electronics

MDPI

Article

CE-PBFT: An Optimized PBFT Consensus Algorithm for Microgrid Power Trading

Xu Ding, Haihua Lu and Lanxian Cheng *

College of Electronic Engineering (College of Artificial Intelligence), South China Agricultural University, Guangzhou 510640, China; dingxu@stu.scau.edu.cn (X.D.); luhaihua@stu.scau.edu.cn (H.L.)
* Correspondence: chenglx@scau.edu.cn

Abstract: Currently, in the blockchain-based distributed microgrid trading system, there are some problems, such as low throughput, high delay, and a high communication overhead. To this end, an improved Practical Byzantine Fault Tolerance (PBFT) algorithm (CE-PBFT) suitable for microgrid power trading is proposed. First, a node credit value calculation model is introduced, and nodes are divided into consensus, supervisory, and propagation nodes according to their credit values, forming a hierarchical network structure to ensure the efficiency and reliability of consensus. Secondly, the consensus process is optimized by adopting a segmented consensus mechanism. This approach calculates the consensus rounds for nodes and selects the methods for node-type switching and consensus based on these calculations, reaching dynamic changes in node states and credit values, effectively reducing the communication overhead of node consensus. Finally, the experiments show that compared with the IMPBFT and PBFT algorithms, the CE-PBFT algorithm has better performance in throughput, delay, and communication overhead, with a 22% higher average throughput and 15% lower average delay than the IMPBFT algorithm and a 118% higher average throughput and 87% lower average delay than the PBFT algorithm.

Keywords: blockchain; microgrid power trading; PBFT algorithm; consensus mechanism; credit value

Citation: Ding, X.; Lu, H.; Cheng, L. CE-PBFT: An Optimized PBFT Consensus Algorithm for Microgrid Power Trading. *Electronics* **2024**, *13*, 1942. https://doi.org/10.3390/electronics13101942

Academic Editors: Ahmed Abu-Siada and Jianguo Zhu

Received: 6 April 2024
Revised: 1 May 2024
Accepted: 14 May 2024
Published: 15 May 2024

1. Introduction

Currently, traditional microgrid power trading mainly adopts a centralized trading model, where the data are uploaded to a centralized processing system [1]. However, this transaction mode is not applicable to microgrid power transactions, and there are problems such as high cost, the vulnerability of the transaction center to hacking, and the inability to ensure the security of the transaction and the privacy of the users [2]. Therefore, new technologies are needed to achieve the transformation from centralized to distributed transaction models to cope with the microgrid power trading market [3,4]. Blockchain, as an emerging decentralized technology, can effectively solve the problems of the traditional centralized transaction model, such as the vulnerability to attacks and inability to guarantee user privacy, by virtue of its features, such as non-tampering, anonymity, traceability, and autonomy [5–8]. However, it is known from the existing research that a microgrid power trading system using blockchain technology has problems such as low throughput and high delay, which cannot meet the demands of real-time trading [9,10]. The key to the performance of the energy transaction system lies in the consensus algorithm; hence, it is necessary to improve the consensus algorithm to enhance the performance of the system, to meet the user's interaction needs and increase the application scenarios of blockchain.

Among the commonly used consensus algorithms, the classic ones include PoS [11], PoW [12], PBFT [13], Raft [14], Paxos [15], and so on. Different algorithms have their own advantages and disadvantages. Among them, the PBFT algorithm is widely used because of its ability to solve the Byzantine General problem; however, it also suffers from high delay, low performance, and low throughput [16]. Therefore, many scholars

have improved the PBFT algorithm in terms of its consistency protocol [17] and consensus strategy [18]. The authors of [19] proposed to establish a trusted list to improve the flexibility of nodes and to adopt a credit value mechanism to optimize the PBFT consensus process. This scheme improved the performance of the blockchain system; however, there was a cumulative situation of node credibility leading to a high degree of centralization in the system. The authors of [20] proposed the use of credit value to divide nodes, which could dynamically change the node status as well as automatically detect invalid and malicious nodes; however, the scheme did not consider the impact of the interactions between nodes on the consensus. The authors of [21] proposed a consensus mechanism based on proof-of-credit (PoCS) to solve the problem of credit security in microgrids; however, the method focused too much on node credit security and ignored the consensus efficiency, which cannot satisfy the demand for real-time microgrid power transactions. The authors of [22] proposed a performance-optimized consensus mechanism based on node classification to accelerate the consensus efficiency by reducing the number of nodes participating in the consensus; however, the scheme frequently changed the node view, and this affected the consensus speed.

In summary, this paper proposes an improved PBFT algorithm with real-time supervision to improve the performance of a microgrid power trading system; the main improvements include the following: (1) designing a dynamic node usage evaluation mechanism to classify nodes based on credit to ensure the reliability of consensus; (2) forming a hierarchical network structure by classifying nodes to reduce the number of nodes participating in the consensus and adopting a segmented consensus mechanism to improve the efficiency of the consensus; and (3) simplifying the pre-preparation and feedback phases of the consensus process to reduce the communication overhead, improving the efficiency of the consensus.

2. Overview of PBFT Consensus Algorithm

The PBFT is a state machine copy replication algorithm [23], where, under the same input parameters and state, any node has the same execution result. It can guarantee the normal work of the system when the number of Byzantine nodes is not more than one-third. The PBFT algorithm is based on the view replacement, checkpointing, and consistency protocols to reach consensus. Among them, the consistency protocol is the key to reaching consensus in the PBFT algorithm, which divides the consensus process into five stages, as shown in Figure 1.

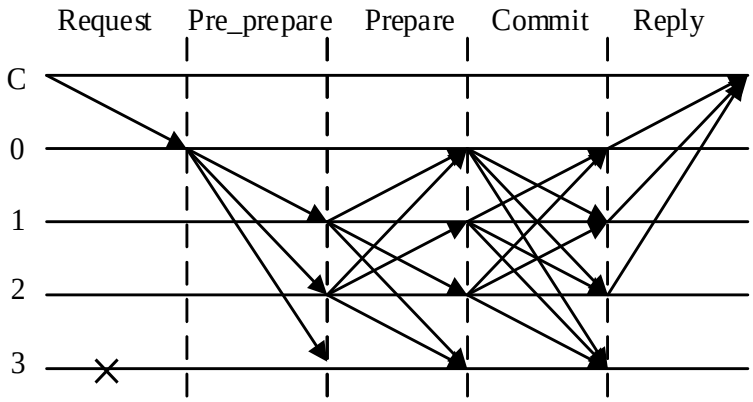

Figure 1. PBFT conformance protocol process.

In general, the PBFT consensus algorithm consists of five phases, which are request, pre-prepare, prepare, commit, and reply.

(1) Request phase: The client sends a request message $\langle request, m, t, c \rangle$, where m denotes the request message sent by the client, t denotes the timestamp, and c denotes the client's identity.

