A DQN-based Consensus Mechanism for Blockchain in IoT Networks

Zhiming Liu, *Student Member, IEEE,* Lu Hou, *Member, IEEE,* Kan Zheng, *Senior Member, IEEE,* Qihao Zhou, *Member, IEEE,* and Shiwen Mao, *Fellow, IEEE*

***Abstract*—The integration of the blockchain and Internet of Things (IoT) systems can effectively guarantee data security in IoT applications. To facilitate the use of blockchain on resource-constrained IoT end-devices, we propose RAFT+ with a new leader selection scheme in this paper, which is based on the distributed consensus algorithm RAFT. The design of RAFT+ aims at mitigating the imparities between different types of IoT end-devices and enabling these devices to allow different types of IoT end-devices to participate in block consensus, thus maintaining strong consistency of the blockchain network. The leader selection scheme is generated by a Deep Q-Network (DQN), which can make the optimal selection of the leader under various conditions by leveraging the limited system resources as well as balancing the load of the consensus mechanism on multiple IoT end-devices. Simulation results show that RAFT+ can enhance the system performance while maintaining the security of the system under high load conditions.**

***Index Terms*—Blockchain, IoT, Consensus mechanism, DQN, RAFT**

I. Introduction

With the rapid development of Internet of Things (IoT) applications, the operating expense and security requirements of IoT systems have become important recently. Due to the increasing number of IoT end-devices, the traditional IoT systems face the following issues, i.e.,

- Data security issues caused by traditional databases. The IoT data is vulnerable to be stolen, tampered or destroyed.
- Performance bottleneck caused by centralized architecture. The capability is highly limited by central servers.
- Resources waste caused by unbalanced workload between central servers and edge servers.

The blockchain technology has become one of the most promising solutions for solving these issues, thanks to its characteristics of immutability and decentralization [1]. With the deployment of blockchain, the data in IoT systems are guaranteed to be consistent and secured [2]. One feasible way to integrate IoT systems with the blockchain is to build a multi-layer blockchain network to meet the different capabilities of the IoT modules [3]. The multi-layer blockchain network can improve the scalability of IoT systems and enable the deployment of services on different layers. Meanwhile, the multi-layer blockchain network can meet the privacy needs of different services and ensure the security of data. Generally, the private blockchain is suitable for IoT systems since the data is transmitted and stored privately [4]. As a decentralized schme, the blockchain requires a consensus mechanism to ensure the consistency of data [5]. The main goal of the consensus mechanism is to reach an agreement on the generation of new blocks among all the participants. However, to avoid the impact of faulty and malicious nodes, the consensus mechanism needs to be crash-fault-tolerant or even byzantine fault tolerant without trusted third party's verification [6]. Usually, an IoT system mainly includes three types of modules, i.e., the central servers, the base stations and the end-devices. As the core module of data processing in IoT systems, central servers have plenty of computing resources to handle the tasks for the deployment of the blockchain [7].

To make full use of the available resources of end-devices and base stations, this paper proposes a blockchain architecture for IoT applications that requires security assurance. In the proposed architecture, the IoT system is divided into three layers, i.e., the cloud layer, the edge layer, and the end-device layer. IoT end-devices are divided into strong end-devices and weak end-devices based on their computing abilities. In the proposed architecture, the IoT modules are layered to distinguish their performance and meet the construction requirements of a multi-layer blockchain network. The multi-layer blockchain network is consists of a blockchain network on the cloud and several local blockchain networks. Meanwhile, the local blockchain network is constructed by strong end-devices and base stations. The multi-layer blockchain network needs to use a proper consensus mechanism to alleviate the load caused by the deployment of blockchain nodes on end-devices with different abilities. Common consensus algorithms, such as Proof of Work (PoW) [8] and Practical Byzantine Fault Tolerance (PBFT) [9, 10], are not suitable for the blockchain network in the proposed architecture due to their high computation or communication requirements for modules. Lightweight consensus algorithms, such as the RAFT algorithm [11], provide a feasible solution to alleviate the computing load caused by the deployment of blockchain nodes. The RAFT algorithm has the characteristic of selecting

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61731004, and in part by the Key Lab of Universal Wireless Communications, Ministry of Education. *(Corresponding author: Kan Zheng.)*

Z. Liu, L. Hou and Q. Zhou are with the Intelligent Computing and Communications Laboratory, Wireless Signal Processing and Networks Lab, Key Laboratory of Universal Wireless Communications, and Ministry of Education, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: liuzhiming@bupt.edu.cn).

K. Zheng is with Ningbo University, Ningbo, China, 315211 (e-mail: kzheng@ieee.org).

S. Mao is with the Department of Electrical and Computer Engineering, Auburn University, Auburn, AL 36849, USA (e-mail: smao@ieee.org).

a leader regularly and the leader is responsible for replicating the blocks of nodes, which makes it suitable for IoT systems [12, 13]. The RAFT algorithm divides the system running time into terms, and each node in the network switches among three states, i.e., leader, follower, and candidate. The Leader selection in the RAFT algorithm is triggered by the missing of heartbeat packets. In each term, the candidate that reaches the quorum of votes becomes the leader. Then, the leader sends heartbeat packets to other nodes to establish leader authority and terminate the selection process. However, the RAFT algorithm does not take into account the different resources in IoT end-devices, because it is a purely random mechanism. Therefore, there are resource wastes when use the RAFT algorithm in IoT-blockchain systems.

Therefore, we propose RAFT+, an improved algorithm based on the RAFT algorithm with a new leader selection scheme. Firstly, RAFT+ collects computing resources of each end-device and samples the time-variant communication environment of the system as the parameters of leader selection. Then, a central node is chosen in the blockchain network to collect selection parameters and select the leader. Considering the computing resources of different IoT modules, base stations are the central nodes in the local blockchain networks. Finally, all blockchain nodes participate in leader selection as candidates. The purpose of the above improvements is to make the optimal leader choice considering the current status of the system resources. The performance metric of leader selection is the block generation latency, which is defined as the sum of the block packaging latency and the block consensus latency. The optimization problem is formulated as a Markov Decision Process (MDP) model, which is solved by the Deep Q-Network (DQN) algorithm. The main contributions of this paper are summarized as follows, i.e.,

- A multi-layer blockchain architecture is proposed for IoT systems. By layering an IoT network on top of a blockchain network, we explain that this layered structure has good adaptability to the multi-layer blockchain network. Meanwhile, the blockchain networks is formed by multiple types of IoT modules to make efficient use of different IoT modules.
- Based on the blockchain architecture, RAFT+ is proposed as the consensus mechanism of the blockchain network, which selects a leader by considering the time-variant communication environment and computing resources of each end-device.
- The leader selection scheme in RAFT+ is based on the DQN algorithm, which improves the original leader selection mechanism and maintains the fault-tolerance performance of the original RAFT algorithm. The optimal selection strategy is generated by neural networks.

The rest of the paper is organized as follows. Section II overviews the existing work related to this paper. Section III gives the system model and problem formulation. Section IV provides the MDP formulation and a detailed description of the DQN-based RAFT+. Section V analyzes the fault-tolerant performance of RAFT+. In Section VI, the performances of RAFT+ are evaluated by simulations. Conclusions are drawn in Section VII.

II. Related Work

In the traditional IoT systems, end-devices and base stations are responsible for data reporting and forwarding, respectively, while computing services and databases are deployed on the central servers [14, 15]. The disadvantage of the centralized data processing structure is that system efficiency completely depends on the performance of the central servers [16, 17]. Meanwhile, there are security problems, such as privacy leakage [18].

