No-Proof Consensus-based Light Blockchain for Distributed Computing Scenarios

*Abstract*—Distributed computing faces a persistent multi-agent trust dilemma. In the computation process, participants may maliciously attack the system for personal gain by providing false data. Blockchain provides a possible solution for this problem with its immutability and multi-party consensus. However, existing blockchain data throughput has long been queried owing to its exorbitant time and energy costs by consensus mechanisms. This paper proposes a light blockchain structure in distributed computing scenarios. A No-Proof consensus (NPC) mechanism is designed for distributed computing problems, with no extra proving process such as Proof-of-Work, Proof-of-Stake. This consensus mechanism notice that the distributed computing result has proven to be valid in the computation process automatically, and doesn’t need to be verified again in the consensus mechanism. Further, single-threaded data processing ability of blockchain structure certainly leads to low efficiency when being applied to distributed computation problems. An NPC-blockchain is constructed in this paper to solve this problem. In this structure, the distributed computing is done off-chain and an oracle is designed to upload the computing results to the blockchain asynchronously. Upon the contributed done in this paper, a distributed energy trading model is provided as a case to verify the superiority of the designed blockchain in contrast with other similar structures.

*Index Terms*—Blockchain, consensus mechanism, distributed computing, optimization

# Introduction

Distributed computing is a method that utilizes multiple computing resources to work together to solve complex problems, and it is widely used in many fields, such as power systems[1]. Superiorities of distributed computation over centralized computation have long been studied and distributed computation significantly surpasses centralized schemes in robustness[2]. Also, distributed computation shows more scalability in large systems in terms of communication times and computing time[3].

However, there is a multi-agent trust problem in distributed computing as participants may attack the system maliciously for their benefit by providing false data in the computation process[4]. Therefore, no participant can trust the computation results provided by other participants. Researchers have made their efforts to solve this problem. A novel resilient distributed computing algorithm against adversarial attacks is proposed, the effectiveness of which is proved by comparing the difference between the final solution and the optimal solution[5]. Also, the method for detecting and isolating malicious agents when solving a distributed computing problem is proposed[6]. However, these researches either can only solve specific distributed computation problems and are not general, or can only reduce the impact of malicious attacks to a certain extent, but cannot eliminate their interference.

As a decentralized distributed data recording and operation structure, blockchain offers a viable solution to this problem by providing a working environment of trusted collaboration in the absence of an authoritative centralized agent. For instance, blockchain-based frameworks have been established for energy trading, preventing manipulation of the trading scheme by an outside attacker[7],[8], and a collaborative approach has been proposed using Hyperledger fabric blockchain for fog node management[9]. It is noticed that in existing studies of combining blockchains with distributed computing, distributed computing work is done on-chain via smart contracts of the blockchain and recorded using the blockchain. However, this results in a decline in distributed computing performance. This is because although the blockchain is distributed as a data recording device, the blockchain structure needs to collectively process data as a whole[10],[11]. This means that blockchain processes business single-threadedly, resulting in a structure that applies blockchain directly to distributed computing will lose its advantages of high robustness and throughput compared to centralized computing.

Apart from this, blockchain has also been found by researchers to be challenging in simultaneously achieving decentralization, reliability, and low resource consumption[12]. Most existing blockchain structures exhibit excessive energy and time consumption while ensuring decentralization and reliability. This occurs because servers need to attach additional proof to recorded results; otherwise, other servers will not trust the unproved results. This process of seeking proof to convince other servers is known as the consensus mechanism, which is crucial for ensuring the trustworthiness of data in blockchains. For example, a problem difficult to solve but easy to prove is proposed in Proof of Work (PoW), and one who provides a result proven to solve the problem will win the accounting rights[13]. Similarly, the stake is the proof in Proof of Stake (PoS)[14], Delegate Proof of Stake (DPoS)[15], and the approval degree of data represented by voting results is the proof of Byzantine Fault Tolerance (BFT)[16] and Practical Byzantine Fault Tolerance (PBFT)[17]. However, the process of finding evidence within consensus mechanisms typically involves complex computations, consuming considerable energy and time, and the problem has not been universally solved.

Researchers have been striving to mitigate the high consumption characteristic of consensus mechanisms. On one hand, some researchers focus on reducing the likelihood of erroneous requests being made and subsequently examined. They propose various methods for selecting suitable consensus mechanism participants, such as weighting participants by their contributions[18] or using Genetic Algorithms to filter participants[19], thereby minimizing the effort spent in verifying dishonest results. However, these consensus mechanisms are not applicable to distributed computing because distributed computing involves multiple participants, and selecting participants may prevent the completion of the distributed computing tasks. On the other hand, researches have shown that the process of verifying evidence can be simplified, thereby reducing the consumption of the consensus mechanism. For instance, a Reputation-based BFT consensus mechanism has been proposed to simplify the process of evidence verification in BFT[20]. In conventional PBFT mechanisms, participants vote on results and cross-verify the voting results, which serve as evidence. In large-scale Internet of Things (IoT) networks, where avoiding forged votes is unnecessary, the reputation of participants can be used to ensure the authenticity of votes. Additionally, in the scenario of tracking carbon emission flow, a consensus mechanism is proposed where constricts of carbon emission flow serve as proof for servers to approve the validity of the carbon emission result[21]. Nevertheless, the blockchain systems in aforementioned studies cannot entirely eliminate the time and resource consumption required for servers to reach consensus; they can only partially reduce their impact. Thus, the consensus mechanism remains a significant obstacle in minimizing blockchain consumption, especially when using blockchain to record results of distributed computations.

To fill the technical gap mentioned above, the contributions of this paper are as follows:

1) A no-proof consensus mechanism is designed for distributed computing problems, with no extra proving process. It employs the constraints in the distributed computation problem as proof in the consensus. In a blockchain applying NPC, when a distributed computation result is to be recorded, the result has already been proved to be fulfilling constraints of the computation by participating servers in the computation process, thus there’s no need for servers to find and verify proof for this result to be valid. This consensus mechanism avoids increasing time and energy costs while maintaining the ability to ensure the credibility of data stored by blockchain structure.