(2) Pre-prepare phase: The master node receives the request, generates a pre-prepare message $\langle\langle PRE-PREPARE, v, n, d\rangle, m\rangle$ and broadcasts it to the rest of the nodes, where v denotes the current view number, n denotes the number of the request message m, and d denotes the content summary of the request message m.

(3) Prepare phase: The general node receives the message sent by the master node and verifies it, then generates the preparation message $\langle PREPARE, v, n, d, i\rangle$ and broadcasts it to the rest of the nodes, where i is the number of the local node.

(4) Commit phase: Each node verifies the message, generates an acknowledgement message $\langle COMMIT, v, n, D(m), i\rangle$ and broadcasts it to other nodes. The nodes examine the received acknowledgement messages. When the node receives the acknowledgement message from at least $2f + 1$ different nodes, it completes the commit phase.

(5) Reply phase: After completing the confirmation, each node generates a feedback message $\langle REPLY, v, t, c, i, r\rangle$ and broadcasts it to the client, where r indicates the result of the node's execution of the client's request. The client completes the request after receiving the same feedback message from at least $2f + 1$ different nodes.

3. Design of the CE-PBFT Algorithm

Distributed technologies are frequently used in microgrids [24]; the PBFT is a highly suitable blockchain consensus algorithm for microgrid transactions [25], offering several advantages, including the provision of both real-time and eventual consistency, adaptability to high-frequency small-scale transactions, high security, and resilience against Byzantine faults. These features enhance the trustworthiness and resistance to attacks of microgrid transaction systems. However, it should be noted that as the number of nodes within the microgrid increases, the communication overhead of the PBFT also significantly escalates, because the communication complexity of the PBFT algorithm is O(n2). This may result in the low throughput or high latency of the blockchain system. Therefore, in order to enhance the system performance in the microgrid trading system, based on the PBFT algorithm, combined with the credit evaluation [26] and pipeline mechanisms, the CE-PBFT algorithm model is designed, as shown in Figure 2, which contains two main parts; one is the node credit calculation and classification, and the other is the consensus process of the CE-PBFT algorithm. The details are as follows:

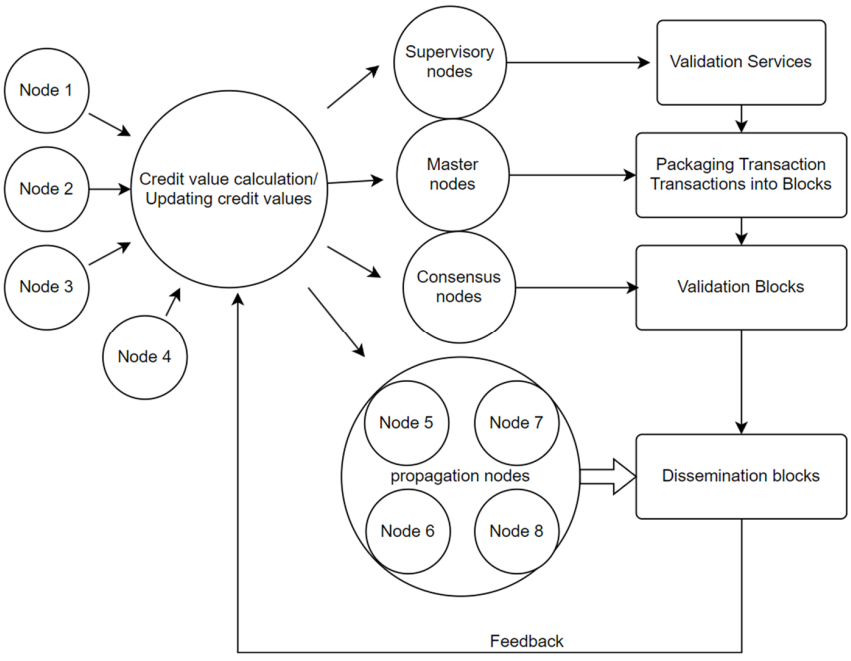

Figure 2. CE-PBFT algorithm consensus workflows.

3.1. Node Credit Value Calculation and Classification

The node credit calculation and categorization can be divided into three steps, as shown in Figure 2.

(1) Calculate the credit value of each node.
(2) Based on the calculated credit value of the nodes, categorize the nodes into three types: consensus, supervisory, and propagation. Subsequently, randomly select an eligible node from the consensus nodes to act as the master node. At the end of each consensus, the supervising node calculates and updates the credit value of each node based on the reputation value formula.
(3) Update the node roles based on the new node credit values.

In the proposed consensus model of the CE-PBFT algorithm, the nodes are divided into consensus, supervisory, and propagation nodes according to their credit value, thus constituting a hierarchical network structure with three networks: the consensus, supervisory, and propagation groups. Thus, it avoids all nodes participating in the consensus during the consensus process and simplifies the two-by-two interaction of nodes originally required for the interaction between other nodes and the master node, which reduces the communication overhead, shortens the time required for consensus, and improves the efficiency of the consensus. In addition, to ensure the reliability of the node, the credit value and role of the node will change with the behavior of the node. If a node behaves maliciously, the credit value will decrease; when it behaves well, it will increase. The supervisory node group will regularly detect the status of the nodes in the network and deal with the malicious nodes in the network in a timely manner after reaching an internal agreement.

3.1.1. Node Division

- Supervisory nodes.

The Supervisory node (C for short) is the hub of the blockchain network, and each supervisory node forms a supervisory group responsible for monitoring the whole network; its specific responsibilities are as follows:

(1) For a newly registered node, the supervisory node will first verify the legitimacy of the node and then randomly select a value in the range of (15, 25) as the initial credit value of the node. The new node will then join the blockchain network as a propagation node.
(2) The supervisory nodes will periodically check the nodes in the network to determine whether the nodes are working properly and deal with the malfunctioning nodes in a timely manner.
(3) According to the credit value formula, it calculates the credit value of each node and divides the nodes based on the credit value, and then randomly selects an eligible node from the consensus node group to act as the master node.
(4) After the supervisory node completes the update of its own state, it verifies whether the states of each other node agree, and if so, it notifies the master node to open the next round of consensus.

- Consensus nodes.

The consensus node (CN for short) is a node where the supervising node selects the top 50% of nodes based on the descending order of the credit value. Each consensus node constitutes a consensus group, and in the consensus process, its main responsibility is to verify the received transaction information and feed the result back to the master node. Meanwhile, the number of CNs should be at least three times the number of Byzantine nodes in the pool of consensus nodes.

- Master nodes.

The master node is a node that is randomly selected from the consensus nodes by the supervisory node, which writes the verified power transaction to a new block and collects and counts the feedback results from other nodes.

- Propagation Nodes.

Propagation nodes (SN for short) are consensus nodes and supervisory nodes with credit values lower than that of the propagation nodes; each propagation node constitutes a propagation group that is responsible for storing block data, propagating verified blocks, and sending preliminarily verified transactions to the master node.

