doi:10.3772/j.issn.1006-6748.2022.01.002

A load balance optimization framework for sharded-blockchain enabled Internet of Things①

YANG Zhaoxin (杨兆鑫)*, YANG Ruizhe*, LI Meng②*, YU Richard Fei*,**, ZHANG Yanhua*
(*Faculty of Information Technology, Beijing University of Technology, Beijing 100124, P. R. China)
(**School of Information Technology, Carleton University, Ottawa K1S 5B6, Canada)

Abstract

Recently, sharded-blockchain has attracted more and more attention. Its inherited immutability, decentralization, and promoted scalability effectively address the trust issue of the data sharing in the Internet of Things (IoT). Nevertheless, the traditional random allocation between validator groups and transaction pools ignores the differences of shards, which reduces the overall system performance due to the unbalance between computing capacity and transaction load. To solve this problem, a load balance optimization framework for sharded-blockchain enabled IoT is proposed, where the allocation between the validator groups and transaction pools is implemented reasonably by deep reinforcement learning (DRL). Specifically, based on the theoretical analysis of the intra-shard consensus and the final system consensus, the optimization of system performance is formed as a Markov decision process (MDP), and the allocation of the transaction pools, the block size, and the block interval are jointly trained in the DRL agent. The simulation results show that the proposed scheme improves the scalability of the sharded blockchain system for IoT.

Key words: Internet of Things (IoT), blockchain, sharding, load balance, deep reinforcement learning (DRL)

0 Introduction

Recently, with the emergence of blockchain technology, the integration of blockchain and the Internet of Things (IoT) has received great attention in both industry and academia[1-4]. On one hand, IoT is entering a new phase of cross-industry integration, merging data generated from various systems or domains to construct a more powerful industrial sector, and thus requiring a dedicated and large-scale cross-industry platform[5]. However, the current centralized architecture in the IoT network is challenged by single point of failure (SPOF), data privacy, reliability, and robustness[6]. On the other hand, the blockchain emerges by combining cryptography, distributed consensus, and chained-block of data recording. With these advantages of decentralized management and immutable storage, blockchain has been regarded as one of the most reasonable candidates to address the above issues of IoT[7].

Blockchain firstly worked as a cryptography-based decentralized public ledger to store transactions for Bitcoin[8]. Then, along with Ethereum[9], the implementation of decentralized applications with smart contract was developed. It means that the blockchain technology can be used for the management of the interactions in generalized multi-peer systems (e.g., IoT systems). Subsequently, researches put more effort into applying blockchain to IoT networks. For example, Ref. [10] introduced a new IoT architecture to implement task offloading and resource allocation through smart contracts on blockchain. Ref. [11] focused on structural applications incorporating IoT and blockchain into distributed systems. However, with the expansion of blockchain application scenarios, transaction throughput, as an important indicator of system performance, has become the key point of improvement.

Fortunately, sharding comes up with the increase in throughput by parallelizing the verification process of the blockchain[6]. A representative platform is Zilliqa[12] that maximizes the blockchain throughput in proportion to the number of shards. Meanwhile, Ethereum 2.0 proposed a version of a sharded blockchain that splits the entire network into multiple portions.

① Supported by the National Natural Science Foundation of China (No. 61901011) and the Foundation of Beijing Municipal Commission of Education (No. KM202010005017, KM202110005021).
② To whom correspondence should be addressed. E-mail: limeng720@bjut.edu.cn.
Received on June 3, 2021

However, due to the impossible triangle of the blockchain that decentralization, security, and scalability cannot be satisfied at the same time, various approaches have been proposed to balance this three-way trade-off issue. Specifically, many studies have focused on performance optimization schemes using reinforcement learning. Ref. [13] proposed a DQN-based blockchain scheme for the IoT network, which optimizes the blockchain throughput under decentralization and security constraints. In Ref. [14], a sharded-blockchain optimization framework based on DQN maximized the throughout by adjusting the shard number and other blockchain parameters while ensuring consensus security.