2) A blockchain structure is constructed for solving distributed computing problems and storing its result reliably and credibly. The blockchain structure is constructed of newly designed consensus mechanism, data depository, smart contract, and oracle. In this structure distributed computing is done off-chain, and the oracle will obtain the result to record it on-chain when the result is inquired. These two steps are designed to be asynchronous, thus distributed computing is still distributed, while result recording is single-threaded due to blockchain working as a whole. The structure can avoid reducing throughput and robustness during the distributed computing process.

3) A local energy trading model is provided as an example for distributed problems. The working process of the blockchain structure is explain based on this example and a case study is done. In the case study, different scenarios with problematic servers are simulated to verify the robustness and trustworthiness of the blockchain proposed. Also, the comparison is done between the no proof-based consensus mechanism with existing schemes to reveal its superiorities.

# No proof-based consensus blockchain for distributed computing

Distributed computation means that multiple servers work together to tun the algorithm. However, the problem of these servers not being able to trust each other is brought up. The No-Proof Consensus-based blockchain (NPC-blockchain) is designed to address distributed computing issues and record their result reliably.

## Blockchain structure for recording computation result

The No-Proof Consensus-based blockchain (NPC-blockchain) is designed to address distributed computing issues and record their result reliably. To better explain the structure of this blockchain, basic assumption is made about distributed computing. It is assumed that each participant is associated with a corresponding server, which is responsible for executing the participant's computational tasks and running the NPC-blockchain. The NPC-blockchain structure is shown in Fig. 1. Information passing in the structure is in patterns as shown in the figure. The consensus mechanism is essential for ensuring the trustworthiness of on-chain results, while other auxiliary structure make the performance of the NPC-blockchain better by avoiding to process computation single-threadedly. The auxiliary structure is divided into 4 parts, i.e., data depositary, communication network, oracle, and smart contract. They will be separately explained in the following structure.

## Asynchronous No Proof-based Consensus Mechanism

The consensus layer usually refers to the encapsulation of blockchain consensus mechanisms. In essence, its primary role revolves around resolving trust issues among blockchain participants, ensuring the consistency of multiple records across distributed servers, and thwarting unauthorized data tampering attempts. A no-proof consensus mechanism is proposed in this paper for the trustworthy optimization result to be recorded by the blockchain. Specifically, it is found that in a distributed computation scenario, participants in distributed computing agree upon a computation protocol beforehand to determine known parameters, target results, implementation steps, and constraints in the computation problem. The distributed computation result is considered correct only if it meets the agreed constraints, thus this characteristic of the result serves as credible evidence in NPC. The working steps of this consensus mechanism is as followed.

1) A server proposes a request to upload the distributed computation result to the blockchain. It packs the result it recorded off chain into a block and broadcast the block to all participating servers in the consensus mechanism.

2) Each server that receives the broadcasted result check whether the proposing result is the same with the result it recorded off-chain, as the result it recorded is seen to be valid by itself.

3) If the two results aren’t the same, the server will send a veto opinion against the result together with the two results to the server on which the oracle runs. Otherwise, the server will return its approval together with the result.

Until the termination time *t*ter ends, if the oracle receives over *n*/2 votes approving the proposing result, consensus on the result is deemed to be reached. Oracle forms a response with this result and report it to the smart contract for further use. Servers form block with this result and add it to the blockchain. If the oracle receives less than *n*/2 approving votes, it is agreed that the result does not pass the consensus mechanism. The no-proof consensus mechanism proposed in this paper reduces the complexity, energy consumption, and time expenditure of consensus mechanisms. To clarify this, we compare the implementation steps of existing consensus mechanisms with those of NPC. The implementation process of existing consensus mechanisms can be divided into four phases, **as:**

****

Fig. 1 No-proof blockchain structure for distributed computing

**1) Request proposing phase: A server proposes request for uploading the result to the chain.**

**2) Proof proposing phase: A proof is proposed to guarantee the trustworthiness of the result.**

**3) Proof verification phase: A proof is found to guarantee the trustworthiness of the result. Other servers verify the trustworthiness of the result with the proof.**

**4) Consensus reaching phase: All servers reach a consensus of the result.**

Therefore, in NPC, the results stored in the off-chain database during distributed computation are already proven to be trustworthy. As a result, when recording results of distributed computation problems, NPC enables servers to upload the computation results to the chain directly, and the phase of proving the reliability of results is skipped. Applying NPC to distributed computation problems can avoid the time and resource consumption associated with proof proposing phase in existing consensus mechanisms. In contrast, PBFT is provided as a typical example that includes an additional proof proposing process unrelated to the recorded results. In each phase of the consensus mechanism, the information transmission processes required for implementing NPC and PBFT are compared as shown in Fig. 2. In the figure, 0 represents the server initiating the consensus request in NPC, and 1, 2, and 3 are the other servers. In PBFT, the Requestor initiates the request to chain data on behalf of server 0, with 1, 2, and 3 being the other servers. Notably, in both NPC and PBFT, 3 represents a server that cannot communicate with other servers. Fig. 2 also illustrates the communication processes in each phase of the two consensus mechanisms. The figure demonstrates that in the request proposing, proof proposing, and proof verification phases, NPC only requires one broadcast, whereas PBFT involves extensive communication, leading to significant time and energy resource consumption. Furthermore, the computational demands on each server are negligible in both approaches, indicating that the primary resource consumption of consensus mechanisms stems from inter-server communication.

Additionally, Ref. [22] provided a consensus mechanism called PoSo for optimization problems. In this consensus mechanism, a server proposes an optimization solution to start the consensus mechanism and uses the optimization solution as the proof. Thus, this method can also simplify the consensus process by combining request proposing phase and proof proposing phase. However, in the proof verification phase of this consensus mechanism, other servers need to crosswise communicate with each other to verify that the result broadcasted is valid. This is because servers may send different solution to different servers in PoSo, while such attempt will be block in NPC because there are not enough approving votes for the result. The communication processes of PoSo are also shown in Fig. 2.

