of competition, the workload of the sidechain becomes greater than that of the mainchain, causing chain swap. In this case, if the new mainchain (original sidechain) extends beyond the original mainchain block by six blocks in the subsequent extension, those blocks in the original mainchain that are replaced become stale. The chain swap is shown in Figure 3.

Judgement of the Stale Block

The blockchain is often described as a chain formed by connecting blocks containing valid transactions in chronological order. In fact, the blockchain stored in the full node is a tree structure. There can be multiple branches of chains at each tree height, while only one branch can go back from the optimal block to the genesis block. A branch traced back to the genesis block is known

Figure 2. Stale Block Generation Diagram

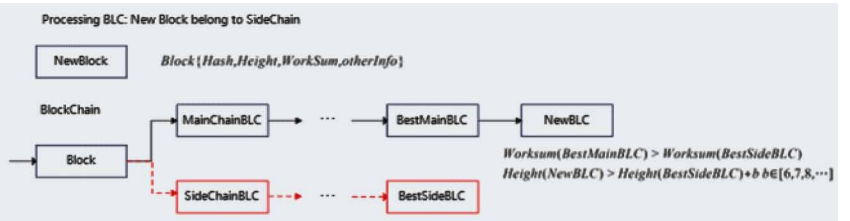
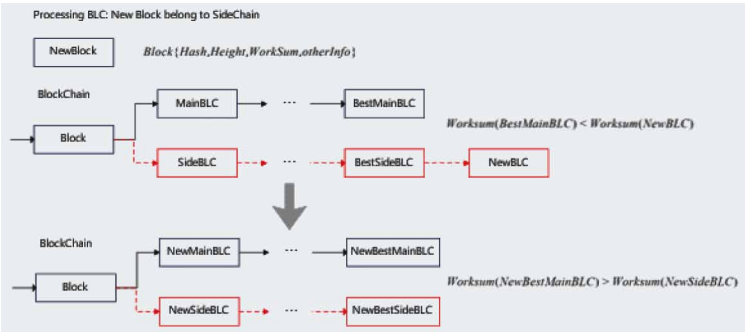


Figure 3. Second Stale Block Generation Diagram



as the blockchain mainchain, while the other branches are known as blockchain sidechains. As mentioned earlier, stale blocks used to be the main or sidechain blocks for a while but eventually failed to connect to the mainchain.

Analysis of the code in this paper shows that the sufficient judgment condition for the stale block is as follows: 1. The block is not a mainchain block. 2. The total workload of the side chain on which the block is located is less than that of main chain 3. The height of the optimal block of the sidechain where the block is located is less than six blocks from the mainchain block.

Detection Node and Data Parser

As mentioned earlier, this paper implements a detection node to collect data related to stale blocks and a parser to parse the real-time data into analyzable data. This paper chooses full Bitcoin nodes (BTCD, 2022) for instrumentation to generate detection nodes. The choice is because the full Bitcoin node is a complete Bitcoin client that can access and maintain the complete blockchain data. The specific instrumentation plan for generating detection nodes is as follows. First, we divide the block synchronization behaviour into its constituent behaviours. Then, this paper parses out the full node operation corresponding to the behavior from the code, and the operations correspond to the function calls. Thus, this paper can realize one-to-one correspondence from the top-level behavior to the segmentation behavior and then to the bottom-level specific function calls. After that, this paper outputs the data in the corresponding function call. For example, in Figure 4, the chain swap operation in the block synchronization behavior of the full node requires three essential function calls: Findfork, getReorganiseNode, and getReorganiseChain. Finally, this paper implements a data analyzer to convert real-time data into analyzable data by string matching to real-time log files.

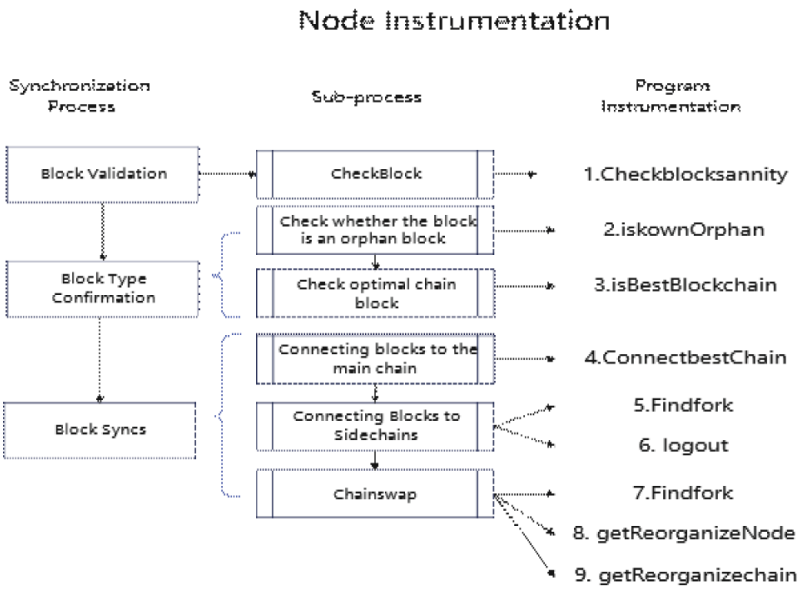
METHODOLOGY

In this section, this paper analyzes the causes that lead to the generation of stale blocks, identifies the factors that influence their generation, and establishes a prediction model to predict the probability of a block becoming stale based on real-time detection data.