Liu *et al.* [19] transfered part of the computing tasks to base stations, utilizing the computing resources of base stations to alleviate the load pressure on the central server and improve the efficiency of the system. Security challenges are generally addressed by combining IoT systems and blockchain in the existing research. In the blockchain network built for IoT systems, Guo *et al.* [20] utilized resources of all the IoT modules in the system to improve system efficiency. In specific IoT scenarios, the blockchain network needs to be designed according to the characteristics of scenarios to maintain the stable operation of the systems [21, 22]. In the Industrial Internet of Things (IIoT) scenario, network security is particularly important [23, 24]. Most IIoT infrastructures are based on a centralized architecture which is easier to manage but does not effectively support validation services between multiple parties. The blockchain-based IIoT architecture provides effective validation services and data storage schemes for resource-constraint IIoT infrastructures [25]. However, many prior works adopt common consensus algorithms, such as PoW, without considering the performance differences between the IoT modules. Due to the use of common consensus algorithms, a heavy workload is placed on end-devices. With the operation of the system, end-devices may stop working, thus affecting the integrity of the blockchain network and the efficiency of the system.

To maintain the stable state of end-devices in the blockchain network, a lightweight blockchain consensus algorithm needs to be developed based on the performance characteristics of end-devices. Existing research on lightweight consensus algorithms can be divided into two categories. One category is to improve the common consensus algorithms [26, 27]. Based on PoW, Alhejazi *et al.* [28] proposed a novel algorithm adapted from the concept of weighted majority algorithm in ensemble learning, called WMCA, which enhanced the detection of malicious anomalies, thus improving the security level of IoT systems. Similarly, Huang *et al.* [29] proposed a credit-based PoW mechanism for IIoT scenario. To protect sensory data confidentiality, a data authority management method was designed in [29] based on directed acyclic graph-structured blockchains. Li *et al.* [30] proposed an improved PBFT consensus mechanism based on reward and punishment strategy. Meanwhile, a storage optimization scheme based on RS erasure code was proposed to reduce the cost of blockchain storage. Choi *et al.* [31] replaced the proof of work in PoW with the proof of trust accumulated by nodes working continuously in the blockchain network. Meanwhile, Zaman

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3132420, IEEE Internet of Things Journal

et al. [32] replaced the proof of work with the proportion of the resources paid by the nodes to the resources they have. Although the above improved methods can retain the mature algorithm flow of PoW, it is difficult to reduce the computational load caused by the consensus mechanism.

The other category is to design new consensus algorithms for specific application scenarios. Biswas *et al.* [33] simplified the consensus process into two steps, transaction verification and consensus formation. Peers were merged according to the number of nodes participating in the consensus to reduce the computing time required to reach consensus. For the consortium blockchain, Wang *et al.* [34] proposed a consensus fusion scheme based on trust levels to complete collaborative learning. The candidates with higher trust levels played a more important role in the consensus. Similarly, Kaci *et al.* [35] proposed a new reputation-based blockchain named PoolCoin based on a distributed trust model for mining pools. The trust model provided a machine learning module to assess the capabilities of mining machines, allowing pool managers to select trusted miners in their mining pools. In [36], each IoT sensor generated its unique tag and broadcasts it to the blockchain network. After a trusted node in the blockchain network authenticates the sensor, the sensor hashes the authenticated block through the lightweight hash function and adds the block to the blockchain. However, the RAFT algorithm has good adaptability to the blockchain network for IoT applications due to its algorithm characteristics [37, 38].

The RAFT algorithm divides the system running time into multiple terms, and the beginning of each term is leader selection. During leader selection, each node in the network switches among three node states according to leader selection results. The leader selection in the RAFT algorithm is triggered by the heartbeat mechanism. When any follower fails to receive the heartbeat packet sent by the leader within the fixed time, the follower converts to a candidate and triggers leader selection. In the selection, all the followers who fail to receive the heartbeat packet are selected as candidates. At the beginning of the selection, each candidate sends requests in parallel to obtain the votes of followers, and each follower votes for the candidate whose request it receives first. The candidate who obtains the votes of the majority nodes becomes the new leader. Additionally, the new leader sends heartbeat packets to other nodes in the network to establish leader authority and terminate the selection.

In a blockchain network with the RAFT algorithm as the consensus mechanism, the leader is responsible for packaging transactions to generate blocks, and blocks are sent to followers for endorsement. After receiving a new block, the followers endorse the block and return confirmation responses containing the endorsement results to the leader. The leader notifies nodes in the network to update the new block synchronously after receiving enough confirmation responses. The efficiency of this consensus algorithm is determined by the performance of the leader and the communication environment [11]. The block consensus method of the RAFT algorithm can avoid heavy computing burdens, while the unique leader ensures the consistency of data in the network [39]. However, the RAFT algorithm does not take into account the limited system resources in IoT applications since its selection parameters are fixed. Moreover, leader selection causes a large communication loss, and there is a randomness in the leader selection process. To be implemented, the RAFT algorithm needs to be modified to fit the characteristics of various IoT scenarios.

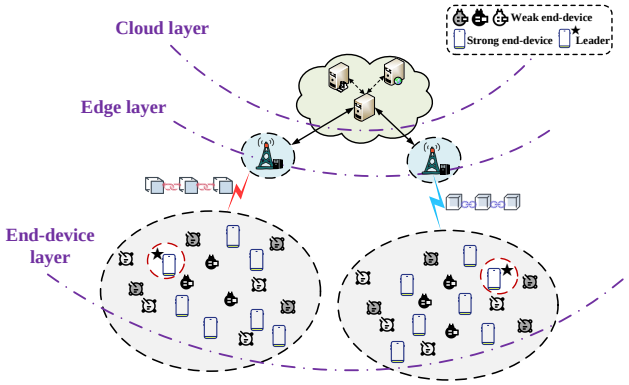

Fig. 1. Blockchain architecture for IoT applications.

III. System Model and Problem Formulation

The proposed IoT-blockchain system is shown in Fig. 1, it consists of a blockchain network on the cloud and several local blockchain networks. The local blockchain network constructed in this paper consists of a base station and strong end-devices in the coverage area of the base station. Meanwhile, the blockchain network on the cloud consists of the central servers. The local blockchain network uses RAFT+ as the consensus mechanism, which stipulates that the leader generates blocks and impels block consensus. When a block is generated in the local blockchain network, the leader notifies nodes in the network to update the new block synchronously. The base station reports the generated new block to the central server in the blockchain network on the cloud.

A. Scenario description

Assuming an IoT system with a single base station coverage scenario, as shown in Fig. 1, where the number of strong end-devices and the number of weak end-devices are I and J, respectively. Strong end-devices are indexed by i, $i \in \{1, 2, ..., I\}$. A base station and strong end-devices form a blockchain network, of which the number of nodes is $I+1$. The base station and strong end-devices are regarded as the central node and candidates of leader, respectively. The communication environment and computing resources of each strong end-device are considered during leader selection. After leader selection, the selected leader is represented by l and the remaining nodes become followers. Weak end-devices collect and report data to the nearest strong end-device, while strong end-devices forward the collected data to leader l. Following the consensus algorithm, the data received by leader l is packaged in the form of transactions into blocks, and the new blocks are replicated to the blockchain ledger of each node in the blockchain network.

B. Communication model

End-devices collect and report data at fixed intervals. Each end-device collects data for only one application type. The system time is divided into multiple epochs, and each epoch starts with the end of previous block consensus. Let n represent the number of data packets reported by each end-device in an epoch, which follows the Poisson distribution and is given as follows,

$$p\left(n\right)=\frac{\lambda^{n}}{n!}e^{-\lambda}, \tag{1}$$

where λ is the average number of data packets collected by the end-device in an epoch. D_m is the size of data packets, and the amount of data reported by the end-device in an epoch is nD_m.