****

Fig. 2 Comparison of communication process in different consensus mechanisms.

## Auxiliary structure of NPC-blockchain

The data depositary can be divided into on-chain and off-chain parts, the off-chain data is stored independently by each server in its database, and the on-chain data is stored in the block of a blockchain. The reliability of data in the off-chain database is not guaranteed; if a server attempts to maliciously attack the NPC-blockchain, it can still tamper with its off-chain database. However, on-chain data is protected and immutable due to the presence of multiple distributed recorders. To avoid the reductions in throughput and stability caused by blockchain's overall single-threaded data processing, data that constantly changes during iterations is recorded off-chain. Therefore, during distributed computing, data is stored in the off-chain database. Only when the computation is complete and the data needs to be reliably recorded, the results of the distributed computation will be uploaded to the blockchain after verification.

The communication network is the part that serves the communication between the servers participating in the blockchain, and the communication is in the form of a peer-to-peer network, that is, any two servers can communicate directly. Communication can be divided into on-chain communication and off-chain communication. For on-chain communication, one server needs to call the routing contract before communicating with other servers, and that will bring extra communication cost comparing with off-chain communication. Thus, off-chain communication is used in the distributed computation process and on-chain communication is only used when the result is to be recorded on the chain.

Oracles are designed to retrieve data from off-chain databases and transfer them onto the blockchain. The blockchain is an isolated system where smart contracts, serving as its "brain," can only accept and provide fixed outputs, unable to dynamically control the blockchain by actively retrieving external information as input. To ensure the correct operation of the NPC-blockchain, oracles retrieve the computing results recorded by servers in their off-chain databases and provide these results as inputs to smart contracts. Generally, the operation of an oracle is divided into three parts: request, integration, and response. The process is as follows: the smart contract sends data requests to participating servers via the oracle, the oracle retrieves off-chain data from servers and integrates it to form a response on-chain, and finally, the oracle returns this response to the smart contract. Unlike existing oracles, during the integration process of the designed oracle in this study, blockchain participants are required to verify the retrieved computation results via the consensus mechanism and upload the verified results to the blockchain. This on-chain result is then returned as a response to the smart contract.

There are 3 smart contracts in the NPC-blockchain: the computation request contract, the result obtaining contract, and the routing contract. The computation request contract is invoked by the requester to initiate distributed computation. This contract requires an input of the distributed computation protocol between participating servers, including the target result, constraints, and relevant parameters. Upon invocation, this smart contract broadcasts the distributed computation information to participants and notifies them to begin computation. The result obtaining contract is invoked by the requestor to request the distributed computation result. This contract first calls the oracle to retrieve the recorded computation results from the off-chain database, subsequently provides the results to the requester, and ultimately informs the requester of the results' blockchain address for future reference. The routing contract must be invoked by servers for on-chain communication with other servers, while off-chain communication does not require the assistance of a smart contract. In the structure, the server loading the oracle can only return the result obtained by the oracle with the routing contract in the NPC-blockchain.

## Oracle obtaining Reliable Data

Considering the complexity of on-chain communication, and the excessive energy cost of on-chain computing, the distributed computing process is done off-chain and uploaded on-chain afterward in the proposed blockchain structure. However, blockchain is a deterministic, closed system that can only obtain data on the chain, but not the real-world data off the chain. Thus, we design an oracle structure that can initiatively acquire the off-chain data and form a response.

The operation of the oracle is divided into three parts, namely, request, integration, and response. Concretely speaking, the blockchain first sends data requests to the oracle through the smart contract, then the oracle obtains off-chain data and forms a response, and finally the oracle returns the response to the smart contract. Different from existing oracle structures, the trustworthiness of the data obtained by the oracle is protected by modifying the data integration process.

The detailed steps of the modified integration process follow:

1) The oracle sets the failure number of the request hop to 0, and the upper limit of hop as *hopmax*.

2) The oracle calls the routing function to send the request to a random server in the blockchain. *hop* increases by 1 and if it reaches *hopmax*, the data obtaining process fails.

3) The server broadcasts the result it recorded off-chain. Participating servers reach a consensus on whether to upload this result to the blockchain or not with the NPC introduced above.

4) Participating servers will return the result to the oracle after they reaches a consensus. If they agree with the result and upload it to the blockchain, then the oracle return the result to the smart contract and the request is processed.

5) For condition that the result doesn’t pass the consensus mechanism, increase *hop* by 1 and send the request another server. If over half servers veto against the result for a same optimal result in the consensus mechanism, the oracle should find the server that firstly proposes this result and send the request to it. Otherwise, send the request to a random server that has not received the request yet. Then return to step 3).

This modified oracle can ensure obtaining the optimal result recorded by the servers off-chain in most circumstances, however, this feature of the oracle may cease to be effective under certain circumstances. We suppose that the number of dishonest servers trying to attack the oracle is *h*, and when *h*>*hopmax*, there is a possibility that an optimal result can’t be obtained. This is because the request may be continuously sent to dishonest servers in the worst circumstance, leading to a lack of optimal solutions recorded by honest servers. As shown in the "Results" part, the time required for NPC is short enough, so the *hopmax* should be set as large as possible. Since NPC-blockchain tolerates at most half of the servers to be dishonest, the best value of *hopmax* is half of the number of servers, in which condition the oracle can guarantee that the optimal result recorded is obtained.

Considering the complexity of on-chain communication, and the excessive energy cost of on-chain computing, the distributed computing process is done off-chain and uploaded on-chain afterward in the proposed blockchain structure. However, blockchain is a deterministic, closed system that can only obtain data on the chain, but not the real-world data off the chain. Thus, we design an oracle structure that can initiatively acquire the off-chain data and form a response.Blockchain-based distributed ADMM optimization

In this part, the ADMM algorithm is introduced as a fully distributed method for solving optimization problems in which multiple entities need to gradually change their decision in the optimization to achieve an optimal value of the objective function. The detailed of ADMM algorithm can be found in appendix at https://github.com/asdr1332/No-Proof-Consensus-based-Light-Blockchain-for-Distributed-Computing-Scenarios.