What Affects Stale Block Generation?

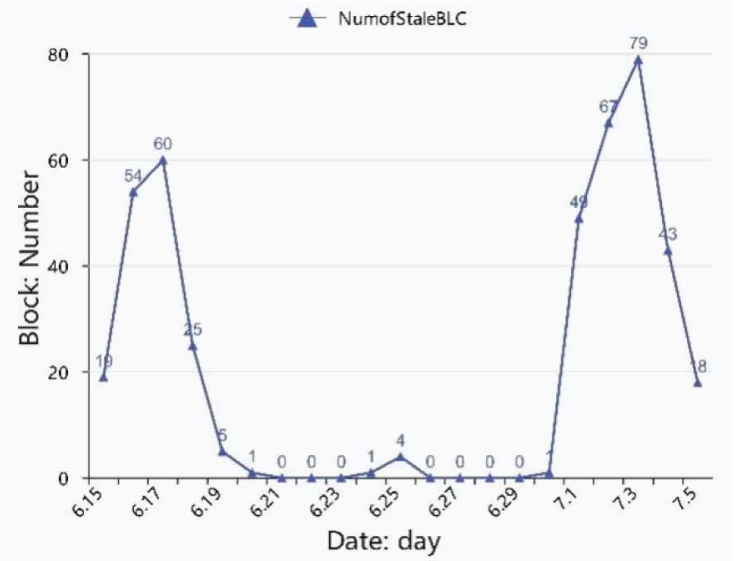
The generation of stale blocks in the Bitcoin network is divided into four main stages: the generation of blocks by miners, the propagation of blocks by the network, the reception and processing of blocks by the full nodes, and the blocks becoming stale. Some factors affect the generation of stale blocks in each of these four stages.

Figure 4. Full Bitcoin Node Instrumentation Diagram



1. **Network difficulty and decentralization degree:** In the block generation process undertaken by miners, two primary factors affect the generation of stale blocks: network difficulty and decentralization degree. Network difficulty refers to the hash difficulty limit that needs to be satisfied when mining by a miner, which directly affects the speed of mining blocks by miners and the speed of reaching consensus in the network. This paper uses the dynamic difficulty in the Bitcoin code (BTCD, 2022) as a measure, which is the average of the difficulty of the most recent 2016 blocks.
2. **Degree of decentralization of the Bitcoin network:** The more decentralized the network is, the greater the theoretical difficulty of reaching consensus across the network and the higher the probability of generating stale blocks. Coinbase CTO Srinivasan (Srinivasan & Lee, 2017) proposed a parameter to measure the degree of decentralization of the blockchain—the Satoshi–Nakamoto coefficient. The principle is to calculate the Gini index of each of the six subsystems that make up the blockchain (such as mining revenue, node client version, or developer participation) as a parameter to measure the degree of decentralization. This paper adopts the Gini index of mining revenue and the Gini index of communication between neighbouring peer nodes to measure the network's degree of decentralization.
3. **The network congestion level:** In the process of new block propagation, the network congestion level is an important factor affecting the generation of stale blocks, which determines whether the blocks can spread to the whole network quickly and reach the correct consensus. Many factors affect the degree of network congestion, such as the block size, the number of nodes in the network, and the version share of nodes in the network. The larger the network congestion is, the longer the network takes to form consensus and the easier the generation of stale blocks (Vujičić et al., 2018).
4. **Aggregation effect:** By performing a review of data concerning block synchronization collected from Bitcoin nodes, it is possible to discern the production of stale and orphan blocks which in turn has an aggregation effect (Figure 5). When the network produces these stale blocks, miners partake in mining different best-received blocks, consequently resulting in diluted computing power across the system and escalating the time taken to reach consensus. This paper uses the number of stale blocks and orphan blocks in the first hundred blocks of a new block as the aggregation effect parameter.

Figure 5. Aggregation Effect Diagram of Stale Blocks



Analysis of Influencing Factors

This paper proposes measuring the probability of a block becoming stale by the following four variables: the network difficulty and decentralization during block generation, the degree of network congestion, and the aggregation effect parameters during propagation.

1. **Network difficulty parameter:** This paper adopts the standard difficulty of a block as the network difficulty parameter to measure the dynamic difficulty of a block when it comes out of a block, where n refers to the block of height n, T_n refers to the time difference between the moment of the previous block coming out of a block of height n and the moment of the block coming out, Dif_n refers to the network difficulty when the block comes out of a block, T_{Ave} refers to the average duration of the block coming out of a block, and $\sum_{n-2016}^{n} T_n$ refers to the duration used for the first 2016 blocks out of the block, given by:

$$Dif_{Std} = Dif_n \times \frac{T_{Ave} \times 2016}{\sum_{n-2016}^{n} T_n} \tag{1}$$

