A Consensus Mechanism Based on an Improved Genetic Algorithm

Chen Yang, Tao Wang, Kun Wang

The College of Nuclear Technology and Automation Engineering, Chengdu University of Technology, Chengdu, China
Email: 604493732@qq.com, wangt@cdu.edu.cn, 1213737146@qq.com

Abstract
An important feature of blockchain technology is that all participants jointly maintain transaction data and can achieve mutual trust relationships without integrated control, which relies on distributed consensus algorithms. Practical Byzantine Fault Tolerant algorithm (PBFT) is a fault-tolerant algorithm based on state machine replication, which solves the Byzantine error, that is, the malicious behavior of nodes. In PBFT, all participating nodes are divided into the primary node and backup nodes. When this primary node commits evil or fails, it will elect a primary node again for message communication. The genetic algorithm (GA) is a computer simulation study inspired by the natural biological genetic evolution criterion “natural selection, survival of the fittest”. Genetic algorithm is actually a method to find the optimal solution. According to it, the best primary node is selected in the PBFT algorithm to improve consensus efficiency. The consensus algorithm is the guarantee of the decentralization feature in blockchain technology. The PBFT algorithm is a commonly used consensus algorithm. However, this algorithm has the following problems: when the primary node fails, it must be selected again, which leads to a decrease in consensus efficiency. This paper proposes a consensus mechanism based on an improved genetic algorithm, which uses an improved genetic algorithm to select the primary node. According to the genetic algorithm, the best primary node is selected, and it meets the minimum number of errors or evils and the highest transaction efficiency with other backup nodes. The improved consensus algorithm can effectively reduce system delay and improve consensus efficiency.

Subject Areas
Bioinformatics

Keywords
Consensus Mechanism, Genetic Algorithm, Improvement
1. Introduction

In blockchain system, the consensus algorithm enables all participants to jointly maintain transaction data and achieve mutual trust relationships without integrated control. Consensus algorithms can be roughly divided into proof-based consensus algorithms and voting-based consensus algorithms; the most classic consensus voting-based algorithm is the PBFT algorithm. But this consensus algorithm has some problems to improve. Then many variant consensus algorithms are presented to issue the shortcomings of PBFT. In BBFT [1], it uses the topology tree structure and message aggregation to reduce the communication complexity. Liu et al. presented the FastBFT [2] in 2018. It uses a message aggregation and trust execution environments technique to improve the PBFT. Since in PBFT, the total number of nodes is 3f + 1 (f is the number of fault nodes), then Giuliana Santos Veronese et al. propose MinBFT [3] and Rudiger Kaitza et al. present CheapBFT [4]. Their methods all reduce the total nodes’ number from 3f + 1 to 2f + 1. Zyzzyva [5] is a Speculative Byzantine Fault Tolerance which is proposed by Ramakrishna Kotla et al. It changes three-protocol of PBFT to improve performance. HAO et al. proposed the CDBFT [6]. It combines DPOS and PBFT to reduce the participation probability of malicious replicas in the consensus process. Jeon et al. proposed the DBFT [7] which makes optimization for the application of public blockchain. Among all the PBFT variant algorithms, few have improved the algorithm by changing the way the master node is selected. So this article proposes a consensus mechanism based on an improved genetic algorithm. BAGLEY took the lead in using the concept of “genetic algorithm” in the paper [8] in 1967. The improved genetic algorithm is used to obtain the best primary node when it is selected. After calculation, the algorithm improves transaction efficiency and processing time. The structure of the paper is as follows: Section 2 describes the traditional PBFT algorithm, Section 3 introduces the consensus algorithm based on the improved genetic algorithm, and Section 4 does a summary to this article.

2. Practical Byzantine Fault Tolerance Algorithm

PBFT algorithm was presented by Castro and Liskio in 1999 [9]. It requires that each consensus be executed in the same order on each copy, and each participant decides the content of the consensus by voting. PBFT adopts a three-phase protocol: PREPREPARE, PREPARE, and COMMIT.

**PRE-PREPARE**

In the PRE-PREPARE phase, the primary sends a pre-prepare message to all replica nodes.

**PREPARE**

When the backup node i has received the pre-prepare message, it enters the PREPARE phase, and sends the prepare message to other nodes. Besides it writes both the pre-prepare message and the prepare message to the log.

In the PBFT algorithm, the PRE-PREPARE phase and the PREPARE phase
ensure that the non-faulty replica nodes agree on the ordering of requests in the same view. BFT systems shall guarantee safety and liveness in the presence of faulty servers and clients [10].

COMMIT

The replica joins in the COMMIT phase and broadcasts to all replicas the commit message. Figure 1 is a flow chart of the three-phase agreement.

The above three steps determine the number of replicas in PBFT is $3f + 1$ and the communication complexity is $o(f^2)$ ($f$ is the number of faulty nodes).

3. A Variant Consensus Mechanism Based on an Improved Genetic Algorithm

Since in the PBFT algorithm nodes are divided into the primary node and other replica nodes, and the primary node as a consensus node executes a three-phase protocol to complete transactions. Once the primary node fails, it will re-select, which will cause the efficiency of system transaction execution to decrease. Therefore, the selection of the primary node is very essential. This paper proposes an improved genetic algorithm to select the best primary node of PBFT. The improved genetic algorithm mainly uses the elitist strategy to retain the best individual, and the best individual does not undergo mutation, crossover and other operations, and directly enters the next generation. The improved algorithm not only improves convergence, but also ensures that the optimal individual characteristics are not destroyed in the evolution process [11] [12]. Firstly, each participant is involved in coding into the corresponding chromosomes to form the initial initial population. Secondly, select the master node according to the fitness function. Then use the elitist selection strategy to perform crossover and mutation operations to achieve the best primary node selection.