The distributed optimization process with NPC-blockchain can be divided into 3 steps as shown in Fig. 3: ①Optimization proposal on chain; ②Distributed optimization off chain; ③Obtain optimization result on chain.



Fig. 3 Blockchain-based process of distributed optimization

For the first step, the requestor calls the computation request contract to initiate the optimization and provides optimization information through the input of the contract, including optimization objective function, optimization termination time, etc. Then the contract determines the servers participating in the optimization and calls the routing function to broadcast optimization information to these servers.

For the second step, each server solves the optimization problem off-chain according to the distributed ADMM algorithm introduced. After an iteration, all servers check whether this received result: 1) fulfills the termination condition of the optimization problem; 2) has a better objective value than the result it previously recorded off-chain, or there is no previous result recorded. If this new result meets the two conditions, the servers will record this result, otherwise, it will keep the previous result (if there is one). Once all servers have recorded a result that satisfy the termination condition off-chain, the second step ends.

For the third step, the requestor calls the result obtaining contract to obtain the result for the distributed optimization. The result obtaining contract will call the oracle for a result. The oracle will send the result requests the servers in turn for their off-chain results, and the server receiving the request will try to upload its result on-chain by the consensus mechanism. If there is a result recorded in a server’s off-chain database, the server will pack the result into block and the block will be uploaded to the chain, thus the result is moved from off-chain data depositary to on-chain data depositary. The server will send the result to the oracle and the oracle will return this result on-chain to the smart contract. All results obtained from servers will be sent back to the requestor by the smart contract to end the result obtaining process. The requestor will use the result with most appearance in the returned information. If a subsequent requestor needs to call the optimization result again, the requestor can get the result directly from the blockchain instead of calling the result obtaining contract. However, if the optimization is not completed yet when part ③ begins, no result will be uploaded to the blockchain. The oracle will fail to obtain a valid result, and the requestor will be informed that the optimization has no result yet by the result obtaining contract.

It should be noticed that parts ① and ② may be asynchronous with part ③, as the former starts when the requestor proposes the optimization problem, and the latter starts when the requestor calls for the optimization result. Part ③ ends as well in this condition.

# Applications to distributed energy trading

Also, a typical local energy trading model is presented. The no-proof blockchain-based distributed ADMM optimization algorithm is applied to the model.

## Application in energy trading case

IEEE 14-bus system is tested and assume there are 14 agents trading electricity in an energy trading system. This energy trading model is constructed in the appendix part at https://github.com/asdr1332/No-Proof-Consensus-based-Light-Blockchain-for-Distributed-Computing-Scenarios. The blockchain-based ADMM algorithm proposed in this paper is applied to find the optimal solution for the energy trading in this system. In order to prove the reliability of the results recorded on the blockchain by this method, some scenarios of malicious attacks by servers are considered. These include attacks during iterations and the result-obtaining process.

For the attack behavior during the iteration process, we first simulate the original iteration process, during which we set the termination time to 100 seconds. The result after each iteration in optimization under different scenes is shown in Fig. 4. According to the theory above, the iterative results of the proposed method under normal circumstances (when there is no dishonest server disrupting the process or some server fails to communicate normally) are consistent with those of the direct use of the ADMM algorithm apparent.Three different attack strategies are designed against the normal iteration process, namely, malicious servers providing self-regarding data instead of real iteration data, providing random data instead of real iteration data, and refusing to provide iteration data. Obviously, the last type of malicious attack has the same effect as an uncommunicable server and is therefore considered as the same problematic server. Thus, *x* is used to represent the number of malicious servers that transmit incorrect data, and *y* is used to represent the number of servers that do not send information, including malicious servers that refuse to send data and servers with communication failures.

In the above cases, we respectively considered the situation that there are different numbers of problematic servers in the system, simulated the process of optimizing, reaching a consensus, and recording the result for optimization scenes with different values of *x* and *y*. The line with label *x*=0, *y*=0 in the figure represents the iterative process under normal circumstances. It is not difficult to see that the optimization is completed in all cases before the terminal time, and in the scenes with interference, the results of some iteration rounds are better than those of no interference (that is, closer to the final iteration result), while the results of other rounds are almost unaffected by interference. This conclusion shows that attempts to misrepresent data to interfere with the optimization process can be identified and corrected in the distributed computation process off-chain.



Fig. 4 Optimizing process in different problematic conditions

In order to further illustrate the reliability and stability of the proposed method and prove that the method can still maintain good performance when problematic servers exist, we compared the final optimization results, the time required to reach the feasible solution, the number of rounds required to reach the feasible solution, and the total iteration rounds under the above simulation conditions in Tab. 1. It is not difficult to find that when problematic servers exist, the time required to arrive at the fastest feasible solution and the total iteration rounds increase slightly. This is because the method in this paper needs to judge the solution and correct them. In contrast, the final optimization result and the number of rounds required to arrive at the feasible solution barely change. This shows that the proposed method is quite stable as well as ensuring the reliability of the results, and will not be greatly affected by the emergence of malicious servers and uncommunicable servers.