During the data transmission process between the IoT modules, only large-scale fading including path loss is considered in the wireless channel model. The transmission rate between the IoT modules based on the Shannon formula is given as follows,

$$r_{l,i}=\frac{B}{I}\log_2\left(1+\gamma_{l,i}\right), \tag{2}$$

where B is the bandwidth used for block consensus in the system, and $\gamma_{l,i}$ is the signal-to-noise ratio (SNR) of data transmission between leader l and follower i, i.e.,

$$\gamma_{l,i}=\frac{P_{\mathrm{CON}}d_{l,i}^{-\beta}\Psi}{BN_0}, \tag{3}$$

where P_{CON} is the total transmission power that is equally divided for the followers for block consensus, N_0 represents the average power spectral density of white noise, $d_{l,i}$ represents the distance between leader l and follower i, β stands for the path loss ratio, and $\Psi\sim\mathcal{LN}\left(0,\sigma^2\right)$ denotes the shadow fading which follows a log-normal distribution. An $I\times I$ matrix is used to represent the SNR of data transmission between the IoT modules in the blockchain network, which is given as follows,

$$\begin{pmatrix} 0 & \gamma_{1,2} & \cdots\gamma_{1,I-1} & \gamma_{1,I} \\ \gamma_{2,1} & & & \gamma_{2,I} \\ \gamma_{3,1} & & \ddots & \gamma_{3,I} \\ \vdots & & & \vdots \\ \gamma_{I-1,1} & & & \gamma_{I-1,I} \\ \gamma_{I,1} & \gamma_{I,2} & \cdots\gamma_{I,I-1} & 0 \end{pmatrix}. \tag{4}$$

C. Leader selection model

To enhance the overall communication efficiency in the process of block consensus, RAFT+ selects a central node in the blockchain network to collect the parameters of other nodes and selects the leader. If the leader does not receive a sufficient number of confirmation responses with endorsement results during the longest block consensus time $t_{\max}^{\mathrm{CON}}$, block consensus is considered failed and the leader selection process will be triggered. There could be two reasons that leads to such a scenario, i.e.,

- **Hardware failure:** Hardware failure is defined as a hardware failure of the leader. Since if any follower has a hardware failure, the rest of the followers can still return confirmation responses and the blocks can reach consensus. A hardware failure of the leader causes the leader to fail to receive confirmation responses, resulting in a failed block consensus. The probability of such a fault is represented by p_{hard}.
- **Communication failure:** Data transmission rate $r_{l,i}$ between leader l and follower i is determined by the SNR, and SNR $\gamma_{l,i}$ between leader l and follower i changes with the time-variant communication environment of the system. When data transmission rate $r_{l,i}$ falls below the minimum rate requirement, the communication between nodes is interrupted. Data transmission rate $r_{l,i}$ falls below the minimum requirement, which means SNR $\gamma_{l,i}$ falls below the minimum SNR requirement. The communication interruption probability $p_c\left(l,i\right)$ between leader l and follower i is expressed as follows,

$$p_c\left(l,i\right)=\Pr(\gamma_{l,i}<\gamma_0)=\int_0^{\gamma_0}f(x)dx, \tag{5}$$

where γ_0 represents the minimum SNR requirement for data transmission between nodes, and $f(x)$ is the probability density function of SNR. If more than $\lceil I/2\rceil$ followers meet the communication interruption with the leader, communication failure occurs in the blockchain network. Let B_{tr} represent the set of all followers at timestamp t, and B_{er} represent the set of followers which meet the communication interruption during data transmission at timestamp t, $B_{\mathrm{er}}\subseteq B_{\mathrm{tr}}$. When the number of elements in set B_{er} is more than $\lceil I/2\rceil$, the probability p_{com} of communication failure due to communication interruption is expressed as follows,

$$p_{\mathrm{com}}=\prod_{j\in B_{\mathrm{er}}}p_c\left(l,j\right)\cdot\prod_{m\notin B_{\mathrm{er}}\cap m\in B_{\mathrm{tr}}}\left[1-p_c\left(l,m\right)\right]. \tag{6}$$

The probability of triggering the leader selection process by various faults is expressed as follows,

$$p_e=p_{\mathrm{hard}}+p_{\mathrm{com}}-p_{\mathrm{hard}}\cdot p_{\mathrm{com}}. \tag{7}$$

After completing leader selection, the central node sends the selection result to nodes in the blockchain network. Each node updates its node status and resends the reported but unpackaged data to the new leader. The new leader completes block packaging and block consensus tasks.

D. Block consensus model

The basic block consensus flow of RAFT+ is shown in Fig. 2. The data reported by weak end-devices and forwarded by strong end-devices is stored in the cache pool of the leader in the form of transactions. The leader packages all transactions in the cache pool at the packaging interval to generate blocks. The number of transactions contained in the block is determined by the amount of data reported by each end-device. After the block is generated, the leader sends it to the followers in the blockchain network for endorsement. Each follower returns a confirmation response containing the endorsement result. After the leader receives $\lceil I/2\rceil$ confirmation responses, the generation of the block reaches a consensus within the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3132420, IEEE Internet of Things Journal

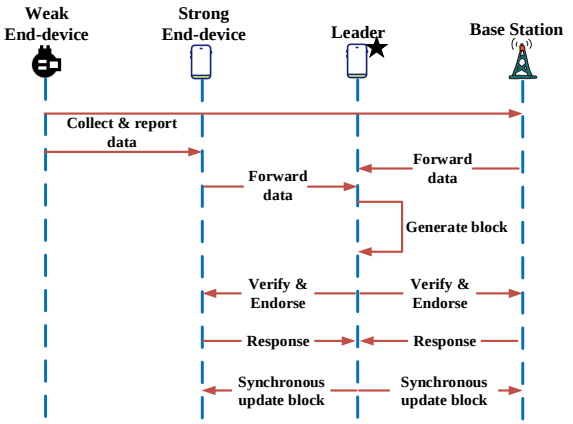

Fig. 2. The basic block consensus flow.

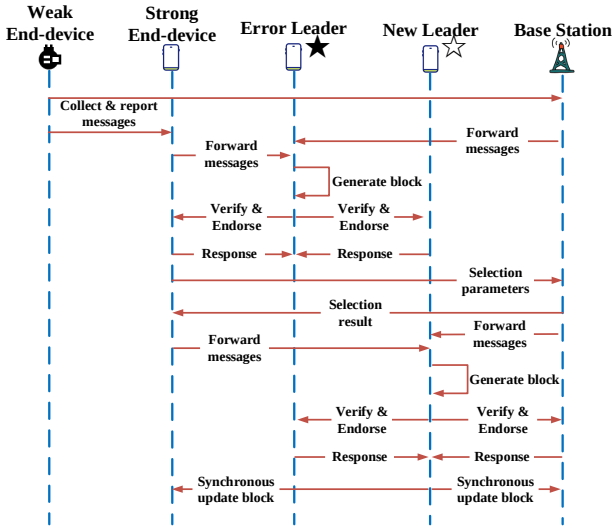

Fig. 3. The block consensus flow when leader failures occur.

blockchain network and the leader notifies the followers to synchronously update the block. The block generation process is mainly divided into two steps, i.e., block packaging and block consensus.

During a block consensus process, if the leader fails to receive enough confirmation responses, block consensus will be invalid and leader selection will be triggered. There are two main reasons for the above situation, one is the leader's hardware failure, and the other is the poor communication environment of the system. When the new leader is selected through the leader selection scheme, followers will report the data that fails to generate the block to the new leader. The new leader will package the data to generate a new block and start block consensus. Leader selection in RAFT+ is dominated by the central node, which collects the parameters of each node in the blockchain network and selects the new leader. The new leader collects and packages data in the form of transactions to generate a new block. After the new block is generated, block consensus is carried out in the network. The specific process is shown in Fig. 3.