2. **Network congestion degree parameter:** This paper uses the number of messages required to obtain a new block as the indicator of the congestion degree of the network, where T_{IN} refers to the difference between the block-out time and the connection time (s), calculated as the block-up time minus the block-out time expressed by the block-out timestamp. T_M refers to the split-average message reception of the full node, whose expression is as follows:

$$Congestion = \frac{T_{In}}{60} \times T_M \tag{2}$$

3. **Network decentralization parameters:** This paper uses the mining revenue Gini index (ChainGini) and the message Gini index (InfGini) to measure the degree of decentralization of the network at synchronization. The message Gini index is also used to measure whether the information received by the probing nodes may be subject to information attacks against a single node and whether it can be used to effectively measure the network. This paper uses $Miner_k$ to represent the number of blocks mined by a special miner's address in the last 100 blocks. This paper uses $node_k$ to represent the number of network messages received by the full node from a particular IP address in the last 10 minutes:

$$ChainGini = 1 - \left(\sum_1^K \left(\frac{Miner_k}{100} \right)_k^2 \right) \tag{3}$$

$$InfGini = 1 - \left(\sum_1^K \left(\frac{Node_k}{Congestion} \right)_k^2 \right) \tag{4}$$

4. **Aggregation effect parameter:** It measures the aggregation effect of stale and orphan block generation in the network. Its parameter is the number of stale and orphan blocks in the last 100 blocks:

$$Cluster = StaleN + OrphanN \tag{5}$$

Prediction Model

This paper uses the XGBoost model (Chen & Guestrin, 2020; Xu et al., 2017) as the prediction model for stale blocks. The objective function of the XGBoost algorithm is shown below:

$$Obj = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i\right) + \sum_{k=1}^{k} \Omega\left(f_k\right) \tag{6}$$

The essence of its training is the process of gradually reducing the overall error by iteratively generating more classification trees to fit the classification error based on the predictions of the previously trained classification trees. Compared to the Adaboost and GBDT models, the XGBoost model adds a regular term to the loss function to control the complexity of the model. It can make the learned model simpler and prevent overfitting. The XGBoost model supports the user in defining the objective function and the loss function, which is more flexible. Compared to the random forest model, the XGBoost model can achieve better training results by adjusting the model training parameters. The XGBoost model also supports parallel processing, which allows for faster training of the model.

In the general XGBoost model, the loss function used for logistic regression is the following logistic loss:

$$l\left(y_i, \hat{y}_i\right) = y_i \ln\left(1 + e^{-\hat{y}_i}\right) + \left(1 - \hat{y}_i\right) \ln\left(1 + e^{\hat{y}_i}\right) \tag{7}$$

As seen in the previous section on data processing, in the process of stale block generation and prediction, the number of positive samples (stale blocks) to be predicted is much smaller than the number of negative samples (mainchain blocks), and there is an aggregation effect in the generation of stale blocks. Therefore, the difficulty of predicting a sample as a negative sample is much smaller than the difficulty of predicting it as a positive sample. As a result, this paper adopts the focal loss method for dense object detection (Lin et al., 2017; Wu et al., 2021) to adjust the loss function of XGBoost. The adjusted loss function is as follows. By adjusting the loss function, the XGBoost model in this paper can better distinguish more difficult positive samples (i.e., stale block samples):

$$l\left(y_i,\hat{y}_i\right)=-\alpha\left(1-p_t\right)^{\gamma}y_i\ln(1+e^{-\hat{y}_i})+\left(1-\alpha\right)p_t^{\gamma}\left(1-y_i\right)\ln(1+e^{\hat{y}_i}) \tag{8}$$

Many parameters of the XGBoost model may lead to overfitting the model during the training process. Therefore, this paper also tunes these parameters. We describe the specific tuning results in the experimental section.

Evaluation Indicators

Obviously, in practical application cases, the miner needs the model to accurately identify the stale blocks from all the received blocks and the model to accurately distinguish the mainchain blocks of the same height from the stale blocks and adjust the mining strategy accordingly. This paper uses accuracy and precision to determine whether the model can identify the stale blocks from all the received blocks. Additionally, accuracy is used to check whether the model can effectively distinguish between blocks of the same height and a similar number of blocks in the mainchain and stale blocks. Finally, accuracy indicates the number of correctly classified samples as a percentage of the total number of samples:

$$Accuracy=\frac{TP+TN}{FN+TN+TP+FP} \tag{9}$$

Precision indicates the ratio of the number of samples correctly identified by the model as positive classes to the total number of positive samples:

$$Precision=\frac{TP}{TP+FP} \tag{10}$$

MINING STRATEGY UNDER THE INFLUENCE OF STALE BLOCKS

The generation of stale blocks is inevitable in a decentralized Bitcoin network. If a miner chooses stale blocks as the basis for mining, there is a high probability that he blocks it mines will become stale due to chain swap. However, miners can reduce the impact of stale blocks on their revenue by adopting different mining strategies, thus increasing their revenue. In this section, this paper compares the gains from different mining strategies.