3.1.2. Calculation of Node Credits

The node credit value is a comprehensive evaluation of the performance of nodes in a blockchain network. Its calculation contains two key components: incentives and penalties.

Incentive component: Nodes that contribute to the blockchain network are rewarded based on their contributions; timely and successful processing of transactions will increase the node's credit value.

Penalty component: Nodes in the blockchain network will be punished according to their bad behavior, and nodes will have their credit value reduced due to their unsuccessful processing of transactions or node failures; nodes that are actively malicious will have their credit value changed to 0 and will be kicked out of the blockchain transaction network.

As a result, a new credit value calculation method is proposed to apply to microgrid power transactions. The specific formula is as follows:

$$\begin{cases} CV(i) = R + \frac{100N(i) \times W_1}{N_{\max}} + 5W_2S(i) - 10W_3F(i) & E(i) \leq 0 \\ CV(i) = 0 & E(i) > 0 \end{cases} \quad (1)$$

where $\boldsymbol{CV(i)}$ denotes the credit value of the node; $\boldsymbol{R}$ denotes the credit value of the node in the last round, where the newly registered node is the initial credit value assigned by the system, and for other nodes, it is the credit value from the previous round of consensus; $\boldsymbol{W_1}$, $\boldsymbol{W_2}$, $\boldsymbol{W_3}$ denote the weight parameters in the process of calculating the credit value of the node, which is generally taken to be $\boldsymbol{W_1 = 0.65}$, $\boldsymbol{W_2 = 0.2}$, $\boldsymbol{W_3 = 0.25}$, used to ensure that more penalties are earned when a transaction fails to be processed in order to encourage more successful transactions [27]; $\boldsymbol{N(i)}$ denotes the turnover power of the node $\boldsymbol{i}$; $\boldsymbol{N_{max}}$ denotes the maximum turnover power of the node in the blockchain network; $\boldsymbol{S(i)}$ denotes the number of times that the node has successfully processed the transaction; $\boldsymbol{F(i)}$ denotes the number of times that the node has malfunctioned as well as the number of times that it failed to process the transaction; and $\boldsymbol{E(i)}$ denotes the number of times that the node has actively been malicious. The credit value will be zeroed out when a node behaves badly, and considering the possibility of data transmission errors, no penalty is applied when $\boldsymbol{E(i) < 0}$.

3.1.3. Update of the Node Status

The node state change is mainly based on the change in the node credit value, which makes the state of the consensus, supervisory, and propagation nodes change, as shown in Figure 3.

(1) Changes in consensus nodes

For a consensus node to be transformed into a supervisory node, it needs to fulfil the following conditions:

$$f\max(SN) < CV(CN) < f\min(C) \quad (2)$$

where $\boldsymbol{CV(CN)}$ denotes the credit value in the last round of the consensus node $\boldsymbol{CN}$ update, $\boldsymbol{f}$ max($\boldsymbol{SN}$) denotes the maximum credit value of the propagation node, and $\boldsymbol{f}$ min($\boldsymbol{C}$) denotes the minimum credit value of the supervisory node.

For a consensus node to be transformed into a propagation node, it needs to fulfill the following conditions:

$$CV(CN) < f\max(SN) \quad (3)$$

For a consensus node to be transformed into a master node, it needs to fulfill the following condition:

$$Random(\{CN\})$$

where ***Random*({*SN*})** denotes taking a random node for the set ***SN*** of propagating nodes. When a consensus node's updated credit value is between the maximum credit value of the propagation node and the minimum credit value of the supervisory node, it can become a supervisory node. When it is less than the minimum credit value of the consensus node, it becomes a propagation node. It can only become one by random selection.

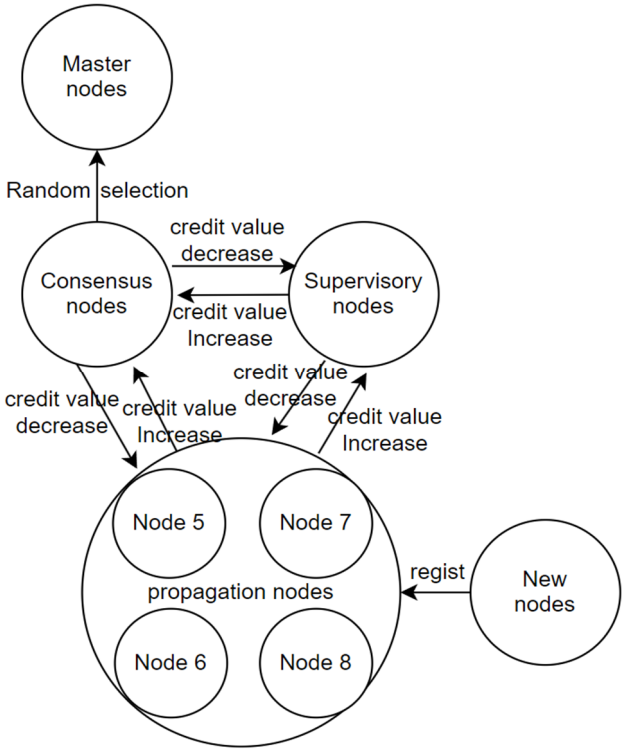

Figure 3. Node state transition diagram.

(2) Changes in supervisory nodes

For a supervisory node to be transformed into a consensus node, it needs to fulfill the following condition:

$$CV(C) > f\max(C) \tag{4}$$

For a consensus node to be transformed into a propagation node, it needs to fulfill the following condition:

$$CV(C) < f\min(C) \tag{5}$$

(3) Changes of propagation nodes

For a propagation node to be transformed into a consensus node, it needs to fulfill the following condition:

$$CV(SN) > f\max(C) \tag{6}$$

For a propagation node to be transformed into a supervisory node, it needs to fulfill the following condition:

$$f\max(SN) < CV(SN) < f\min(C) \tag{7}$$

(4) Changes in new registration nodes

When a new registered node (RN for short) enters the network, it is assigned a credit value $\boldsymbol{b}$ by the system, which needs to be satisfied to be less than or equal to the maximum value of the credit value of the propagating node. The formula is expressed as follows:

$$\begin{cases} CV(RN) = b \\ CV(RN) \leq f\max(SN) \end{cases} \tag{8}$$

The node state change algorithm design process is as follows (Algorithm 1):

Algorithm 1 Node state change

Input:
Node sample
Output:
None
1. If Node's state == None&& b ≤ SN_max
2. Node's credit value = b;
3. Node's state = SN;
4. Return;
5. Else if Node's state == SN
6. If Node's credit value > C_max
7. Node's state = CN;
8. Return;
9. Else if Node's credit value > SN_max&&Node's credit value < C_min
10. Node's state = C;
11. Return;
12. Else if Node's state == C
13. If Node's credit value > C_max
14. Node's state = CN;
15. Return;
16. Else if Node's credit value < C_min
17. Node's state = SN;
18. Return;
19. Else if Node's state = CN
20. If Node's credit value > SN_max&& Node's credit value < C_min
21. Node's state = C;
22. Return;
23. If Node's credit value < SN_max
24. Node's state = SN;
25. Return;
26. Else
27. Return;

3.2. Consensus Process of the CE-PBFT Algorithm

The improved PBFT algorithm uses four types of nodes to optimize the consistency process as follows:

(1) The supervisory node first checks the legitimacy of the new node, and if it is legitimate, it assigns a credit value to it.