In the proposed blockchain architecture, a block is composed of a block header and a block body. The block header mainly stores metadata for identifying blocks and the Merkle root is calculated based on the transactions stored in the block. Specifically, the Merkle root is generated by the Merkle tree, and each leaf node in the Merkle tree represents a transaction stored in the block. For block k containing n_k transactions, the number of hash values n_s contained in the Merkle root is given as follows,

$$n_s = 2n_k - 1. \tag{8}$$

The block body stores specific data in the form of transactions. The amount of data reported by strong end-device i received in block k is represented by $n_{k,i}$, and the number of transactions stored in block k is given as follows,

$$n_k = \sum_{i=1}^{I} n_{k,i}. \tag{9}$$

Let η_l indicate the number of Central Processing Unit (CPU) cycles available for leader l in a unit time, and ξ_h represent the number of CPU cycles required to calculate a hash value from the hash function for a transaction. For block k with n_k transactions, $n_k - 1$ times of hash operations are required to generate the Merkle root. The number of CPU cycles required for a hash operation to generate the Merkle root is ξ_m. Combine the above variables, the block packaging latency is given as follows,

$$t_{l,k}^{\mathrm{PACK}} = \frac{n_k \xi_h + (n_k - 1)\,\xi_m}{\eta_l}. \tag{10}$$

In the RAFT algorithm, the leader sends heartbeat packets to followers in the blockchain network to maintain the state of each node. In RAFT+, the new blocks are regularly packaged by leader l and reported to followers also serve to maintain the state of each node. The data transmission process of block consensus includes two parts, which are (i) the leader sends the new block to the followers and (ii) the followers return confirmation responses containing the endorsement results. The block consensus latency between leader l and follower i can be expressed as follows,

$$t_{i,k}^{\mathrm{CON}} = \frac{n_k D_t + n_s D_h}{r_{l,i}} + \frac{D_f}{r_{i,l}} = \frac{n_k D_t + n_s D_h + D_f}{r_{l,i}}, \tag{11}$$

where D_t represents the data size of a transaction, D_h represents the data size of a hash value, and D_f represents the data size of a confirmation response. Assuming that the confirmation response returned by follower i is the $\lceil I/2 \rceil$th confirmation response received by the leader, the generation latency for block k is given as follows,

$$t_{l,k}^{\mathrm{GEN}} = t_{l,k}^{\mathrm{PACK}} + t_{i,k}^{\mathrm{CON}}. \tag{12}$$

E. Objective

The optimization goal of the system is to obtain the optimal leader selection strategy π^* for RAFT+. In the strategy, selection parameters are comprehensively analyzed to choose a leader that can minimize the average block generation latency. The specific optimization problem can be defined as follows,

$$\min \mathbb{E}_\tau \left[\frac{1}{K} \sum_{k=1}^{K} \left\{ (1-p_e)\, t_{l,k}^{\mathrm{GEN}} + p_e \left(t_{\max^{\mathrm{CON}}} + t_{z,k}^{\mathrm{GEN}} \right) \right\} \right],$$
$$\text{s.t. } \mathrm{C1}: l \in \{1,2,...,I\}, z \in \{1,2,...,I\},$$
$$\mathrm{C2}: l \neq z, 0 \leq p_e \leq 1, \tag{13}$$

where z represents the new leader selected by the central node. The confirmation response returned by follower i is the $\lceil I/2 \rceil$th confirmation response that the leader has received. $t_{l,k}^{\mathrm{GEN}}$ represents the block generation latency of block k generated by leader l, and $t_{z,k}^{\mathrm{GEN}}$ represents the block generation latency of block k generated by new leader z.

IV. DQN BASED RAFT+ ALGORITHM

The optimization problem (13) is formulated as an MDP model which consists of a state set $\mathcal{S}$, an action set $\mathcal{A}$, and a reward function $R: \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. Let τ denote the epoch when the leader starts to generate a new block. Without extra explanations, the symbol τ is omitted for simplicity, i.e., $S_\tau := S$, $A_\tau := A$. The DQN algorithm is adopted to find the optimal policy π^* by a training procedure.

A. MDP model formulation

1) State set: System state S is defined as follows,

$$S = (M, L, P, N, E), \tag{14}$$

where M represents the number of data received by the leader. Let M_i represent the number of data received by each strong end-device at epoch τ. M can be specifically expressed as follows,

$$M = \sum_{i \in \{1,2,...I\}} M_i. \tag{15}$$

L represents the serial number of the leader in the blockchain network, i.e., $L = l$. P represents the probability of hardware failure occurred at the leader, i.e., $P = p_{\mathrm{hard}}$. N represents the SNR values between the leader and the followers in the blockchain network during data transmission, which can be obtained from (4). E represents the node state of the leader at epoch τ: $E = 0$ means that the leader does not need to be replaced, and $E = 1$ means that the leader needs to be replaced.

2) Action set: In this model, action A taken at epoch τ is defined as the serial number of the selected leader, which can be expressed as follows,

$$A = l, \tag{16}$$

where $l \in \{1,2,...,I\}$. The block packaging interval is represented by t_p and the model generates actions at intervals of t_p. If more than $\lceil I/2 \rceil$ followers return confirmation response in the process of block consensus, the leader is functioning normally and action A_τ remains the same as action $A_{\tau-1}$ at the last epoch. All possible actions form action set $\mathcal{A}$.

3) State transition function: With an action A at a given state S, the following state S' can be determined via the state transition function. The system state $N_{\tau+1}$ at the next epoch can be expressed as follows,

$$N_{\tau+1} = \left\{ \gamma_{l,i}^{\tau+1} | i \in B_{tr} \right\}. \tag{17}$$

According to system states L_τ, $N_{\tau+1}$ and $P_{\tau+1}$, the failure of leader l is examined, thus system state $E_{\tau+1}$ at the next epoch is determined. The specific system state $E_{\tau+1}$ can be expressed as follows,

$$E_{\tau+1} = \begin{cases} 0, \text{normal} \\ 1, \text{error.} \end{cases} \tag{18}$$

Assuming that the selected leader at epoch τ is A_τ, the state transition function of system state L can be obtained according to $E_{\tau+1}$, which can be expressed as follows,

$$L_{\tau+1} = \begin{cases} L_\tau, E_{\tau+1} = 0 \\ A_\tau, E_{\tau+1} = 1. \end{cases} \tag{19}$$

4) Reward function: The instant reward is defined as the block generation latency at epoch τ, which consists of the block packaging latency and the block consensus latency. The reward function is given as follow,

$$R(\tau) = \begin{cases} t_{l,k}^{\mathrm{GEN}} + t_{i,k}^{\mathrm{CON}}, & E_\tau = 0 \\ t_{\max}^{\mathrm{CON}} + t_{z,k}^{\mathrm{GEN}} + t_{i,k}^{\mathrm{CON}}, & E_\tau = 1. \end{cases} \tag{20}$$

B. Leader selection scheme

In the MDP model established in this paper, the system states are discrete and the number of the states is limited. Meanwhile, the size of the action set is determined by the number of strong end-devices. So this paper adopts the DQN algorithm to learn function Q and strategy π^*. The DQN algorithm combines reinforcement learning with deep neural networks and utilizes an experience replay strategy to eliminate the correlation between input sequences. The DQN algorithm includes two neural networks, i.e., the estimation network and the target network. The estimation network is trained in real-time according to a properly defined loss function. The target network uses the weights of the estimation network at intervals to update its network parameters, and the target network is used to calculate the objective function value. The two neural networks are used independently to eliminate the correlation between the estimated Q value and the target Q value. As the central node, the base station collects selection parameters stored in each follower and selects the leader according to strategy π^*, which can be expressed as follows,

$$\pi^*(S, A) = \arg\max_{A \in \mathcal{A}} Q(S, A). \tag{21}$$

The DQN algorithm uses the following formulas to update the parametric loss function $\mathcal{L}(\theta)$ during iterations,

$$\mathcal{L}(\theta) = \mathbb{E}[(Q(S, A, \theta) - Q_{\mathrm{Tar}})^2], \tag{22}$$

$$Q_{\mathrm{Tar}} = R(S, A) + \gamma \max_{A' \in \mathcal{A}} Q(S', A', \hat{\theta}), \tag{23}$$

where θ represents the parameters of the estimation network, $\hat{\theta}$ represents the parameters of the target network, γ is a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3132420, IEEE Internet of Things Journal

discount coefficient, $0 \leq \gamma \leq 1$. S' and A' are the system state and the action at the next epoch. RAFT+ is based on the DQN algorithm, which is shown in Algorithm 1. After sufficient training, neural networks can generate the optimal leader selection strategy.