Tab. 1 Relative data in optimization process in different problematic scenarios

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Number of problematic servers | The final optimization results (transaction cost) | The time required to reach a feasible solution | The number of rounds required to reach the feasible solution | the total iteration rounds |
| *x*=0,*y*=0 | $18,850.56 | 77.52s | 43 | 49 |
| *x*=1,*y*=0 | $18,895,54 | 80.72s | 44 | 54 |
| *x*=2,*y*=0 | $18,801.54 | 80.83s | 42 | 52 |
| *x*=1,*y*=1 | $18,866.72 | 83.79s | 44 | 51 |
| *x*=2,*y*=1 | $18,840.50 | 85.78s | 45 | 51 |
| *x*=6,*y*=2 | $18,770.80 | 87.18s | 45 | 49 |

For the attack behavior during the result-obtaining process, two kinds of fraud measures are designed to be done by dishonest servers, namely, lying and not reporting data. To this end, we divide the servers involved in the result-obtaining process into three groups: A, B, and C, representing honest servers, dishonest servers that misrepresent data, and dishonest servers or uncommunicable servers that do not report data. Group A records the correct data 18.85 thousand dollars, while group B is divided into two groups, B1 and B2. B1 is a group of servers that has network problem in the distributed computation process and does not record the final result. Servers in this group record a result valueing 19.21 thousand dollars. This result also fulfills the constraints in the optimization problem, but it is not the optimal solution. Servers in B2 will try to attack the blockchain by providing errorous results. The result provided by B2 does not meet the optimized termination condition, and the results they record are favorable for the dishonest servers in groups B2 and C. All dishonest servers will collude to assist in the attack. Additionally, the server on which the oracle runs is denoted as group R.

Based on this, the four fraud scenarios are designed with different ways of attacking. To visualize the message-exchange process, servers’ operation in the consensus mechanism and the final result uploaded to the chain is represented in Tab. 2. In the table, word like “<*tc*>” represents the result with the result value *tc*. “A<*tc*>” represents the server approves the broadcasted result <*tc*>, while word like “D<*tc*>” represents the server dissents the broadcasted result <*tc*>.

For scenario 1, B1 and B2 provide error results, but the two groups together account for less than half of the total number of servers. B2 will provide a error result valuing 17.93 thousand dollars, and this result certainly don’t fulfill the constraints in the problem. B1's result is protested by A and B2, so R rejects B1's result, and B2's result is protested by A and B1. Both branches will lead to failure in handling the request and the request will be passed on as *hop* increases by 1. If the request is sent to the server in A before the *hop* reaches *hopmax*, the request will be processed, with 18.85 thousand dollars as the final result obtained by R. For scenario 2, there are servers from C that can’t communicate with other servers properly. If the request is sent to C, the consensus process won’t begin because C won’t broadcast its result and try to upload it on-chain. Thus, results from A will pass the consensus mechanism and R will obtain the result with the result value of 18.85 thousand dollars. For scenario 3, servers are the same as in scenario 1. However, B2 sends different result to different servers to win their approval. For servers in A, it will send the result with the result value of 18.85 thousand dollars, while the result will be 19.21 thousand dollars of B1. Yet this behaviour will let the correct result pass the consensus mechanism as R will receive over half approval for the correct results. For scenario 4, there are servers from group A, B1, B2 and C, and over half servers are from B1 and B2. In this condition, B2 may choose to approve the result broadcasted by B1. Thus R will receive over half approval of this result, and the correct result won’t be uploaded on-chain.

The above scenarios prove that the method in this paper is quite reliable in the process of result obtaining. Only when over half servers record errorous results off-chain in the system, the result cannot be uploaded to blockchain correctly.

Tab. 2 Result-obtaining process in different fraud scenarios

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Scenario | Proposing server | Votes from A | Votes from B1 | Votes from B2 | Votes from C | Consensus result | Result obtained by R |
| 1 | B1 | D<19.21> | A<19.21> | D<19.21> | \ | Fail | <18.85> |
| B2 | D<17.93> | D<17.93> | A<17.93> | \ | Fail |
| A | A<18.85> | D<18.85> | D<18.85> | \ | Pass |
| 2 | C | \ | \ | \ | \ | Fail | <18.85> |
| A | A<18.85> | \ | \ | \ | Pass |
| 3 | B1 | D<19.21> | A<19.21> | D<19.21> | \ | Fail | <18.85> |
| B2 | A<18.85> | A<19.21> | A<17.93> | \ | Pass |
| A | A<18.85> | D<18.85> | D<18.85> | \ | Pass |
| 4 | B1 | D<19.21> | A<19.21> | A<19.21> | \ | Fail | <19.21> |
| B2 | D<19.21> | A<19.21> | A<19.21> | \ | Pass |
| C | \ | \ | \ | \ | Fail |
| A | A<18.85> | D<18.85> | D<18.85> | \ | Fail |

## Comparison revealing advantages

The distributed ADMM optimization algorithm with NPC-blockchain is compared with other methods to solve and record distributed computation problems, especially with blockchains that do not use the verifiable feature for optimization problems. Specifically, the following methods that can compute and store optimization results are compared with the methods described.

Ref. [23] proposed a method of building a database directly on the optimized server after centralized optimization. However, using the centralized database has stability defects[24]. Once the server storing data is in malfunctions, it may lead to data loss or data communication failure and thus the data in the database can’t be obtained successfully. In addition, performance in data throughput and scalability is a bottleneck in centralized structure[25], while the blockchain-based distributed data storage used in this paper can provide higher data throughput as each server can handle data output business with oracle, and is highly scalable.

Compared with the method of storing results with a distributed database, the results stored by this method are highly reliable. The blockchain-based approach used in this paper guarantees the trustworthiness of the information, in contrast to the additional complex identification[26] or other measures to avoid erroneous information published by dishonest servers in a distributed database structure.

Furthermore, some blockchain with existing consensus mechanisms is used when simulating the process of storing optimization results with blockchains. These common consensus mechanisms include PoW, DPoS, Proof of Authority (PoA), Raft, and PBFT. Ref. [27] provides some indicators of comparison between blockchains with different consensus mechanisms and demonstrates the performance of the common consensus mechanisms mentioned above. Meanwhile, a consensus mechanism called proof of solution(PoSo) is designed for the optimization problem in Ref. 23 where the voting result of the optimization result is used as proof. It can be concluded that these consensus mechanisms have defects of different aspects in distributed computing when compared with the no proof-based consensus mechanism. For a clearer view of this, Tab. 3 is provided to visually demonstrate the advantages of the proposed method from different perspectives. Where, in the process of comparing tolerance for dishonest servers, we use *n* to represent the total number of servers participating in data storage. In addition, when comparing the timespan of the consensus process, we took the optimization scenario constructed in this paper as an example, storing its optimization results, and for the database storage method that does not require consensus and does not guarantee the reliability of results, corresponding timespans are left blank.