Step I

Encoding all participating codes to chromosome. Suppose there are $m$ participating nodes, a chromosome of an $m$-dimensional vector is obtained after encoding, this dimension is the number of all participating nodes $N = \{n_1, n_2, ..., n_a, ..., n_m\}$. 

![Figure 1. PBFT algorithm flow chart.](image-url)
Step II
Setting fitness function \( f(n_i) = \sum_{n_{i \in N}} Fau(n_i)EFF(n_i) \) for all encoded chromosomes. \( n_i \) represents a legal chromosome, \( N \) represents the set of all chromosomes, \( Fau() \) represents the number of times that the node has problems, and \( Eff() \) represents the efficiency of the node’s transaction with other nodes. According to the \( f(n_i) \) to obtain the optimal solution, the best primary node is selected.

Step III
Selecting the best master node according to the elitist selection strategy, and other nodes perform mutation and selection operations.

Step IV
When the termination conditions are met, the optimal solution of the function will be used as the primary node to participate in the implementation of the PBFT three-phase protocol.

Figure 2 is a flowchart of the improved algorithm.
4. Conclusions

Latency is an important indicator to measure the performance of the blockchain. The delay is mainly affected by two aspects: one is the network communication performance when the message is propagated between nodes, and the other is the operating efficiency of the consensus algorithm. According to formula $\text{Delay} = T_{\text{broadcast}} + T_{\text{consensus}}$ in the fitness function $f(n_i) = \sum_{n_j \in N} Fau(n_i) \cdot \text{EFF}(n_j)$, if a node meets the minimum $Fau()$ and the maximum $\text{EFF}()$, during the consensus process, $T_{\text{broadcast}}$ decreases, so the entire $\text{Delay}$ decreases. Compared with the traditional PBFT algorithm, the delay is effectively reduced.

This paper proposes a PBFT algorithm based on an improved genetic algorithm, which uses genetic algorithm to improve the selection strategy of the primary node, so that the improved PBFT algorithm consensus efficiency is higher.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] https://github.com/bystackcom/BBFT-Whitepaper/blob/master/whitepaper.pdf
[2] Liu, J., Li, W., Karam, G., et al. (2016) Scalable Byzantine Consensus via Hardware-Assisted Secret Sharing. IEEE Transactions on Computers, 68, 139-151. https://doi.org/10.1109/TC.2018.2860009
[3] Veronese, G.S., Correia, M., Bessani, A.N., et al. (2013) Efficient Byzantine Fault-Tolerance. IEEE Transactions on Computers, 62, 16-30.
[4] Kapitza, R., Behl, J., Cachin, C., Distler, T., Kuhnle, S., Mohammadi, S.V., Schröder-Peikschat, W., and Stengel, K. (2012) Cheapbft: Resource-Efficient Byzantine Fault Tolerance. Proceedings of the 7th ACM European Conference on Computer Systems, April 2012, 295-308. https://doi.org/10.1145/2168836.2168866
[5] Kotla, R., Clement, A., Wong, E., et al. (2008) Zyzzyva: Speculative Byzantine Fault Tolerance. Speculative Byzantine Fault Tolerance. https://doi.org/10.1145/1294261.1294267
[6] Wang, F.Y., Cai, S.S., Lin, T.C., et al. (2019) Study of Blockchains’s Consensus Mechanism Based on Credit. IEEE Access, PP(99), 1-1. https://doi.org/10.1109/ACCESS.2019.2891065
[7] Jeon, S., Doh, I. and Chae, K. (2018) RMBC: Randomized Mesh Blockchain Using DBFT Consensus Algorithm. 2018 International Conference on Information Networking (ICOIN), Chiang Mai, 10-12 January 2018. https://doi.org/10.1109/ICOIN.2018.8343211
[8] Rudolph, G. (1994) Convergence Analysis of Canonical Genetic Algorithms. IEEE Transactions on Neural Networks, 5, 96. https://doi.org/10.1109/72.265964
[9] Castro, M. and Liskov, B. (2002) Practical Byzantine Fault Tolerance. ACM Transactions on Computer Systems, 20, 398-461. https://doi.org/10.1145/571637.571640
[10] Cai, Q., Lin, J., Li, F., et al. (2014) EFS: Efficient and Fault-Scaleable Byzantine Fault...
[11] Ardjmand, E., et al. (2020) A Hybrid Artificial Neural Network, Genetic Algorithm and Column Generation Heuristic for Minimizing Makespan in Manual Order Picking Operations. Expert Systems with Applications, 159, 113566. https://doi.org/10.1016/j.eswa.2020.113566

[12] Veeramsetty, V., Lakshmi, G.V.N. and Jayalaxmi, A. (2012) Optimal Allocation and Contingency Analysis of Embedded Generation Deployment in Distribution Network Using Genetic Algorithm. 2012 International Conference on Computing, Electronics and Electrical Technologies (ICCEET), Kumaracoil, 21-22 March 2012. https://doi.org/10.1109/ICCEET.2012.6203763