Current Mining Strategy and Revenue

Currently, the mining strategy of a miner is to receive the first block of a certain height as the optimal block and mine based on it. The miner's revenue depends on whether the first block it receives is a

stale block (i.e., whether it is mining on a chain with a high probability of cropped). The total revenue expectation, when affected by stale blocks, is as follows, where p_{mb} refers to the probability that the mainchain block arrives first, p_T refers to the probability that chain swap occurs, p_i refers to the probability that the miner will mine the block earlier than other miners, P_{Btc} refers to the price of a single bitcoin, N_{Btc} and N_{Pbtc} refer to the bitcoin reward and the transaction fee when the block is released, respectively, n refers to the number of hash operations performed by the miner to mine the current block, m refers to the total number of operations performed by other nodes in the trading network, Cal refers to the current miner's computing power, and Cal_A refers to the current total computing power. T refers to the time taken to mine the block:

$$E = p_{mb}\left(1-p_T\right)\left(p_i\left(N_{Btc}+N_{Pbtc}\right)P_{Btc}\right)+(1-p_{mb})p_T\left(p_i\left(N_{Btc}+N_{Pbtc}\right)P_{Btc}\right) \tag{11}$$

$$p_i = \begin{pmatrix}1\\T\end{pmatrix}\left(1-p_n\right)^{T-1}p_n\left(1-p_m\right)^T \tag{12}$$

$$n = T\times Cal, p_n = \frac{n}{Dif\times 2^{32}}, m = T\times Cal_A, p_m = \frac{m}{Dif\times 2^{32}} \tag{13}$$

New Mining Strategies

This paper proposes two new mining strategies based on stale block detection, and miners can choose the strategy for mining according to the situation. The optimized mining strategy is when the miner detects that the probability of the received block being stale is greater than a threshold value based on the model. The miner avoids the stale block and selects a mainchain block as a mining base. By mining through this strategy, the total revenue expectation function of the miner when affected by stale blocks is as follows, where A_c refers to the accuracy of the model in determining the stale blocks and T_M represents the arrival time difference between the mainchain blocks arriving at the node earliest at a certain height and the arrival time of the blocks arriving at the highest priority:

$$E = A_c\left(1-p_T\right)\left(p_i\left(N_{Btc}+N_{Pbtc}\right)P_{Btc}\right) \tag{14}$$

$$p_i = \begin{pmatrix}1\\T-T_M\end{pmatrix}\left(1-p_n\right)^{T-T_M-1}p_n\left(1-p_m\right)^T \tag{15}$$

$$n = \left(T-T_M\right)\times Cal, p_n = \frac{n}{Dif\times 2^{32}}, M = T\times Cal_A, p_m = \frac{m}{Dif\times 2^{32}} \tag{16}$$

The contrasting strategy involves selecting stale blocks for mining when the miner detects that the probability of the received blocks being stale blocks is greater than a threshold value based on the model. The total revenue expectation function of the miner, when affected by stale blocks, is as follows, where T_S refers to the arrival time difference between the earliest arriving stale block at a node at a certain height and the arrival time of the most preferred arriving block:

$$E = A_c p_T \left(p_i \left(N_{Btc} + N_{Pbtc} \right) P_{Btc} \right) \tag{17}$$

$$p_i = \begin{pmatrix} 1 \\ T - T_S \end{pmatrix} \left(1 - p_n\right)^{T - T_S - 1} p_n \left(1 - p_m\right)^T \tag{18}$$

$$n = \left(T - T_S\right) \times Cal, p_n = \frac{n}{Dif \times 2^{32}}, M = T \times Cal_A, p_m = \frac{m}{Dif \times 2^{32}} \tag{19}$$

Optimization Strategy vs. Attack Strategy

Our proposed strategy is an optimization of the honest mining strategy in which the method recommends that miners select mainchain blocks as the basis for transaction validation. This method reduces the probability that the blocks mined by itself become stale blocks but does not impact the blocks mined by other honest mining nodes and improves miners' gains under the influence of stale blocks. Moreover, the miners choose the mainchain blocks to mine, which also facilitates the nodes of the whole network in reaching consensus faster, saves the computing power resources of the entire network, and improves the efficiency of the transaction verification of the entire network. These features are essentially different from attack methods, such as selfish mining (Bag et al., 2016; Eyal & Sirer, 2018; Heilman et al., 2015), which exploits the concept of chain forking to craft maliciously generated sidechains thus leading to mainchain blocks created by honest nodes becoming stale and consequently hindering transaction networks from achieving consensus and posing economic hardships upon honest nodes. Therefore, our optimization strategy is more suitable for most honest miners in the network who do not commit evil acts to obtain larger revenue with a small judgment cost.

EXPERIMENTAL RESULTS

This section describes the data collection results, preprocessing process, model tuning, experimental results, and application cases.

Data Collection

This paper implements a data collection and parsing tool. This paper deploys three detection nodes to collect block synchronization data of the Bitcoin Testnet for 20 days, including full node block synchronization data, chain transformation data, and internode communication data. The specific types and amounts of data are shown in Table 1.

By parsing the data collected by the parser, this paper can obtain the data about mainchain blocks, stale blocks and orphan blocks synchronised by the full node in 20 days, as well as the data about the network environment at the time of synchronising to the relevant blocks. According to the statistics, this paper collects 879 stale blocks, 3568 orphan blocks, and 41584 mainchain blocks in

Table 1. DataType Table

	BLC	INF	WRN	ERR
BLC INFO	250562	-	-	-
CHAN INFO	-	1966	-	-
SYNC INFO	-	208871	-	-
ERR INFO	-	-	213	453
Other INFO	503	214	127	42

20 days by three detection nodes (the nonrepetitive blocks are 426 stale blocks, 2327 orphan blocks, and 16838 mainchain blocks).