Nevertheless, the load unbalance issue exists in the blockchain system due to uneven distribution between the different computing resources of nodes and the different requirements of various blockchain users. On one hand, in order to address the heavy computation load of the mining clusters of the blockchain enabled cellular V2X networks, Ref. [15] proposed a game-theoretic approach for balancing the load at mining clusters while maintaining fairness among offloading vehicles. In addition, Ref. [16] proposed a blockchain-based storage system with financial incentives for load-balancing the data storage between nodes. On the other hand, for the requirement differences of users, for instance, a single user is involved in a great number of transactions, monoxide blockchain system[17] resolves the issue with the co-design of a virtualization applications at the upper layer, which virtualize the user addresses in different shards for load balancing. However, few studies have been published on the load balance between validator groups and transaction pools. There are often significant performance differences between nodes in the real blockchain system, and these differences directly result in an even distribution of nodes with different performance within each shard. Although the random sharding method reduces the risk of a shard being controlled by a malicious node, the large difference between shards usually affects the throughput of the blockchain and, therefore the quality of service for users.

In order to ensure the shards' load balance and improve the system throughput, in this paper, a performance optimization framework for sharded-blockchain enabled IoT is proposed based on deep reinforcement learning (DRL), where the allocation of the validator groups and transaction pools is done according to the computation capacity and transaction load. The contributions of this paper are summarized as follows. To meet the needs of data sharing in large-scale IoT networks, a sharded based permissioned blockchain for IoT is proposed, where transactions are parallelly processed by validator groups. To keep the load balance of different shards, the transaction load requirements is quantified based on the computation cost of intra-shard consensus and its theoretical analysis of performance. Then, the shard forming (allocating the validator groups to transaction pools) and the parameters adjustment (block size and block interval) is formulated as a joint optimization problem and solved by using DRL. Simulation results are presented to show the effectiveness of the proposed scheme.

The rest of this paper is organized as follows. Section 1 describes the system model, followed by the theoretical analysis of consensus protocols in Section 2. Then, the problem formulation is presented in Section 3. Simulation results are shown and discussed in Section 4. Finally, in Section 5, the conclusions and future work are given.

1 System model

In this section, the structure of the sharded blockchain for IoT networks is introduced first, followed by its two-phase consensus.

1.1 Sharded-blockchain based IoT networks

The structure of the blockchain-based IoT network is illustrated in Fig. 1, in which smart devices collect ambient data, which might be shared and processed

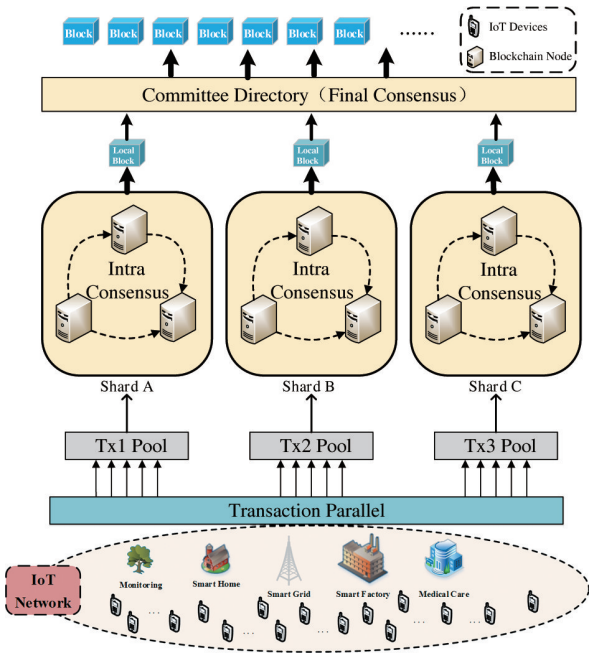

Fig. 1 Blockchain-based IoT network via sharding

among different applications (e.g., smart factory, smart home, smart grid, medical care, monitoring systems, etc.)[14]. For instance, the traffic monitoring data captured by road side units (RSU) and the location information of smart vehicles might be required for route navigation and traffic condition prediction. Here, the sharing of data is recorded in transactions running on a scalable blockchain via sharding, so that secure data storage and reliable data management are implemented.