Tab. 3 Comparison between no-proof blockchain and other data storing structure

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Data storage structure | | Decentralization level | Tolerance for communication failure | Tolerance for dishonest servers | Timespan of the consensus process | Energy cost | Scalability | Examples |
| No-proof blockchain | | Semi-decentralized | Yes | n/2 | ≈0s | Little | Strong | This article |
| Centralized database | | Fully centralized | No | 0 |  | Little | Weak | [28] |
| Distributed database | | Decentralized | Yes | 0 |  | Little | Strong | [29] |
| Blockchain with extra proof in consensus mechanism | PoW | Decentralized | Yes | n/2 | ≈600s | High | Strong | [30] |
| DPoS | Semi-decentralized | No | n/2 | ≈3s | Low | Strong | [31] |
| PoA | Decentralized | Yes | n/2 | ≈6s | High | Strong | [32] |
| Raft | Semi-decentralized | Yes | 0 | ≈2.4s | Low | Strong | [33] |
| PBFT | Decentralized | Yes | (n-1)/3 | ≈30ms | Low | Weak | [34] |
| PoSo | Semi-centralized | Yes | n/2 | ≈19s | Low | Strong | [22] |

# Discussion

## Application of no proof-based consensus mechanism

In this paper, the researchers provide a no proof-based consensus mechanism alternative to the traditional multi-party consensus mechanism in the blockchain structure. In the traditional consensus mechanism, the servers rely on adding specific proof with nothing about the content to determine whether the content is trustworthy. In this consensus mechanism, it is noticed that the result of the distributed computation is verified to be valid in the computation process, thus there is no need to provide additional proof. Similar to the consensus mechanism proposed in this study, consensus mechanisms that avoid extra proof can be constructed in more scenarios where there are public constraints that can be verified by the servers independently. This is because public conditions can provide proof for the recorded content in these scenarios, and we call the problems in these scenarios “verifiable problems”.

No-proof consensus mechanism can be built with multi-party computation. This is because there must be a computation protocol among the participants and this protocol can be proof for the computation result. For example, in the application scenario in Ref. [21], blockchain is used to record the carbon emission flow of buses in the power system. In this scenario, all buses check whether the carbon emission flow to be uploaded on-chain satisfies the law of the power system, which is the computation protocol for all participants.

Moreover, this consensus mechanism can be used in computing definite solution problems. This is because there must be one or more definite conditions to determine the solution of these problems. These definite conditions are publicly known and can be the proof in the consensus mechanism. Servers can publicly and independently check whether the result is reliable. More application scenarios are listed in Tab. 4 to enhance the readers' comprehension, with detailed descriptions of whether they are verifiable or not, i.e. whether the result can server as proof or not.

Tab. 4 Verifiable problems and corresponding scenarios

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Application scenario | Scenario description | References | Verifiable or not | Decentralized or not | Verifiability description |
| Distributed mechine learning | Using blockchain to record training process so as to create a reliable decentralized machine learning platform. | [35],[36] | √ | √ | Training result can server as reliable proof for training contribution. |
| Distributed search engine | Using blockchain to record user searching information in a distributed search engine. | [37] | √ | √ | Search results that match the searching need are the proof when recording distributed search information. |
| Energy auction | Using blockchain to record trading in a decentralized double auction energy market. The constrictions in auction energy market can definite a trading result. | [38],[39] | √ | √ | Each server will verify whether the result meets the constraints in the auction market. |
| Carbon trading | Using blockchain to registor carbon trading record and manage carbon asset. | [40],[41] | √ | √ | Public statement from both side of the trading can be proof as the deal only needs their approval. |
| Transportation planning | Using blockchain to plan for the traffic schedule and record the schedule. | [42] | √ | × | Traffic condition can be the public proof of whether the traffic planning is successful or not. |
| Traffic events validation | Using blockchain to reliablily record traffic events. | [43] | √ | × | Eyewitness and scene evidence can be used as proof of a traffic accident. |
| [Digital-information tracking](https://www.academia.edu/download/60719565/120190927-101867-17o7qp.pdf) | Using blockchain to record personal information. | [44] | × | × | Personal information is known only to oneself and cannot be verified by others. |
| Real estate proof | Using blockchain to digitally record real estate so as to protect owners' rights. | [45] | × | × | Blockchain participants other than the head of the household cannot determine the authenticity of the real estate information. |

## Defect in privacy protection

There is also a defect in privacy protection in the NPC-blockchain structure proposed in the paper. While centralized computation means that one and only one participant can know all system parameters, distributed computation left space for privacy protection. This is because participants can use the encrypted data of others to complete computation and decrypt the result in some problems in distributed computation problems. Current studies have studied on using homomorphic cryptosystem to ensure privacy in blockchain system[4],[46]. For example, Ref. [21] provided a lattice-like blockchain structure to show data after homomorphic encryption instead of original data on-chain and decrypting data when obtaining it from the chain. Similarly, Ref. [47] designed a framework where the homomorphic encryption is used to hide the real amount of each transaction, and commitment proof is used for checking the validity of the encrypted amount. However, in these structures, original data is firstly encrypted and then computed linearly. Thus, constructing a privacy-protecting NPC-blockchain requires linearly processing the encrypted data so that the encrypted computation result can still fulfill constraints in the distributed computation problem and thus serve as proof in the no proof consensus mechanism.

Viewing in contrast, in some distributed computation problems (e.g. the distributed optimization problem which is used as an example in this paper), the lattice-like blockchain structure can’t work properly because the computation process contains non-linear steps, as the minimum value of the objective function is needed. If privacy protection is to be involved in NPC-blockchain, an encryption method that can ensure the result to fulfill constraints after encryption should be found, and in which participants cannot obtain data from others. Regretfully, current studies have not shown that such an approach exists, which means that privacy protection is not yet feasible for these specific distributed computation problems while applying NPC-blockchain.