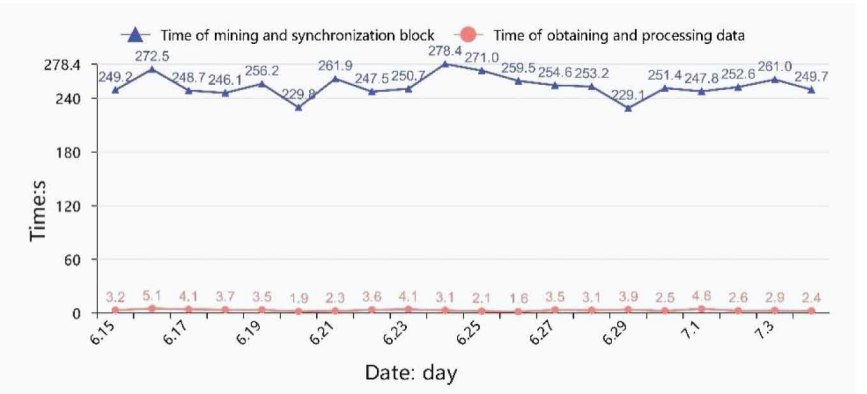
To verify real-time data collection, this paper counts the average time of block mining, the average time of block synchronization, the average time of data acquisition by the tool, and the average time of data processing by the tool within 20 days, as shown in Figure 6. The average time for block mining and synchronization in the network over a 20 days is approximately 253 seconds. The average time for the tool to obtain and process the data is approximately 3 seconds. Moreover, the block mining and synchronization time fluctuate between 10 and 20 seconds for a Bitcoin node, which is also greater than the processing time. In summary, the blockchain consensus in the network does not change significantly while the tool processes the data. Therefore, this paper concludes that the tool can meet the real-time requirements of data processing.

Data Preprocessing

After collecting the data of blocks, stale blocks, and orphan blocks within 20 days, the following data preprocessing work is performed on the data to facilitate data analysis and model building.

1. **Missing Data Completions:** This paper cross-references the relevant data collected by different detection nodes to complete the missing values (except for temporal data).
2. **Data resampling:** This paper adopts the method of undersampling mainchain blocks and oversampling stale blocks to balance the data volume.
3. **Exception handling:** In processing block data, this paper finds that there are problems with the timestamps of some blocks, and the generation timestamps of blocks are smaller than the block

Figure 6. Time Comparison



arrival timestamps. Thus, this paper addresses the block timestamps according to the method of dealing with abnormal timestamps mentioned by Bowden et al. (2020). The method re-samples unreliable timestamps evenly between adjacent reliable timestamps. Then, the processing makes the timestamp data normal again.

4. **Outlier processing:** This paper excludes outliers greater than four times the average cookie distance.
5. **Data standardization and normalization:** Finally, this paper standardized the data.

After data processing, this paper divides the data into two parts to train the model. The first part is 70% of the data randomly selected as the training set, and the second comprises the remaining data, which is used as a test set to measure the model's ability to distinguish normal blocks from stale blocks. Additionally, to measure whether the model can effectively distinguish between mainchain blocks and stale blocks of the same height and similar number, normal blocks and stale blocks of the same height (852 blocks in total) are distinguished as the second test set.

Modeling and Tuning

This paper customizes the loss function of the XGBoost model. It can have better prediction results when the number of positive and negative samples in the training differs significantly. As described in the article on focal loss for dense object detection (Lin et al., 2017; Wu et al., 2021), adjusting the α parameter allows the model to focus more on positive samples with smaller sample sizes. Adjusting the γ parameter allows the model to focus more on more difficult-to-judge samples. This paper tunes the α and γ parameters of the loss function with the same other training parameters. The paper also compares the strengths and weaknesses of the models with two different loss functions. We can conclude that the model predicts best when using the adjusted loss function with $\alpha = 0.8$ and $\gamma = 1$.

Then, this paper tunes the other parameters of the XGBoost model, as shown in Table 3. The optimal prediction model is obtained when the learning degree is 0.5, the number of iteration layers is 20, the maximum tree depth is 10, the maximum incremental step of tree nodes is 5, the minimum tree divergence loss value is 1, and the division threshold is 0 in Table 2. The model can distinguish the stale blocks from all the received blocks (the precision is 0.976, and the precision ratio is 83/85) and distinguish the stale blocks from the mainchain blocks with the same height and stale blocks (the accuracy is 0.991, and the accuracy ratio is 844/852).

This paper also compares the performance of the random forest, GBDT, and decision tree models with that of the XGBoost model when the same training and test sets are used. As shown in Figure 7, the XGBoost model has a higher prediction accuracy than the other models.