Algorithm 1 DQN based RAFT+ algorithm

Input: Initialize p_{hard}, Ψ and the number of end-devices. Initialize neural network parameters ε, γ, θ, $\hat{\theta}$, and learning rate α. Set start time of iteration and the total iteration time $\tau_{\max}$. Set update timestep t_e of the target network.

Output: strategy π^*

Initialize a starting action $A = A_0 = 0$ and a starting state $S = S_0 = (M_0, L_0, P_0, N_0, E_0)$, building neural Networks.
while $\tau \neq \tau_{\max}$ **do**
  Update the real-time SNRs of data transmission between end-devices;
  Update the state N according to the SNRs;
  According to the system state, determine whether the system has a hardware failure or communication failure;
  if Hardware failure or communication failure **then**
    Update state E, generate random number e, start leader selection;
    if $e \geq \varepsilon$ **then**
      Generate Q values corresponding to all actions according to the estimation network;
      A new leader is selected according to the (21) from Q values;
    else
      Randomly select new a leader as action A_τ;
    end if
    Update state $L_\tau = A_\tau$;
  else
    Keep the same action as the previous epoch;
    Update state $L_\tau = L_{\tau-1}$
  end if
  Each end-device execute action A_τ and leader observe R and $S_{\tau+1}$;
  Push the observation sequence $(S_\tau, A_\tau, R, S_{\tau+1})$ into the experience replay pool;
  Sample S, S', A and R from the experience replay pool;
  The estimation network and the target network respectively predict Q value according to S and S';
  Update the corresponding Q value in the estimation network based on R and predicted Q values according to (22);
  Update loss function $\mathcal{L}(\theta)$ and parameter θ of the estimation network according to (23);
  if $\tau \bmod t_e = 0$ **then**
    Update the target network parameter $\hat{\theta}$ according to the estimation network parameter θ;
  end if
  Set $\tau + 1 \to \tau$ and $S_{\tau+1} \to S_\tau$;
end while

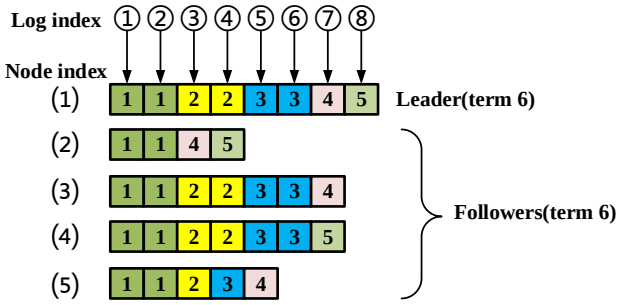

Fig. 4. Example of block inconsistency.

V. Fault-tolerant performance analysis

In the RAFT algorithm, the leader is responsible for block generation and dominating block consensus. After a new block is generated, the leader sends the new block to the followers for endorsement to reach a consensus on the new block. The followers then send confirmation responses containing the endorsement results to the leader. When the leader receives a sufficient number of confirmation responses, an agreement has been achieved for the new block within the blockchain network. A synchronous block update message is reported to the followers by the leader. In the RAFT algorithm, leader selection is triggered by the heartbeat mechanism. If any follower fails to receive the heartbeat packet reported by the leader as scheduled, leader selection will be triggered. Meanwhile, the RAFT algorithm provides a data synchronization mechanism controlled by the leader to solve the problem of block inconsistency between different nodes with less computation required.

A. Block inconsistency

In the blockchain network with the RAFT algorithm as the consensus mechanism, the blocks between nodes may not be synchronized, as shown in Fig. 4. The number in each block indicates the term of the block.

RAFT+ modifies the trigger conditions of leader selection and combines leader selection with block consensus. The leader in the blockchain network sends the new block to the followers to maintain the node state of the followers. The followers endorse the block and return confirmation responses including the endorsement results. If the leader receives more than $\lceil I/2 \rceil$ confirmation responses reported by the followers, an agreement is reached for the generation of the block in the network. During the block consensus process, a follower's failure to return confirmation response means the communication interruption occurs between the follower and the leader, and the follower is not able to update the block synchronously. During the operation of the system, the followers receive new blocks sent by the leader and return confirmation responses containing the endorsement results. After receiving a confirmation response, the leader checks the related data contained in the confirmation response. The data synchronization mechanism is triggered if the leader finds the index of the block contained in the confirmation response is inconsistent with the index stored by itself. To maintain data

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3132420, IEEE Internet of Things Journal

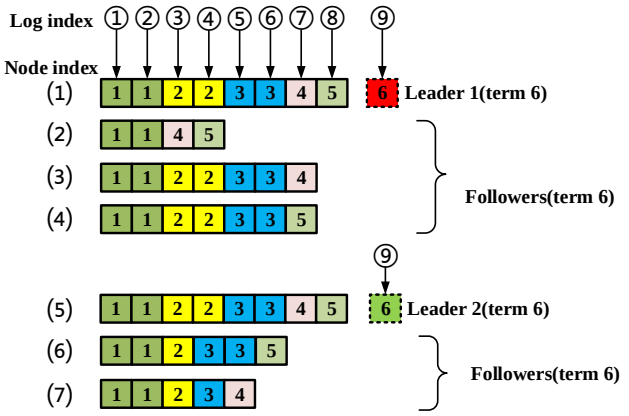

Fig. 5. Example of independent block generation after network fragmentation.

consistency between the leader and the follower, the leader traverses both ledgers to find the closest consistent block. After the closest consistent block is found, the leader sends a command to delete all blocks after that block in the follower's ledger. Then, all blocks in the leader's ledger after that block are sent to the follower to complete the data synchronization.

In RAFT+, leader selection is triggered by hardware failure or communication failure. Communication failure means that more than $\lceil I/2 \rceil$ followers and the leader have communication interruptions. After leader selection is triggered, the central node in the blockchain network collects the parameters of each follower to select a new leader. The new leader collects the ledger of each follower and compares it with its ledger. The different parts of the follower's ledger will be covered according to the leader's ledger. Meanwhile, each follower reports the data that has been reported but not stored in the ledger to the new leader. As a part of block consensus, the new leader generates a block and sends it to the followers. After block consensus is approved, the followers update the block synchronously at the time determined by the new leader. Therefore, RAFT+ can solve the problem of block asynchronization among nodes that may occur in the blockchain network.

B. Network fragmentation

During the operation of an IoT-blockchain system using the RAFT algorithm as the consensus mechanism, it may occur that several followers cannot receive the heartbeat packets at the same time and are converted to candidates. Several candidates send vote requests in parallel to compete for the votes of the followers in the network. The followers then vote for the candidate whose vote request they receive first (the candidate shall meet the requirements of selection parameters, such as term). Each candidate may have a random chance of gaining votes from the followers due to factors like communication environment and distance between nodes. If no candidate receives more than half of the votes, the vote will be invalid and leader selection will be restarted. In this case, the nodes in the blockchain network cannot work normally until a new leader is selected. The RAFT algorithm stipulates that only one leader can exist in a network to avoid network fragmentation. An example of network fragmentation is shown in Fig. 5, where the simultaneous existence of multiple leaders directly results in a split of the network and the independent generation of blocks. After the original network is split into multiple networks, each network will be independent of the others.