(2) The supervisory node will check the consensus round $\boldsymbol{t}$ to decide on the next consensus operation and it will set $\boldsymbol{t}$ to 3 to balance the efficiency and reliability of the algorithm. If the consensus round $\boldsymbol{t}$ is 3, the credit value of each node will be calculated by the credit value formula and then the consensus node, supervisory node, and propagation node will be divided according to the credit value, and at the same time, the master node will be selected from among the consensus nodes, and $\boldsymbol{t}$ will be set to 0. Otherwise, it will be carried out only for the computation of the node's credit value and the update.

(3) The supervisory node and the propagation node will initially validate the format of the received power transaction transactions and send the validated power transaction transactions to the master node.

(4) The master node collects a certain number of transactional transactions and packages them to generate a prep block $\langle\langle Pre-Prepare, tx, hr, t, p, w\rangle, D_{pr}\rangle$, where $\boldsymbol{tx}$ denotes the transactional data, $\boldsymbol{hr}$ denotes the hash of the parent block of that prep block, $\boldsymbol{t}$ denotes the timestamp of the generation of the prep block, $\boldsymbol{p}$ denotes the node sequence number, $\boldsymbol{w}$ denotes the node credit value, and $\boldsymbol{D_{pr}}$ denotes the generated prep block. At this point, the pre-preparation phase is entered.

(5) The master node broadcasts the generated preparatory block to the consensus node, and the consensus node verifies the accepted block, verifies whether the hash $\boldsymbol{hr}$ of the parent block in the prep block is equal to the hash of the highest block of the current consensus node, and whether the transaction information of the block is legal and valid, and sends the voting result $\langle feedback, tx, hr, t, rs\rangle$ back to the master node and enters into the feedback phase, where $\boldsymbol{rs}$ is the voting result. The master node collects and counts the feedback information $\boldsymbol{V_T}$.

$$\begin{cases} V_T \geq \frac{2(n-1)}{3} + Num(S) \\ V_T < \frac{2(n-1)}{3} + Num(F) \end{cases} \tag{9}$$

In the previous equation, $\boldsymbol{n}$ denotes the total number of consensus nodes in the network, ***Num(S)*** represents the number of nodes with successful confirmation, and ***Num(F)*** represents the number of nodes with failed confirmation.

(6) The master node generates a confirmation message $\langle\langle confirm, tx, hr, t, p, w\rangle, D_r\rangle$ based on the feedback results and broadcasts it to the consensus node, and at the same time enters the confirmation phase, in which the master node generates a formal block $\boldsymbol{D_r}$ based on the hash of the reserve block. The consensus node receives the official block hash and the hash of the local reserve block for comparison, confirms the storage, and returns its own state to the master node. The master node collects the statistics and verifies it according to the judgment conditions in the previous step, and if the verification passes, it broadcasts the confirmation block to the remaining nodes. After receiving it, the propagation node and the supervisory node store the confirmation block directly and send the confirmation message to the master node.

(7) The master node, while waiting for the feedback information from the propagation nodes and the supervisory nodes, will check whether the consensus round $\boldsymbol{t}$ is 3. If so, it will set it to 0, recount it and continue to wait; otherwise, the master node will collect the transaction transaction packaged into chunks and verify it with the consensus node, and will broadcast the result of this time to the supervisory nodes and propagation nodes only after both of them have confirmed the transaction in the previous round.

(8) The master node counts the feedback and after confirming the consensus, it will send a message to notify the supervisory node, which will calculate and update the credit value of each node and open the next round of consensus at the same time.

In addition, the failure of any of the above steps in the consensus process results in the failure of the consensus for that round. The unconfirmed transactions in that round are then put into the next round to continue the consensus.

Based on the above consensus process, Figure 4 shows the flowchart of the consensus process of the proposed CE-PBFT algorithm.

The CE-PBFT algorithm consensus process is as follows (Algorithm 2):

Algorithm 2 CE-PBFT consensus algorithm

Input:
New Node sample (as N)
Initial parameters
Output:
None
1. If New Node is legal
2. Node state change(N);
3. Consensus round(t) = 3;
4. Create transaction queue $\boldsymbol{Q_1}$;
5. where consensus round(t) < 3
6. Format validation of generated power transactions by propagation nodes;
7. Supervisory node secondary validation;
8. Transaction enters the queue $\boldsymbol{Q_1}$;
9. The master node collects a certain number of transactions from $\boldsymbol{Q_1}$ and packs them into prep block $\langle\langle Pre-Prepare, tx, hr, t, p, w\rangle, D_{pr}\rangle$;
10. The master node broadcasts prep block to the consensus nodes;
11. The consensus nodes verify:
12. If prep block's parent Hash == highest block's Hash
13. Return master node information: Verify Successfully;
14. Else
15. Return master node information: Verify failed;
16. Master node collects feedback information (V_{T0}):
17. If $V_{T0} \geq 2 \times (n-1)/3$ + number of Verify Successfully;
18. Broadcast to other nodes;
19. Else
20. t+=1;
21. Continue;
22. If The nodes receive the confirmation block:
23. Synchronize to their own local to complete the consistency process;
24. Feedback to master node;
25. Else
26. t+=1;
27. Continue;
28. If all Supervisory and propagation nodes confirm:
29. Master node broadcast results;
30. Else
31. t+=1;
32. Continue;
33. Supervisory nodes calculate the credit value of each node;
34. If t == 3
35. Node state change (each node);
36. Random(C) become Master node;
37. t = 0;
38. End