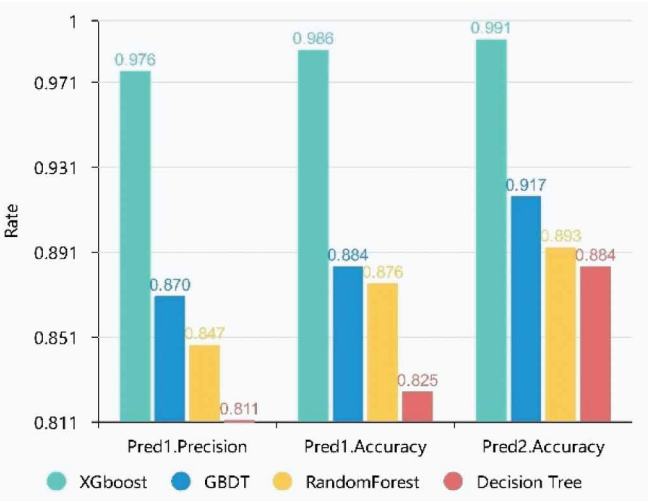
Table 2. Loss Function Parameter Tuning Table

Loss Function	α	γ	Pred1. Precision		Pred1. Accuracy		Pred2. Accuracy
			Test	Train	Test	Train	
Logit loss	-	-	0.905	0.998	0.820	0.978	0.957
New Loss	0.8	1	0.976	0.995	0.986	0.980	0.991
New Loss	0.8	0	0.917	0.985	0.978	0.982	0.957
New Loss	0.8	0.5	0.953	0.989	0.984	0.973	0.978
New Loss	0.8	2	0.987	0.998	0.953	0.994	0.984
New Loss	0.7	1	0.974	0.995	0.967	0.978	0.956
New Loss	0.6	1	0.941	0.985	0.935	0.973	0.937
New Loss	0.9	1	0.969	0.998	0.980	0.976	0.990

Table 3. Model Tuning Table

ETA	Nroud	MaxDepth	MaxDeltaStep	MinSplitLoss	Pred1. Precision		Pred1. Accuracy		Pred2. Accuracy
					Test	Train	Test	Train	
0.5	20	10	5	1	0.976	0.995	0.986	0.980	0.991
0.5	20	5	5	1	0.953	0.985	0.983	0.970	0.978
0.5	20	20	5	1	0.989	0.998	0.978	0.982	0.983
0.5	20	10	1	1	0.940	0.989	0.984	0.973	0.988
0.5	20	10	10	1	0.976	0.995	0.978	0.980	0.978
0.5	20	10	5	0	0.917	0.998	0.976	0.988	0.984
0.5	20	10	5	2	0.988	0.999	0.953	0.989	0.984
0.5	10	10	5	1	0.976	0.995	0.953	0.981	0.984
0.1	20	10	5	1	0.976	0.995	0.957	0.979	0.957

Figure 7. Performance of the XGBoost, GBDT, Random Forest, and Decision Tree Models



Benefits of the New Strategy

In a determined Bitcoin network environment, two factors affect the revenue of miners and mining pools: miners' computing power and the time difference between the arrival of mainchain blocks and the earliest stale blocks. This paper uses 20 days of Bitcoin test network data from July 3rd to July 23rd as our database. From the previous prediction model, we can obtain $A_c = 0.991$, the average value of the 20-day difficulty is 28T, the current total network hash computing capacity is 204.75 Eh/s, the outgoing block bitcoin reward is 6.25BTC, the average transaction fee is 0.015 $BTC/block$, and the bitcoin price is approximately 24000$USD/block$. This paper calculates $p_{mb} = 373/426$ and $p_T = 4/426$ in that period. According to Shahsavari et al. (2020), the current bitcoin blockchain block size is approximately 1 M. The time for the block to spread to ninety percent of the nodes across the network is approximately 10 s. Thus, the propagation interval between most of the stale blocks generated by honest miners and the mainchain blocks is also within 10 s. Therefore,

we set the propagation interval between the stale and mainchain blocks to 10 s and the total mining time to 10 min.

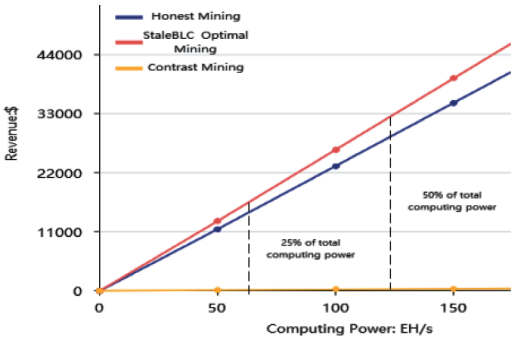
Using the above data as a basis, this paper compares the gains of three different strategies at different computing powers when affected by stale blocks in Figure 8. Meanwhile, this paper also uses the simulator to compare the revenue of selfish mining mentioned by Eyal and Sirer (2018) and the revenue of RL mining mentioned by Wang et al. (2021) in the same scenario. Finally, this paper denotes γ the proportion of honest miners who choose to mine on the node or pool chain branch that adopts the strategy. Notably, the effects of stale blocks and propagation delays are not considered in both articles.

The propagation interval between the stale and mainchain blocks is not fixed. This paper calculates the gains of different strategies under the influence of stale blocks with variable propagation intervals. To make the revenue more intuitive, this paper sets the miner's computing power to 10 EH/s with other conditions unchanged. The results are shown in Figure 9.

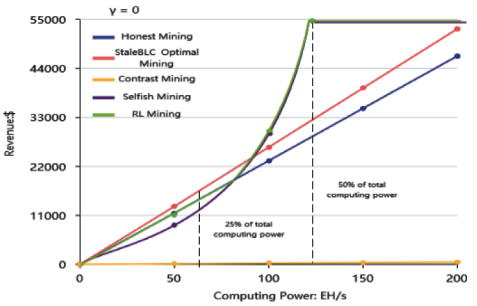
As a result, this paper concludes that miners can obtain more revenue with the optimized honest mining strategy compared to honest mining under the influence of stale blocks. Ideally, when the arrival time of stale blocks is close to that of mainchain blocks, it can increase total revenue by 10% to 15% compared to the original mining model under the influence of stale blocks, which can lead to 1.3% to 3% profit growth (the normally stale block rate is 2 to 3 percent, and the cost of block mining is approximately 85% of the revenue). Furthermore, the strategy will improve profits if the network environment in which the miner is located is volatile and the stale block rate is higher.