The blockchain networks require data consistency between nodes, the block asynchronization caused by network fragmentation is unacceptable. To deal with the problem of network fragmentation in the blockchain networks, RAFT+ modifies the process that several candidates send vote requests in parallel to compete for the votes of the followers in leader selection. Firstly, RAFT+ determines whether to select a new leader based on the failure situation of the original leader. Secondly, RAFT+ considers all followers as candidates for leader selection, so that the optimal leader choice can be made in each term. Finally, RAFT+ selects a central node in the blockchain network to collect the parameters of each follower. The central node selects the unique leader by combining the original leader selection parameters in the RAFT algorithm with the unique parameters under the IoT application scenarios, such as the computing resources of end-devices and time-variant communication environment of the system. This kind of leader selection method can effectively avoid the existence of multiple leaders in the network at the same time, thus preventing network fragmentation.

Based on the above analysis, it can be seen that RAFT+ proposed in this paper can effectively avoid network fragmentation and maintain the fault-tolerant performance of the original RAFT algorithm.

VI. Simulation Results and Analysis

To evaluate the performance of RAFT+, we developed a simulation platform based on python. In the simulations, IoT end-devices collect and report perception data according to preset intervals. Meanwhile, the location of IoT end-devices is fixed and all IoT end-devices are directly connected to the power supply. The amount of data generated in the area within a fixed time is constant, and this paper assumes that the energy consumption generated by each IoT end-device performing the block packaging task is the same. In the blockchain network with RAFT+ as the consensus mechanism, blockchain nodes do not need to participate in mining, thus reducing the number of computing tasks for each node and greatly reducing the system energy consumption. Due to the limited storage resources of IoT end-devices, this paper adopts the method of building a multi-layer blockchain network to control the scale of the local blockchain network to minimize the amount of data that strong end-devices need to store. The number of strong end-devices is determined by the specific simulation settings. Meanwhile, the number of weak end-devices is consistent with the number of strong end-devices. End-devices are randomly distributed in the simulation area with a radius of 100 meters to collect sensory data at fixed intervals. Weak end-devices collect and report the data to the closest strong end-devices, and strong end-devices forward the data to the leader in the blockchain

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3132420, IEEE Internet of Things Journal

TABLE I: System simulation parameters and values

Parameter	Value
Area radius	1 km
Number of strong end-devices	3, 4, ..., 12
Number of weak end-devices	3, 4, ..., 12
CPU cycles of end-devices for hash operations	20, 40, 60 MHz
Number of messages reported by end-devices	2, 3, ..., 11
Hardware failure probability	5%
System bandwidth	10 MHz
Bandwidth for each weak end-device	200 kHz
Bandwidth for each strong end-device	400 kHz
SNR threshold	−20 dB

TABLE II: Neural network parameters and values

Parameter	Value	Parameter description
Layers of neural network	3	Number of layers of neural network
Neurons per layer	128	Number of neurons in each layer of neural networks
Training steps	10^6	Number of training steps per simulation
Learning rate	10^{-3}	Control the learning progress during iterations
Reward_decay	0.9	Discount rate for long term reward
ε_greedy	0.8	Probability of choosing the action with the highest Q value
Replace_target_iter	300	Interval of replace the target network parameters with the estimation network parameters
Memory size	500	Cache pool size
Batch size	32	Size of the sample from cache pool

network. At each packaging time, the leader packages the data received since the last packaging time to generate a block and sends the block to the blockchain network. RAFT+ serves as the consensus mechanism of the blockchain network. The total available bandwidth of the system is 10 MHz. The system bandwidth is used for data reporting by weak end-devices, data forwarding by strong end-devices, and data transmission between the leader and the followers when block consensus occurs. The SNR of data transmission between the leader and each follower is determined by system state N at that epoch. Only the large-scale fading propagation is considered in our simulations, and path loss is considered in the wireless channels. The main simulation parameters of the IoT system are shown in Table I.

Each training round of the system completes 10^6 epochs (steps). In order to achieve optimal simulation results within a long simulation epoch, the learning rate of neural networks in DQN is set as 10^{-3}. The specific parameters of neural networks are listed in Table II.

As the number of iterations increases, the leader selection strategy obtained by neural networks is gradually approaching the optimal strategy. Fig. 6 shows the values of loss under different numbers of end-devices, where I represents the number of strong end-devices in the simulations. The smaller the loss value, the closer the predicted value to the target value. It means that more training steps are required for the convergence of the loss function with the increase of end-devices in the system. Since there are two neural networks in the DQN algorithm for independent training and regular updating of neural network parameters, as well as the SNR values between end-devices are randomly distributed, the loss values fluctuate to a certain extent after the convergence of the

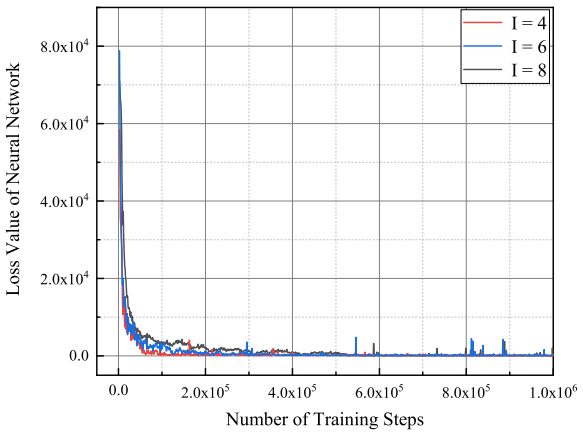

Fig. 6. The convergence of estimation network with different numbers of end-devices.

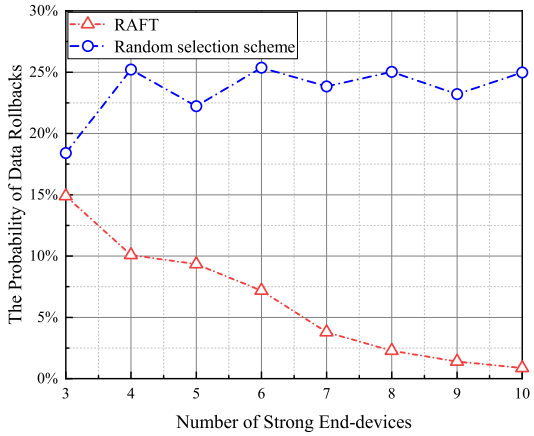

Fig. 7. The probability of data rollback under different schemes.

loss function. Simulation results show that the loss values of the estimation network are from 10^4 to 10^2, and the average number of iterations is about 10^5.

In order to show the impact of different consensus mechanisms of the blockchain network on system performance, RAFT+ is compared with two baseline schemes, i.e.,

- **Random selection scheme:** Before each block packaging moment, the leader is selected randomly from all the nodes in the blockchain network to complete block generation and lead block consensus.
- **RAFT:** The original RAFT algorithm determines whether the leader fails at each block packaging time. In particular, when the original RAFT algorithm is used as the consensus mechanism of the blockchain network, all followers are selected as candidates to make an intuitive comparison. If the original leader fails, the leader is re-selected according to certain parameters, such as term and index.