# Conclusion

In this paper, a blockchain structure is proposed for the distributed computing problem. In the blockchain structure, a consensus mechanism is specially constructed without extra proof added, and an oracle is designed to obtain data off-chain to reduce the time and energy burden of the blockchain. The case study verifies that the proposed method can solve the distributed computing problem reliably and credibly. The proposed blockchain structure can record the distributed computation result on the blockchain successfully even with malicious attacks and unsteady network conditions. Also, the consensus mechanism designed in the proposed blockchain shows priority compared with existing consensus mechanisms being or can be used in the distributed computation field. However, the current method cannot guarantee the privacy of participants' data in the optimization process. The no-proof blockchain proposed is versatile and portable, its method of constructing a consensus mechanism can be applied to solving other verifiable problems.

Reference

1. Y. Wang, S. Wang, and L. Wu, “Distributed optimization approaches for emerging power systems operation: A review,” *Electric Power Systems Research,* vol. 144, pp. 127-135, 2017.
2. Y. Zheng, and Q. Liu, “A review of distributed optimization: Problems, models and algorithms,” *Neurocomputing,* vol. 483, pp. 446-459, 2022.
3. C. Ma, A. Li, Y. Du, H. Dong, and Y. Yang, “Efficient and scalable reinforcement learning for large-scale network control,” *Nature Machine Intelligence,* vol. 6, no. 9, pp. 1006-1020, 2024.
4. Q. Sun, H. Ma, T. Zhao, Y. Xin, and Q. Chen, “Break down the decentralization-security-privacy trilemma in management of distributed energy systems,” *Nature Communications,* vol. 15, no. 1, pp. 4508, 2024.
5. C. Zhao, J. He, and Q.-G. Wang, “Resilient distributed optimization algorithm against adversarial attacks,” *IEEE Transactions on Automatic Control,* vol. 65, no. 10, pp. 4308-4315, 2019.
6. N. Ravi, A. Scaglione, and A. Nedić, "A case of distributed optimization in adversarial environment." *ICASSP 2019*, pp. 5252-5256.
7. L. Thomas, Y. Zhou, C. Long, J. Wu, and N. Jenkins, “A general form of smart contract for decentralized energy systems management,” *Nature Energy,* vol. 4, no. 2, pp. 140-149, 2019.
8. S. Chen, Z. Shen, L. Zhang, Z. Yan, C. Li, N. Zhang, and J. Wu, “A trusted energy trading framework by marrying blockchain and optimization,” *Advances in Applied Energy,* vol. 2, pp. 100029, 2021.
9. A. A. Khan, A. A. Laghari, T. R. Gadekallu, Z. A. Shaikh, A. R. Javed, M. Rashid, V. V. Estrela, and A. Mikhaylov, “A drone-based data management and optimization using metaheuristic algorithms and blockchain smart contracts in a secure fog environment,” *Computers and Electrical Engineering,* vol. 102, pp. 108234, 2022.
10. P. Zheng, Q. Xu, X. Luo, Z. Zheng, W. Zheng, X. Chen, Z. Zhou, Y. Yan, and H. Zhang, “Aeolus: Distributed execution of permissioned blockchain transactions via state sharding,” *IEEE Transactions on Industrial Informatics,* vol. 18, no. 12, pp. 9227-9238, 2022.
11. C. Xu, C. Zhang, J. Xu, and J. Pei, “SlimChain: Scaling blockchain transactions through off-chain storage and parallel processing,” *Proceedings of the VLDB Endowment,* vol. 14, no. 11, pp. 2314-2326, 2021.
12. J. Liu, and J. Wu, “A Comprehensive Survey on Blockchain Technology and Its Applications,” *Highlights in Science, Engineering and Technology,* vol. 85, pp. 128-138, 2024.
13. S. Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System,” 2008.
14. F. Saleh, “Blockchain without waste: Proof-of-stake,” *The Review of financial studies,* vol. 34, no. 3, pp. 1156-1190, 2021.
15. S. M. S. Saad, and R. Z. R. M. Radzi, “Comparative review of the blockchain consensus algorithm between proof of stake (pos) and delegated proof of stake (dpos),” *International Journal of Innovative Computing,* vol. 10, no. 2, 2020.
16. E. Buchman, “Tendermint: Byzantine fault tolerance in the age of blockchains,” University of Guelph, 2016.
17. J. Mišić, V. B. Mišić, and X. Chang, “Design of proof-of-stake PBFT algorithm for IoT environments,” *IEEE Transactions on Vehicular Technology,* vol. 72, no. 2, pp. 2497-2510, 2022.
18. H. Qin, Y. Cheng, X. Ma, F. Li, and J. Abawajy, “Weighted Byzantine Fault Tolerance consensus algorithm for enhancing consortium blockchain efficiency and security,” *Journal of King Saud University-Computer and Information Sciences,* vol. 34, no. 10, pp. 8370-8379, 2022.
19. C. Yang, T. Wang, and K. Wang, “A consensus mechanism based on an improved genetic algorithm,” *Open Access Library Journal,* vol. 7, no. 9, pp. 1-6, 2020.
20. X. Yuan, F. Luo, M. Z. Haider, Z. Chen, and Y. Li, “Efficient Byzantine consensus mechanism based on reputation in IoT blockchain,” *Wireless Communications and Mobile Computing,* vol. 2021, no. 1, pp. 9952218, 2021.
21. Z. Han, T. Ding, C. Mu, X. Shen, J. Li, and M. Shahidehpour, “Lattice-Like Blockchain System for Maintaining the Data Privacy in the Distributed Computation of Carbon Emission,” *IEEE Transactions on Smart Grid*, 2024.