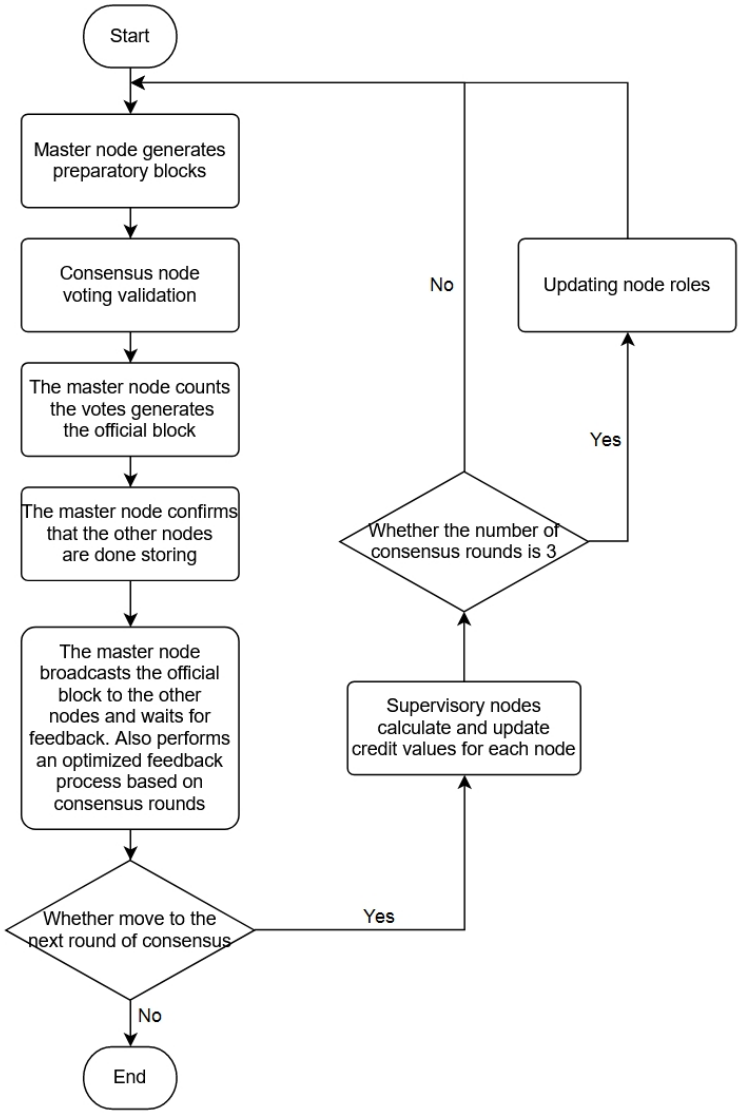

Figure 4. CE-PBFT algorithm consensus flow diagram.

4. Analysis and Simulation Experiment

4.1. Simulation Experiment Environment

In a blockchain-based microgrid power trading system, the consensus algorithm's throughput, delay, and communication overhead are important; in order to test the optimization effect of the CE-PBFT algorithm, the performance of the IMPBFT [28] and PBFT algorithms were compared and analyzed, by experimentally setting up different block sizes and different numbers of nodes, in three respects: communication overhead, delay, and throughput.

In addition, in the simulation experiment, the identity of each node was randomized; however, in a consensus round, a node can only be a seller/buyer of electricity and cannot have both identities, and the trading power generated in the simulation experiments was generated by a random function. Considering that the transaction power data in the actual production are not easily obtained directly by the computer, while, in a practical application, the transaction power data can be realized using a smart meter [29], the simulation experiments in this paper focused only on the period after data are obtained.

Hyperledger Fabric provides hot-plugged consensus algorithm modules, and we replaced the PBFT module with the CE-PBFT algorithm [25,30]. The CE-PBFT, IMPBFT, and PBFT algorithms were deployed to Hyperledger Fabric in local physical machine servers

for the simulation and evaluation of their performance. The detailed server configuration is shown in Table 1.

Table 1. Server configuration.

Type	Version/Data
CPU Clock Speed	3.70 Ghz
Memory	32 GB
Network Bandwidth	50 Mpbs
Operating System	Ubuntu v18.04
Hyperledger Fabric	v0.6
Docker	v1.8.2
Golang	V1.16

The Hyperledger Fabric deployment and the three consensus algorithms are implemented using Golang programming. We plan to show the performance of the three algorithms by varying the block size and the number of nodes in the system observing the throughput and delay changes of each of the three algorithms and also discussing their communication overheads by calculating the number of communications.

4.2. Impact of Block Size on Throughput

In blockchain architectures, the metrics of throughput and latency are pivotal in determining system performance. Throughput (TPS), defined as the aggregate number of transactions that the system is capable of processing per unit time, serves as a critical parameter for assessing the efficacy of a blockchain network. This metric is generally defined as follows:

$$TPS = \frac{TN_{\Delta t}}{\Delta t} \tag{10}$$

In this equation, the symbol $\boldsymbol{\Delta t}$ denotes the time interval during which the block remains open, and $\boldsymbol{TN_{\Delta t}}$ represents the number of transactions encapsulated within the block throughout the duration of $\boldsymbol{\Delta t}$.

The experiment focused on evaluating the variability in the throughput for the CE-PBFT, IMPBFT, and PBFT algorithms under conditions of varying block sizes of 500, 1000, 1500, 2000, 2500, and 3000 transactions, respectively, while maintaining a constant node count of seven. The empirical results represented the mean values derived from ten independent experiments. These results were subsequently utilized to construct a comparative analysis of the throughput for the CE-PBFT, IMPBFT, and PBFT, as illustrated in Figure 5.

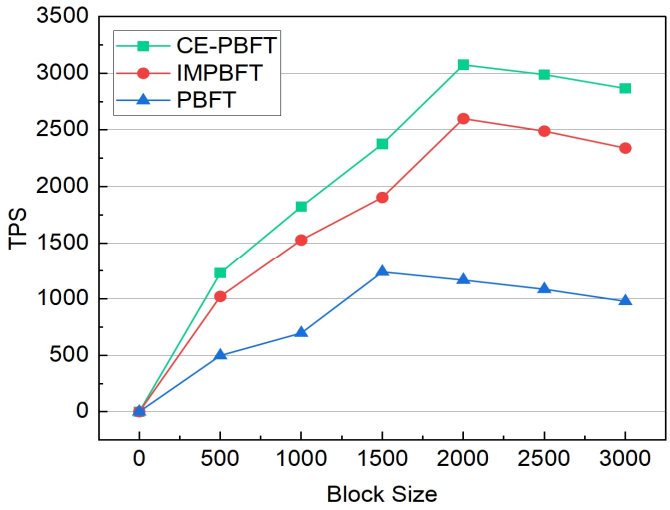

Figure 5. TPS comparison chart for different block sizes.

As depicted in Figure 5, the CE-PBFT algorithm achieved a peak throughput of approximately 3074 transactions, significantly surpassing the peak throughputs of the IMPBFT and PBFT algorithms, which were around 2600 and 1240 transactions, respectively. Notably, the throughput for all three algorithms initially increased with the increase in the number of transactions per block, followed by a subsequent decline. A comparative analysis revealed that the average throughput of the CE-PBFT algorithm was 21% higher than that of the IMPBFT and 120% higher than that of the PBFT. This enhancement can be attributed to the optimized consensus mechanism within the CE-PBFT, which allows nodes to more efficiently pack and verify transactions according to their identities, thereby accelerating the transaction processing and enhancing the consensus speed among the nodes.