For the sake of comparison, data rollback is defined as

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3132420, IEEE Internet of Things Journal

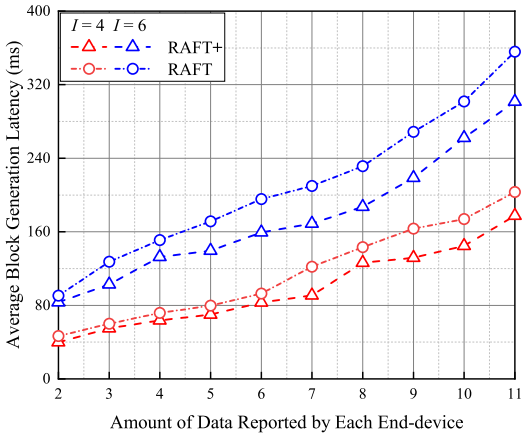

Fig. 8. The average block generation latency with different amount of data reported by each end-device.

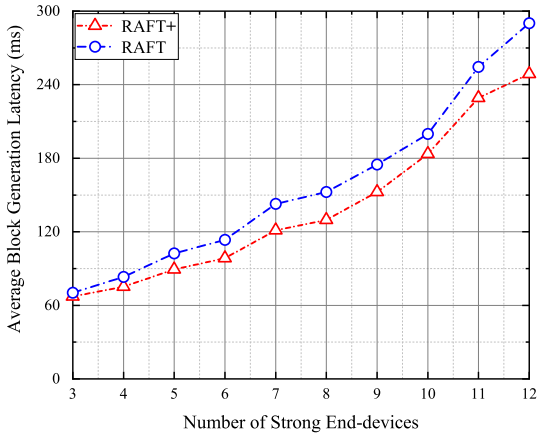

Fig. 9. The average block generation latency with different numbers of strong end-devices.

the case that a follower's data is overwritten due to data inconsistency between the leader and the followers. In the blockchain network, data rollbacks may occur due to hardware failure or communication failure. During the block consensus process, if the leader notices that the blocks stored in a follower are inconsistent with the leader, the inconsistent blocks in the follower will be deleted. Meanwhile, the follower's ledger will be updated to be consistent with the leader's ledger. Fig. 7 shows the probability of data rollbacks under different consensus mechanisms. Due to the unique leader selection scheme, the RAFT algorithm can significantly reduce the probability of data rollbacks with the number of end-devices increases. The random selection scheme cannot limit data rollbacks, resulting in data loss. RAFT+ inherits the characteristics of the RAFT algorithm, which can minimize the possibility of data rollbacks.

Fig. 8 shows the impact of different consensus mechanisms of the blockchain network on the average block generation latency. In the simulations, the amount of data reported by each end-device is gradually increased. In Fig. 8, I represents the number of strong end-devices in the simulations. It is shown that the average block generation latency rises gradually with the amount of data reported by end devices increases. Compared with the RAFT algorithm, RAFT+ can reduce the average block generation latency by 10% to 21%. Meanwhile, with the increased amount of data reported by the end-devices, the impact of RAFT+ on reducing the average block generation latency becomes gradually obvious. The above results indicate that RAFT+ as the consensus mechanism can effectively improve the operating efficiency of the system and the load capacity of the blockchain network.

Fig. 9 shows the impact of different consensus mechanisms of the blockchain network on the average block generation latency under different numbers of IoT end-devices. In the simulations, the number of strong end-devices is increased from 3 to 12, and each end-device reports two data units in each epoch. In the simulations, the computing resources of each strong end-device are independently selected in a

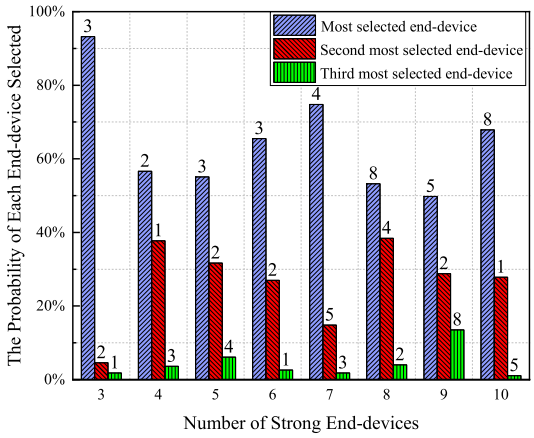

Fig. 10. The possibility of selecting different end-devices as the leader with different numbers of strong end-devices.

certain range, which directly results in the fluctuation of the average block generation latency. Though the amount of data rises with the increase of end-devices, the computing resources of strong end-device still affect the block packaging latency and lead to the fluctuation of the average block generation latency. As the number of strong end-devices is increased, RAFT+ can maintain a low average block generation latency. When IoT end-devices have stronger computing resources, the system efficiency is further improved. RAFT+ provides an approximately 15% performance improvement over the RAFT algorithm. The performance improvement is achieved because the computing resources of strong end-devices and the time-variant communication environment of the system are not taken into account in the RAFT algorithm. However, the leader selection scheme in RAFT+ comprehensively considers the IoT system resources and obtains the optimal strategy through neural network training. It can be seen that the minimum average block generation latency can be obtained by using RAFT+ according to Fig. 8 and Fig. 9. The simulation results

indicate that RAFT+ is scalable. Considering the composition of the local blockchain network and the limited computing resources of IoT end-devices, the number of nodes in the blockchain network is limited. Meanwhile, with the increase of the number of IoT end-devices, the simulation time increases dramatically. From the simulation results, the RAFT+ is capable of adapting to blockchain networks with IoT end-devices within the order of 10^2.

Fig. 10 shows the probability of selecting different strong end-devices as the leader under different numbers of strong end-devices. Numbers on the histograms represent the serial numbers of strong end-devices as the leader. It can be seen that RAFT+ is able to select the optimal leader in the blockchain network according to the computing resources of IoT end-devices and the time-variant communication environment of the system.

VII. Conclusion

In this paper, a multi-layer blockchain architecture was designed to utilize the resources of IoT end-devices while balancing the loads among blockchain networks consisting of end-devices and base stations. The architecture divided an IoT system into three layers to improve the adaptability to the multi-layer blockchain networks. Based on the architecture, RAFT+ was proposed as the consensus mechanism of the blockchain network, which can significantly reduce the communication loss and computing load caused by block consensus. Moreover, RAFT+ incorporated a DQN-based leader selection scheme, which inherits the workflow of the original leader selection mechanism. Meanwhile, the leader selection scheme improved the fault-tolerance performance of the blockchain network and effectively avoided network fragmentation. In this paper, a performance evaluation was conducted based on simulations to eliminate unstable factors during system operation. Simulation results showed that the average block generation latency of RAFT+ was reduced by 10% to 21% compared with the original RAFT algorithm under different conditions. It means that RAFT+ is able to complete block consensus with low latency, so as to improve the efficiency of the system and maintain its stability under high load conditions. In the next steps, the proposed consensus mechanism will be implemented in an IoT-blockchain system in real scenarios to verify the effectiveness.

References

[1] S. Singh, A. S. M. S. Hosen, and B. Yoon, "Blockchain security attacks, challenges, and solutions for the future distributed IoT network," *IEEE Access*, vol. 9, pp. 13 938–13 959, 2021.

[2] Y. Sun, L. Zhang, G. Feng, B. Yang, B. Cao, and M. A. Imran, "Blockchain-enabled wireless Internet of Things: Performance analysis and optimal communication node deployment," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5791–5802, 2019.

[3] L. Hou, K. Zheng, Z. Liu, X. Xu, and T. Wu, "Design and prototype implementation of a Blockchain-enabled LoRa system with edge computing," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2419–2430, 2021.

[4] H.-T. Wu and C.-W. Tsai, "An intelligent agriculture network security system based on private Blockchains," *Journal of Communications and Networks*, vol. 21, no. 5, pp. 503–508, 2019.

[5] S. Pahlajani, A. Kshirsagar, and V. Pachghare, "Survey on private Blockchain consensus algorithms," in *2019 1st International Conference on Innovations in Information and Communication Technology (ICIICT)*, 2019, pp. 1–6.