We find three improvements if we compare the optimization strategy in this paper with some existing attack strategies of the selfish mining class. First, the optimization strategy does not require miners to keep private attack chains. Second, it does not require malicious attack methods such as

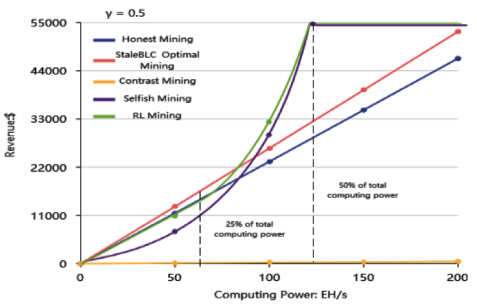
Figure 8. Variation in Revenue with Computing Power



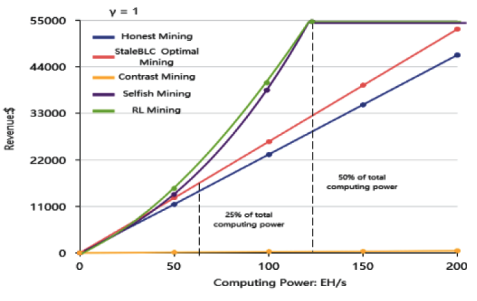
(a) Original



(b) $\gamma = 0$

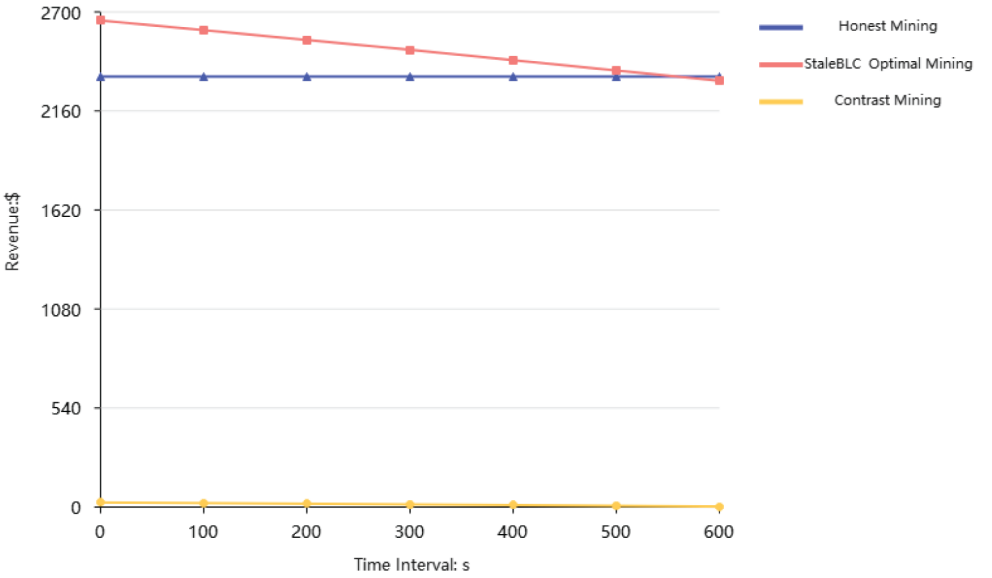


(c) $\gamma = 0.5$



(d) $\gamma = 1$

Figure 9. Variation in Revenue with Broadcast Time Space



chain substitution to damage the revenue of other miners. Third, it does not require miners to obtain more information about the network and nodes and does not harm the current bitcoin network. Furthermore, under the influence of stale blocks, with $\gamma = 0.5$, the optimization strategy can effectively maximize the profit of all honest miners with less than 80 EH/s of computing power (approximately 40% of the network's computing power) to increase the miner's total revenue during the mining process.

CONCLUSION

This paper investigates the generation of stale blocks in the Bitcoin network. The current research has not analyzed the factors influencing the generation of stale blocks. Therefore, in this paper we first analyze the factors that influence the generation of stale blocks. Then, this paper proposes a new machine-learning prediction model based on the XGBoost model. The model is the first to predict network stale block generation by machine learning. However, the current blockchain data collection methods cannot collect enough blockchain formation data to train the model. Therefore, we propose a new node data collection method to obtain real-time data to train models.

The method generates detection nodes by code instrumentation to obtain more detailed real-time data. First, this paper labels real-time data based on the principles of stale block generation. Then, this paper separates the real-time data into a training set and a test set for model tuning.

After tuning the loss function and parameters of XGBoost, the model can accurately predict the generation of stale blocks. The precision of the model is 0.976. The model can also distinguish stale blocks from mainchain blocks with the same height and stale blocks with 0.991 accuracy. Ultimately, this paper proposes an optimized mining strategy based on the prediction model. The model can boost profits for mining nodes by 1.5% to 3.3%.