22. S. Chen, H. Mi, J. Ping, Z. Yan, Z. Shen, X. Liu, N. Zhang, Q. Xia, and C. Kang, “A blockchain consensus mechanism that uses Proof of Solution to optimize energy dispatch and trading,” *Nature Energy,* vol. 7, no. 6, pp. 495-502, 2022.
23. J. Wang, and N. Elia, "A control perspective for centralized and distributed convex optimization," *2011 50th IEEE conference on decision and control and European control conference*, pp. 3800-3805, 2011.
24. B. Okardi, and O. Asagba, “Overview of distributed database system,” *International Journal of Computer Techniques,* vol. 8, no. 1, pp. 83-100, 2021.
25. J.-P. Vergne, “Decentralized vs. distributed organization: Blockchain, machine learning and the future of the digital platform,” *Organization Theory,* vol. 1, no. 4, pp. 2631787720977052, 2020.
26. Y. Kang, J. Cho, and Y. B. Park, “An empirical study of a trustworthy cloud common data model using decentralized identifiers,” *Applied Sciences,* vol. 11, no. 19, pp. 8984, 2021
27. S. M. H. Bamakan, A. Motavali, and A. B. Bondarti, “A survey of blockchain consensus algorithms performance evaluation criteria,” *Expert Systems with Applications,* vol. 154, pp. 113385, 2020.
28. M. Nasar, and M. A. Kausar, “Suitability of influxdb database for iot applications,” *International Journal of Innovative Technology and Exploring Engineering,* vol. 8, no. 10, pp. 1850-1857, 2019.
29. T. Gkamas, V. Karaiskos, and S. Kontogiannis, “Performance evaluation of distributed database strategies using docker as a service for industrial iot data: Application to industry 4.0,” *Information,* vol. 13, no. 4, pp. 190, 2022.
30. L. Qi, J. Tian, M. Chai, and H. Cai, “LightPoW: A trust based time-constrained PoW for blockchain in internet of things,” *Computer Networks,* vol. 220, pp. 109480, 2023.
31. F. Yang, W. Zhou, Q. Wu, R. Long, N. N. Xiong, and M. Zhou, “Delegated proof of stake with downgrade: A secure and efficient blockchain consensus algorithm with downgrade mechanism,” *IEEE Access*, vol. 7, pp. 118541-118555, 2019.
32. Y. Zhang, W. Wang, and F. Shi, “Reputation-based Raft-Poa layered consensus protocol converging UAV network,” *Computer Networks,* vol. 240, pp. 110170, 2024.
33. E. Sakic, and W. Kellerer, “Response time and availability study of RAFT consensus in distributed SDN control plane,” *IEEE Transactions on Network and Service Managemen*t, vol. 15, no. 1, pp. 304-318, 2017.
34. W. Li, C. Feng, L. Zhang, H. Xu, B. Cao, and M. A. Imran, “A scalable multi-layer PBFT consensus for blockchain,” *IEEE Transactions on Parallel and Distributed Systems,* vol. 32, no. 5, pp. 1146-1160, 2020.
35. F. Zerka, V. Urovi, A. Vaidyanathan, S. Barakat, R. T. Leijenaar, S. Walsh, H. Gabrani-Juma, B. Miraglio, H. C. Woodruff, and M. Dumontier, “Blockchain for privacy preserving and trustworthy distributed machine learning in multicentric medical imaging (C-DistriM),” *Ieee Access,* vol. 8, pp. 183939-183951, 2020.
36. D. Li, D. Han, T.-H. Weng, Z. Zheng, H. Li, H. Liu, A. Castiglione, and K.-C. Li, “Blockchain for federated learning toward secure distributed machine learning systems: a systemic survey,” *Soft Computing,* vol. 26, no. 9, pp. 4423-4440, 2022.
37. S. Yu, C. Yeom, and Y. Won, “Implementation of search engine to minimize traffic using blockchain-based web usage history management system,” *Journal of Information Processing Systems,* vol. 17, no. 5, pp. 989-1003, 2021.
38. C. Mu, T. Ding, M. Shahidehpour, S. Liu, B. Chen, W. Jia, H. Ying, and Y. Huang, “A Light Blockchain for Behind-the-Meter Peer-to-Peer Energy Transactions in Cyber-Physical Power Systems,” IEEE Transactions on Smart Grid, vol. 15, no. 1, pp. 1063-1074, 2023.
39. M. Foti, and M. Vavalis, “Blockchain based uniform price double auctions for energy markets,” *Applied Energy,* vol. 254, pp. 113604, 2019.
40. S. Jiang, Y. Li, Q. Lu, Y. Hong, D. Guan, Y. Xiong, and S. Wang, “Policy assessments for the carbon emission flows and sustainability of Bitcoin blockchain operation in China,” *Nature communications,* vol. 12, no. 1, pp. 1-10, 2021.
41. Y. Pan, X. Zhang, Y. Wang, J. Yan, S. Zhou, G. Li, and J. Bao, “Application of blockchain in carbon trading,” *Energy Procedia,* vol. 158, pp. 4286-4291, 2019.
42. M. Haouari, M. Mhiri, M. El-Masri, and K. Al-Yafi, “A novel proof of useful work for a blockchain storing transportation transactions,” *Information Processing & Management,* vol. 59, no. 1, pp. 102749, 2022.
43. Y.-T. Yang, L.-D. Chou, C.-W. Tseng, F.-H. Tseng, and C.-C. Liu, “Blockchain-based traffic event validation and trust verification for VANETs,” *IEEE Access,* vol. 7, pp. 30868-30877, 2019.
44. A. Arora, and M. Arora, “Digital-information tracking framework using blockchain,” *Supply Chain Manag. Syst,* vol. 7, pp. 1-7, 2018.
45. H. P. Wouda, and R. Opdenakker, “Blockchain technology in commercial real estate transactions,” *Journal of property investment & Finance,* vol. 37, no. 6, pp. 570-579, 2019.
46. Q. Feng, D. He, S. Zeadally, M. K. Khan, and N. Kumar, “A survey on privacy protection in blockchain system,” *Journal of network and computer applications,* vol. 126, pp. 45-58, 2019.
47. Q. Wang, B. Qin, J. Hu, and F. Xiao, “Preserving transaction privacy in bitcoin,” *Future Generation Computer Systems,* vol. 107, pp. 793-804, 2020.