Furthermore, the experimental observations suggested a correlation between the throughput and block size. Specifically, when the block size remained below a certain threshold, an increase in the block size resulted in a larger number of transactions processed per unit time, thus enhancing the throughput. However, surpassing this threshold led to an inability of the nodes to process transactions in a timely manner, causing thread blocking and a subsequent decrease in the throughput.

4.3. Impact of Block Size on Transaction Delay

In blockchain architectures, the transaction delay is a critical metric that encapsulates the consensus efficiency and the system's overall communication performance. The transaction delay, as investigated in this study, denotes the duration from the moment a node initiates the first transaction request until the system achieves consensus. This interval was quantitatively assessed using the following mathematical formulation:

$$Delay = T_c - T_f \tag{11}$$

In the above formula, ***Delay*** signifies the system's latency, while $\boldsymbol{T_c}$ represents the temporal point at which consensus is achieved within the system, and $\boldsymbol{T_f}$ indicates the moment when the node dispatches the initial transaction request. To explore the influence of the block size on the transaction delay, this study incrementally adjusted the number of transactions per block from 500 to 3000, maintaining a constant seven nodes within the blockchain network.

The resultant transaction delays were rigorously measured across 10 experimental iterations to ascertain a robust average, which was subsequently presented as the definitive value. Figure 6 elucidates the comparative delay across three distinct algorithms, highlighting their performance characteristics in relation to block size variations.

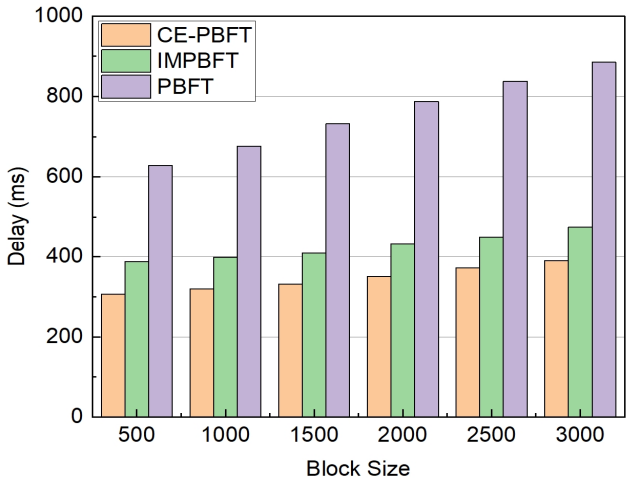

Figure 6. Delay comparison chart for different block sizes.

As delineated in Figure 6, there was a discernible increase in the delay for all three algorithms as the number of transactions per block increased. Notably, the delay associated with the CE-PBFT algorithm consistently remained lower than that observed with the IMPBFT and PBFT algorithms. Quantitatively, the average delay of the CE-PBFT algorithm was 19.2% lower than that of the IMPBFT and 54% lower than the PBFT algorithm. This enhanced performance is primarily attributable to the CE-PBFT algorithm's reduction in inter-node communication and the acceleration of the consensus process through a streamlined protocol.

Additionally, the experimental findings indicated that larger blocks, which encompassed a greater number of transactions, required more time for the propagation and verification of data among nodes. This process was further constrained by network bandwidth limitations, resulting in increased system latency.

4.4. Impact of Number of Nodes on Throughput and Delay

The scalability of blockchain systems, particularly as it pertains to the number of nodes, significantly impacts system performance metrics such as the transaction throughput and latency. So, we explored these dynamics by varying the node count and observing the resultant system throughput per second (TPS) and latency under a load of 500 transactions per block.

As illustrated in Figures 7 and 8, an increase in the node count correlated with a reduction in the throughput and an escalation in the latency across all three evaluated algorithms. Notably, the CE-PBFT algorithm consistently outperformed the IMPBFT and PBFT algorithms in both the throughput and latency metrics.

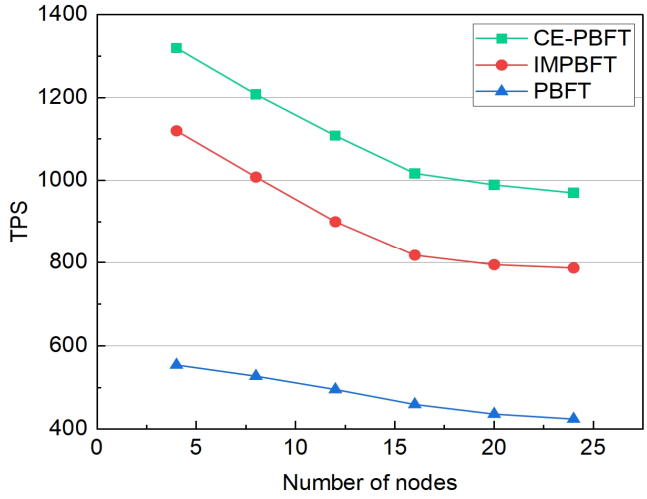

Figure 7. TPS comparison chart for different number of nodes.

Specifically, the CE-PBFT algorithm demonstrated an average throughput that was 22% superior to that of the IMPBFT and 118% greater than that of the PBFT. Furthermore, it exhibited an average latency that was 15% lower than the IMPBFT and 87% lower than the PBFT.

This performance advantage is primarily attributed to the increasing complexity of the node-to-node communications required for consensus as the network expands and the number of nodes increases. Unlike the standard PBFT and IMPBFT algorithms, the CE-PBFT introduces an enhanced consensus mechanism that minimizes the likelihood of Byzantine nodes assuming leadership roles, thereby reducing the frequency of view changes and the overall number of nodes engaged in the consensus process. This optimization not only diminished the latency but also enhanced the throughput, particularly evident in larger networks where the CE-PBFT algorithm sustained significantly lower communication delays due to its O(n) communication overhead, in stark contrast to the $O(n^2)$ communication complexity characteristic of the PBFT algorithm.

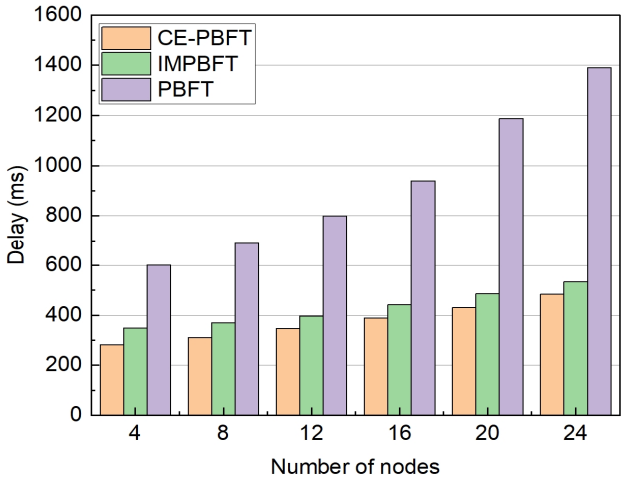

Figure 8. Delay comparison chart for different number of nodes.