Some problems remain and will be addressed in future research, of which there are two key issues. First, the number of detection nodes for collecting and analyzing data is small,

which results in incomplete data collection. Second, this paper needs to consider more data from the detection network to increase the prediction accuracy. In future research, we will focus on methods to arrange detection nodes in the network more efficiently. As a result, it can obtain more complete real-time network data and improve the prediction accuracy achieved by the method.

REFERENCES

Bag, S., Ruj, S., & Sakurai, K. (2016). Bitcoin block withholding attack: Analysis and mitigation. *IEEE Transactions on Information Forensics and Security*, *12*(8), 1967–1978. doi:10.1109/TIFS.2016.2623588

Böhme, R., Christin, N., Edelman, B., & Moore, T. (2015). Bitcoin: Economics, technology, and governance. *The Journal of Economic Perspectives*, *29*(2), 213–238. doi:10.1257/jep.29.2.213

Bonneau, J., Miller, A., Clark, J., Narayanan, A., Kroll, J. A., & Felten, E. W. (2015). Sok: Research perspectives and challenges for bitcoin and cryptocurrencies. *2015 IEEE Symposium on Security and Privacy*, 104–121. doi:10.1109/SP.2015.14

Boscovic, D., Chawla, N., & Tapp, D. (2022). *Block propagation applied to Nakamoto networks*. https://www.darrentapp.com/pdfs/Block_Propagation_Applied_to_Nakamoto_Networks.pdf

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

Btcd. (2022). *Bitcoin in Go*. https://github.com/btcsuite/btcd/blob/52082fcbf9273b28e152fa27a29c7f63863b1518/blockchain/difficulty.go (Original work published 2013)

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. *KDD '16: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794. doi:10.1145/2939672.2939785

Chen, T., Li, Z., Zhu, Y., Chen, J., Luo, X., Lui, J. C.-S., Lin, X., & Zhang, X. (2020). Understanding Ethereum via graph analysis. *ACM Transactions on Internet Technology, 20*(2), 18:1-18. 10.1145/3381036

Chen, T., Zhang, Y., Li, Z., Luo, X., Wang, T., Cao, R., Xiao, X., & Zhang, X. (2019). Tokenscope: Automatically detecting inconsistent behaviors of cryptocurrency tokens in Ethereum. *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 1503–1520. doi:10.1145/3319535.3345664

Eyal, I., & Sirer, E. G. (2018). Majority is not enough: Bitcoin mining is vulnerable. *Communications of the ACM*, *61*(7), 95–102. doi:10.1145/3212998

Feng, C., & Niu, J. (2019). Selfish mining in Ethereum. *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, 1306–1316. doi:10.1109/ICDCS.2019.00131

Göbel, J., Keeler, H. P., Krzesinski, A. E., & Taylor, P. G. (2016). Bitcoin blockchain dynamics: The selfish-mine strategy in the presence of propagation delay. *Performance Evaluation*, *104*, 23–41. doi:10.1016/j.peva.2016.07.001

Grossman, S., Abraham, I., Golan-Gueta, G., Michalevsky, Y., Rinetzky, N., Sagiv, M., & Zohar, Y. (2017). Online detection of effectively callback free objects with applications to smart contracts. *Proceedings of the ACM on Programming Languages, 2*(48), 1-48. doi:10.1145/3158136

Heilman, E., Kendler, A., Zohar, A., & Goldberg, S. (2015). Eclipse attacks on bitcoin's peer-to-peer network. *Proceedings of the 24th USENIX Conference on Security Symposium*, 129–144.

Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. *2017 IEEE International Conference on Computer Vision (ICCV)*, 2980–2988. doi:10.1109/ICCV.2017.324

Long, W., Wu, C. H., Tsang, Y. P., & Chen, Q. (2021). An end-to-end bidirectional authentication system for pallet pooling management through blockchain internet of things (BIoT). *Journal of Organizational and End User Computing*, *33*(6), 1–25. doi:10.4018/JOEUC.290349

Nakamoto, S. (2008). *Bitcoin: A peer-to-peer electronic cash system*. https://bitcoin.org/bitcoin.pdf

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

Srinivasan, B. S., & Lee, L. (2017). *Quantifying decentralization*. news.earn.com. https://news.earn.com/quantifying-decentralization-e39db233c28e

Tang, G., & Zeng, H. (2021). E-Commerce Model Oriented to Cloud Computing and Internet of Things Technology. *International Journal of Information Technologies and Systems Approach*, *14*(2), 84–98. doi:10.4018/IJITSA.2021070106

Vujičić, D., Jagodić, D., & Ranđić, S. (2018). Blockchain technology, bitcoin, and Ethereum: A brief overview. *2018 17th International Symposium INFOTEH-JAHORINA (INFOTEH),* 1–6. doi:10.1109/INFOTEH.2018.8345547

Wang, T., Liew, S. C., & Zhang, S. (2021). When blockchain meets AI: Optimal mining strategy achieved by machine learning. *International Journal of Intelligent Systems*, *36*(5), 2183–2207. doi:10.1002/int.22375

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

Xu, Y., Wu, G., & Chen, Y. (2022). Predicting patients' satisfaction with doctors in online medical communities: An approach based on XGBoost algorithm. *Journal of Organizational and End User Computing*, *34*(4), 1–17. doi:10.4018/JOEUC.287571