[6] Y. Xiao, N. Zhang, W. Lou, and Y. T. Hou, "A survey of distributed consensus protocols for Blockchain networks," *IEEE Communications Surveys Tutorials*, vol. 22, no. 2, pp. 1432–1465, 2020.

[7] S. Sinche, D. Raposo, N. Armando, A. Rodrigues, F. Boavida, V. Pereira, and J. S. Silva, "A survey of IoT management protocols and frameworks," *IEEE Communications Surveys Tutorials*, vol. 22, no. 2, pp. 1168–1190, 2020.

[8] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the security and performance of proof of work Blockchains." New York, USA: Association for Computing Machinery, 2016, p. 3–16.

[9] H. Sukhwani, J. M. Martínez, X. Chang, K. S. Trivedi, and A. Rindos, "Performance modeling of PBFT consensus process for permissioned Blockchain network (Hyperledger Fabric)," in *2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)*, 2017, pp. 253–255.

[10] Y. Meshcheryakov, A. Melman, O. Evsutin, V. Morozov, and Y. Koucheryavy, "On performance of PBFT Blockchain consensus algorithm for IoT-applications with constrained devices," *IEEE Access*, vol. 9, pp. 80 559–80 570, 2021.

[11] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm," in *2014 USENIX Annual Technical Conference (USENIX ATC 14)*, 2014, pp. 305–319.

[12] L. Hou, X. Xu, K. Zheng, and X. Wang, "An intelligent transaction migration scheme for RAFT-based private Blockchain in Internet of Things applications," *IEEE Communications Letters*, pp. 1–1, 2021.

[13] P. Danzi, A. E. Kalør, e. Stefanović, and P. Popovski, "Delay and communication tradeoffs for Blockchain systems with lightweight IoT clients," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2354–2365, 2019.

[14] G. Fortino, C. Savaglio, G. Spezzano, and M. Zhou, "Internet of Things as system of systems: A review of methodologies, frameworks, platforms, and tools," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 1, pp. 223–236, 2021.

[15] L. Lei, Y. Tan, K. Zheng, S. Liu, K. Zhang, and X. Shen, "Deep reinforcement learning for autonomous internet of things: Model, applications and challenges," *IEEE Communications Surveys Tutorials*, vol. 22, no. 3, pp. 1722–1760, 2020.

[16] Q. Zhou, K. Zheng, L. Hou, J. Xing, and R. Xu, "Design and implementation of open LoRa for IoT," *IEEE Access*,

vol. 7, pp. 100 649–100 657, 2019.

[17] X. Xiong, K. Zheng, L. Lei, and L. Hou, "Resource allocation based on deep reinforcement learning in iot edge computing," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 6, pp. 1133–1146, 2020.

[18] W. Iqbal, H. Abbas, M. Daneshmand, B. Rauf, and Y. A. Bangash, "An in-depth analysis of IoT security requirements, challenges, and their countermeasures via software-defined security," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10 250–10 276, 2020.

[19] Z. Liu, Q. Zhou, L. Hou, R. Xu, and K. Zheng, "Design and implementation on a LoRa system with edge computing," in *2020 IEEE Wireless Communications and Networking Conference (WCNC)*, 2020, pp. 1–6.

[20] S. Guo, X. Hu, S. Guo, X. Qiu, and F. Qi, "Blockchain meets edge computing: A distributed and trusted authentication system," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 1972–1983, 2020.

[21] Z. Yang, K. Yang, L. Lei, K. Zheng, and V. C. M. Leung, "Blockchain-based decentralized trust management in vehicular networks," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1495–1505, 2019.

[22] S. Malik, V. Dedeoglu, S. S. Kanhere, and R. Jurdak, "Trustchain: Trust management in Blockchain and IoT supported supply chains," in *2019 IEEE International Conference on Blockchain (Blockchain)*, 2019, pp. 184–193.

[23] Y. Yu, S. Liu, P. L. Yeoh, B. Vucetic, and Y. Li, "Layerchain: A hierarchical edge-cloud Blockchain for large-scale low-delay industrial Internet of Things applications," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 7, pp. 5077–5086, 2021.

[24] B. Li, Y. Wu, J. Song, R. Lu, T. Li, and L. Zhao, "Deepfed: Federated deep learning for intrusion detection in industrial cyber–physical systems," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5615–5624, 2021.

[25] G. Wang, Z. Shi, M. Nixon, and S. Han, "Chainsplitter: Towards Blockchain-based industrial IoT architecture for supporting hierarchical storage," in *2019 IEEE International Conference on Blockchain (Blockchain)*, 2019, pp. 166–175.

[26] C. Huang, Z. Wang, H. Chen, Q. Hu, Q. Zhang, W. Wang, and X. Guan, "Repchain: A reputation-based secure, fast, and high incentive Blockchain system via sharding," *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4291–4304, 2021.

[27] S. Alrubei, E. Ball, and J. Rigelsford, "Securing IoT-Blockchain applications through honesty-based distributed proof of authority consensus algorithm," in *2021 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)*, 2021, pp. 1–7.

[28] M. M. Alhejazi and R. M. A. Mohammad, "Enhancing the Blockchain voting process in IoT using a novel Blockchain weighted majority consensus algorithm (WMCA)," *Information Security Journal: A Global Perspective*, pp. 1–19, 2021.

[29] J. Huang, L. Kong, G. Chen, M.-Y. Wu, X. Liu, and P. Zeng, "Towards secure industrial IoT: Blockchain system with credit-based consensus mechanism," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3680–3689, 2019.

[30] C. Li, J. Zhang, X. Yang, and L. Youlong, "Lightweight Blockchain consensus mechanism and storage optimization for resource-constrained IoT devices," *Information Processing and Management*, vol. 58, no. 4, p. 102602, 2021.

[31] H.-S. Choi, G. M. Lee, and W.-S. Rhee, "Hierarchical trust chain framework for IoT services," in *2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN)*, 2019, pp. 710–712.

[32] M. U. Zaman, T. Shen, and M. Min, "Proof of sincerity: A new lightweight consensus approach for mobile Blockchains," in *2019 16th IEEE Annual Consumer Communications Networking Conference (CCNC)*, 2019, pp. 1–4.

[33] S. Biswas, K. Sharif, F. Li, S. Maharjan, S. P. Mohanty, and Y. Wang, "PoBT: A lightweight consensus algorithm for scalable IoT business Blockchain," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 2343–2355, 2020.

[34] K. Wang, C.-M. Chen, Z. Liang, M. M. Hassan, G. M. L. Sarné, L. Fotia, and G. Fortino, "A trusted consensus fusion scheme for decentralized collaborated learning in massive IoT domain," *Information Fusion*, vol. 72, pp. 100–109, 2021.

[35] A. Kaci and A. Rachedi, "Toward a machine learning and software defned network approaches to manage miners' reputation in Blockchain," *Journal of Network and Systems Management*, vol. 28, pp. 478–501, 2020.

[36] S. Khan, W.-K. Lee, and S. O. Hwang, "Aechain: A lightweight Blockchain for IoT applications," *IEEE Consumer Electronics Magazine*, pp. 1–1, 2021.

[37] D. Huang, X. Ma, and S. Zhang, "Performance analysis of the RAFT consensus algorithm for private Blockchains," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 1, pp. 172–181, 2020.

[38] D. Kim, I. Doh, and K. Chae, "Improved RAFT algorithm exploiting federated learning for private Blockchain performance enhancement," in *2021 International Conference on Information Networking (ICOIN)*, 2021, pp. 828–832.

[39] D. Yu, W. Li, H. Xu, and L. Zhang, "Low reliable and low latency communications for mission critical distributed industrial Internet of Things," *IEEE Communications Letters*, vol. 25, no. 1, pp. 313–317, 2021.