4.5. Network Communication Overhead

The communication overhead refers to the total number of communications required for nodes to achieve consensus within a network.

Consider a blockchain network composed of n nodes, categorized into m consensus, p supervisory, and q propagation nodes. The aggregate number of communications necessary for achieving consensus can be derived from the CE-PBFT algorithm's consensus process:

$$\begin{cases} D(CE-PBFT) = 4m + 2(p+q) - 4 \\ m = 0.5n, p = 0.2n, q = 0.3n \\ \Rightarrow D(CE-PBFT) = 3n - 4 (n \geq 2) \end{cases} \tag{12}$$

The number of communications in a consensus round for the PBFT algorithm is:

$$D(PBFT) = n(n-1) + (n-1)^2 + n - 1 = 2n^2 - 2n \tag{13}$$

A comparative analysis of the communication counts between the CE-PBFT algorithm and the traditional PBFT algorithm was conducted. The results are illustrated in Figure 9, which depicts the communication counts for both algorithms across various node counts.

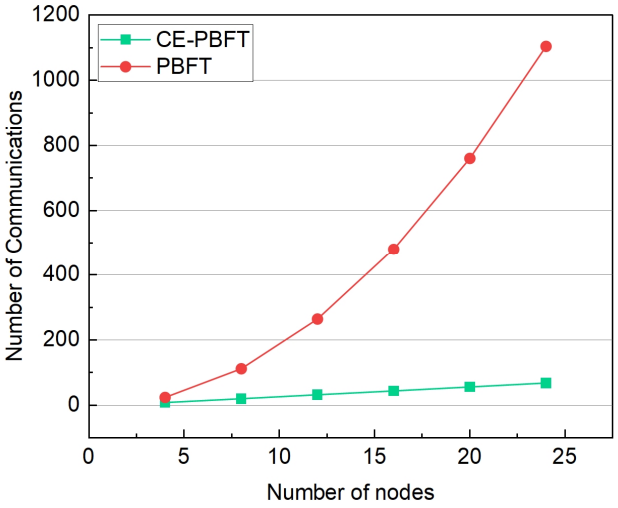

Figure 9. Comparison of the number of communications between PBFT and CE-PBFT algorithms.

As depicted, both algorithms exhibited an increase in communication counts as the number of nodes increased. However, the CE-PBFT algorithm consistently demonstrated

a lower number of communications compared to the PBFT algorithm. This reduction is attributable to the CE-PBFT algorithm's ability to decrease the exponential growth of the PBFT communications to a polynomial level, thereby minimizing both the number of communications and the overall communication overhead. The CE-PBFT algorithm, as proposed in this study, effectively reduces the number of nodes involved in the consensus process and simplifies the consensus mechanism.

5. Conclusions and Outlook

In this article, we proposed the CE-PBFT algorithm, based on the dynamic credit value, to address the problems of low throughput and high latency in the microgrid power trading system, using blockchain technology. The algorithm categorizes nodes by credit value, which reduces the number of nodes participating in consensus and the time required to reach consensus, while ensuring the reliability of the consensus nodes. Meanwhile, the consensus process is optimized to reduce the communication overhead and improve the operation efficiency. In addition, the feasibility of the proposed CE-PBFT algorithm was proved through experiments; in the above cases, compared with the IMPBFT algorithm, the CE-PBFT algorithm had a 21.5% higher average throughput and 17.1% lower latency; compared with the PBFT algorithm, the CE-PBFT algorithm had a 119% higher average throughput and 70.5% lower latency, and the latency was much lower than that of the PBFT algorithm, especially when the number of nodes was large. However, the CE-PBFT algorithm still has deficiencies; for example, how the marginal costs incurred in the actual transaction process are incorporated and the actual energy consumption of the operation of this algorithm are worth discussing, and these will comprise our future research.

Author Contributions: Conceptualization, X.D. and L.C.; methodology, X.D.; software, X.D.; validation, X.D. and L.C.; formal analysis, X.D.; investigation, X.D.; resources, X.D.; data curation, X.D. and H.L.; writing—original draft preparation, X.D.; writing—review and editing, X.D.; visualization, X.D. and H.L.; supervision, L.C.; project administration, X.D.; funding acquisition, L.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China. (Project Grant Number: 61804057).

Data Availability Statement: The data used to support the finding of this study are included within the article.

Acknowledgments: The authors would like to thank the editors and the reviewers for their valuable time and constructive comments.

Conflicts of Interest: The authors declare no conflicts of interest.

List of Abbreviations and Symbols

Abbreviations and Symbols	Full Name or Meaning
C	Supervisory nodes
CN	Consensus nodes
SN	Propagation nodes
RN	New registered nodes
$CV(i)$	Credit value of node i
R	Credit value of node in the last round
$T(i)$	The turnover power of the node i
N_{max}	The maximum turnover power of the node in the network
$S(i)$	The number of times that the node has successfully processed the transaction
$F(i)$	The number of times that the node has malfunctioned as well as the number of times that it failed to process the transaction
$E(i)$	the number of times that the node has behaved maliciously
$f\ \max(i)/f\ \min(i)$	The maximum/minimum credit value of the nodes of class i

n	The total number of consensus nodes in the network
$Num(S)/Num(F)$	The number of nodes with successful/failed confirmation
$TN_{\Delta t}$	The number of transactions packed in the block during the Δt time period
T_c/T_f	The time when the system reaches consensus/the node sends the first transaction request

References

1. Szeberényi, A.; Bakó, F. Electricity market dynamics and regional interdependence in the face of pandemic restrictions and the Russian–Ukrainian conflict. *Energies* **2023**, *16*, 6515. [CrossRef]
2. Meng, T.; Zhao, Y.; Wolter, K.; Xu, C.-Z. On consortium blockchain consistency: A queueing network model approach. *IEEE Trans. Parallel Distrib. Syst.* **2021**, *32*, 1369–1382. [CrossRef]
3. Bukar, A.L.; Hamza, M.F.; Ayub, S.; Abobaker, A.K.; Modu, B.; Mohseni, S.; Brent, A.C.; Ogbonnaya, C.; Mustapha, K.; Idakwo, H.O. Peer-to-peer electricity trading: A systematic review on current developments and perspectives. *Renew. Energy Focus* **2023**, *44*, 317–333. [CrossRef]
4. Ahlqvist, V.; Holmberg, P.; Tangerås, T. A survey comparing centralized and decentralized electricity markets. *Energy Strategy Rev.* **2022**, *40*, 100812. [CrossRef]
5. Mahmood, M.S.; Al Dabagh, N.B. Blockchain technology and internet of things: Review, challenge and security concern. *Int. J. Electr. Comput. Eng.* **2023**, *13*, 718–735. [CrossRef]
6. Mateo-Cortés, J.A.; Arias-Antúnez, E.; Cazorla-López, D. Impact of Blockchain Technology for Business and Information Systems: Automation of Inter-Company Debt Compensation Case Study. *Appl. Sci.* **2023**, *13*, 4805. [CrossRef]
7. Vaigandla, K.K.; Karne, R.; Siluveru, M.; Kesoju, M. Review on blockchain technology: Architecture, characteristics, benefits, algorithms, challenges and applications. *Mesopotamian J. CyberSecurity* **2023**, *2023*, 73–84. [CrossRef]
8. Yin, R.; Yan, Z.; Liang, X.; Xie, H.; Wan, Z. A survey on privacy preservation techniques for blockchain interoperability. *J. Syst. Archit.* **2023**, *140*, 102892. [CrossRef]
9. Li, Y.; Qiao, L.; Lv, Z. An optimized byzantine fault tolerance algorithm for consortium blockchain. *Peer Peer Netw. Appl.* **2021**, *14*, 2826–2839. [CrossRef]
10. Knudsen, H.; Notland, J.S.; Haro, P.H.; Raeder, T.B.; Li, J. Consensus in blockchain systems with low network throughput: A systematic mapping study. In Proceedings of the 2021 3rd Blockchain and Internet of Things Conference, Ho Chi Minh City, Vietnam, 8–10 July 2021; pp. 15–23. [CrossRef]
11. Yang, J.; Paudel, A.; Gooi, H.B.; Nguyen, H.D. A proof-of-stake public blockchain based pricing scheme for peer-to-peer energy trading. *Appl. Energy* **2021**, *298*, 117154. [CrossRef]
12. Cao, B.; Zhang, Z.; Feng, D.; Zhang, S.; Zhang, L.; Peng, M.; Li, Y. Performance analysis and comparison of PoW, PoS and DAG based blockchains. *Digit. Commun. Netw.* **2020**, *6*, 480–485. [CrossRef]
13. Lee, E.Y.; Kim, N.R.; Han, C.R.; Lee, I.G. Evaluation and comparative analysis of scalability and fault tolerance for practical byzantine fault tolerant based blockchain. *J. Korea Inst. Inf. Commun. Eng.* **2022**, *26*, 271–277. [CrossRef]
14. Xu, H.; Zhang, L.; Liu, Y.; Cao, B. Raft based wireless blockchain networks in the presence of malicious jamming. *IEEE Wirel. Commun. Lett.* **2020**, *9*, 817–821. [CrossRef]
15. Zarrin, J.; Phang, H.W.; Saheer, L.B.; Zarrin, B. Blockchain for decentralization of internet: Prospects, trends, and challenges. *Clust. Comput.* **2021**, *24*, 2841–2866. [CrossRef]
16. Yu, J.; Qiao, Z.; Tang, W.; Wang, D.; Cao, X. Blockchain-based decision tree classification in distributed networks. *Intell. Autom. Soft Comput.* **2021**, *29*, 713–728. [CrossRef]
17. Khadke, S.; Gupta, P.; Rachakunta, S.; Mahata, C.; Dawn, S.; Sharma, M.; Verma, D.; Pradhan, A.; Krishna, A.M.S.; Ramakrishna, S.; et al. Efficient Plastic Recycling and Remolding Circular Economy Using the Technology of Trust–Blockchain. *Sustainability* **2021**, *13*, 9142. [CrossRef]
18. Said, D.A. A decentralized electricity trading framework (DETF) for connected EVs: A blockchain and machine learning for profit margin optimization. *IEEE Trans. Ind. Inform.* **2020**, *17*, 6594–6602. [CrossRef]
19. Kang, Q.; Hongbo, T.; Wei, Y.; Lingwei, W. Improved byzantine fault tolerance algorithm based on trusted list. *Comput. Appl. Softw.* **2022**, *39*, 1–8.
20. Chen, P.; Han, D.; Weng, T.-H.; Li, K.-C.; Castiglione, A. A novel byzantine fault tolerance consensus for green IoT with intelligence based on reinforcement. *J. Inf. Secur. Appl.* **2021**, *59*, 102821. [CrossRef]
21. Qin, J.; Sun, W.; Li, Z.; Zhu, Y. Credit consensus mechanism for microgrid blockchain. *Power Syst. Autom.* **2020**, *44*, 10–18.
22. Zhang, P.; Zhou, M.; Zhao, Q.; Abusorrah, A.; Bamasag, O.O. A performance-optimized consensus mechanism for consortium blockchains consisting of trust-varying nodes. *IEEE Trans. Netw. Sci. Eng.* **2021**, *8*, 2147–2159. [CrossRef]
23. Zhang, Z.; Zhu, D.; Fan, W. Qpbft: Practical byzantine fault tolerance consensus algorithm based on quantified-role. In Proceedings of the 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), Guangzhou, China, 29 December–1 January 2020; pp. 991–997. [CrossRef]
24. Dashtdar, M.; Najafi, M.; Esmaeilbeig, M.; Bajaj, M. Placement and optimal size of DG in the distribution network based on nodal pricing reduction with nonlinear load model using the IABC algorithm. *Sādhanā* **2022**, *47*, 73. [CrossRef]

25. Yao, Z.; Fang, Y.; Pan, H.; Wang, X.; Si, X. A secure and highly efficient blockchain PBFT consensus algorithm for microgrid power trading. *Sci. Rep.* **2024**, *14*, 8300. [CrossRef]
26. Xiao, J.; Luo, T.; Li, C.; Zhou, J.; Li, Z. CE-PBFT: A High Availability Consensus Algorithm for Large-Scale Consortium Blockchain. *J. King Saud Univ. Comput. Inf. Sci.* **2024**, *36*, 101957. [CrossRef]
27. Xiong, X.; Liu, Y.; Qu, H.; Cai, Y. Power trading Raft consensus mechanism considering green certificate and carbon emission weights. *Front. Energy Res.* **2023**, *11*, 1298318. [CrossRef]
28. Ren, X.; Zhang, L. Improved multi-master node consensus mechanism based on practical byzantine fault tolerance. *J. Comput. Appl.* **2022**, *42*, 1500–1507.
29. Hasan, M.K.; Alkhalifah, A.; Islam, S.; Babiker, N.B.M.; Habib, A.K.M.A.; Mohd Aman, A.H.; Hossain, M.A. Blockchain Technology on Smart Grid, Energy Trading, and Big Data: Security Issues, Challenges, and Recommendations. *Wirel. Commun. Mob. Comput.* **2022**, *2022*, 9065768. [CrossRef]
30. Lu, S.; Zhang, X.; Zhao, R.; Chen, L.; Li, J.; Yang, G. P-Raft: An Efficient and Robust Consensus Mechanism for Consortium Blockchains. *Electronics* **2023**, *12*, 2271. [CrossRef]