To meet the needs of a large-scale IoT system, sharding on the blockchain is required to process massive transactions in a parallel manner, where consensus nodes (validators) are clustered into different validator groups to deal with different transactions at the same time[18]. Specifically, validators of size N are divided into a directory committee (DC) containing C nodes and K validator groups in a random manner, and accordingly, transactions are allocated to K validator groups. In each shard, the local blocks composed of local transactions are produced via intra-shard consensus. Then all of these local blocks need to be validated via the final consensus in the DC. Finally, the verified local blocks with no faults will be merged to final blocks and will be distributed to all the blockchain nodes. Note that all of the consensus adopt the practical Byzantine fault tolerance (PBFT) protocol, and the nodes take turns being distributed into different shards, producing a block of size S_B (in bits) in an interval of T^I (in seconds).

1.2 Two-phases PBFT consensus model

The PBFT is a revolutionary protocol to meet the Byzantine general's problems[19]. It can significantly reduce the message complexity of reaching consensus from the exponential level to the polynomial level and still tolerate the proportion of malicious peers (1/3). The classic PBFT protocol mainly consists of three steps: pre-prepare, prepare and commit. Based on the classic PBFT protocol, a two-phase consensus model adapting to sharded-blockchain is shown in Fig. 2[12]. It contains intra-shard consensus and final shard consensus.

The intra-shard consensus using the PBFT protocol within a shard is presented as follows.

(1) The selected primary node in each shard is responsible for collecting the transactions and generating new local blocks as well as broadcasting blocks to the local shard via pre-prepare messages.

(2) Each replica node receives the block and verifies the set of transactions in the block, then exchanges the hash digest with other replica nodes.

(3) The nodes who would receive the most (2/3) same hash digests broadcast its commit message.

(4) All the validators, including the primary node, exchange the commit messages between each other, if the primary node receives the most (2/3) commit messages.

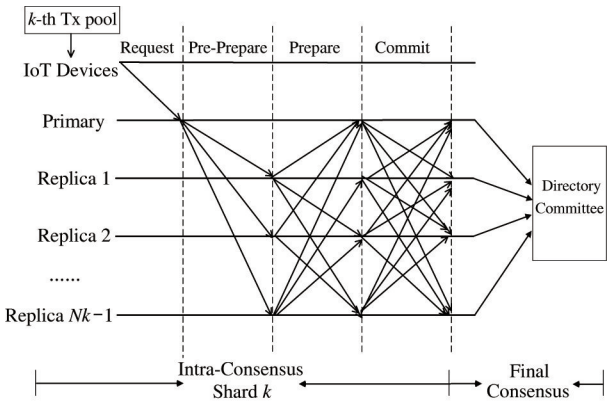

Fig. 2 Two-step consensus structure of shard based blockchain

After the local commit phase, the primary and the replica nodes reply their intra-shard consensus to the DC for the final consensus.

The final consensus in DC also runs the PBFT protocol.

(1) The selected primary node selected in DC network is responsible for collecting the blocks from the shardings, and generating and broadcasting the final blocks.

(2) Each DC replica node receives and verifies the set of transactions in the final blocks, then exchanges the hash digest with other DC replica nodes.

(3) The node who would receive the most (2/3) same hash digests broadcasts a commit message.

(4) All the validators exchange the commit message between each other, if the primary node receives the most (2/3) commit messages.

After the commit phase in DC, the DC nodes reply the final consensus to all the clients in IoT network. Transactions over the two-phase consensus are stored into the blockchain immutably.

2 System performance analysis

In this section, the consideration of the load balance of sharding is presented, and the corresponding theoretical analysis is given. For the clarity of following discussion, the notations are summarized in Table 1.

2.1 Overview of the load balance protocol

As described in Section 1.1, shards are formed by grouping the validators to different transaction

pools. The nodes within one shard produce and verify the local blocks, which contain the block number, block size, signed block summary and transactions. After the intra-consensus, the local block is submitted to the directory committee, where the block header is added if it passes the final consensus.

Table 1 Notations

Symbol	Definition
N_k	The number of nodes within k-th shard
C	The number of nodes within the directory committee
N	The number of nodes in the system
K	The number of shards
x_k	The transaction requests of k-th transaction pool
$\mu^k(c_{k,\min})$	The computing capacity of k-th shard (The least node computation capacity within k-th shard)
$c_{k,p}$	The computing capacity of a primary node with k-th shard
$c_{k,r}$	The computing capacity of a replica node with k-th shard
T_k^{val}	The intra-consensus validation time of the k-th shard
$T_{k,p}^{val}$	The intra-consensus validation time of the k-th shard primary node
$T_{k,r}^{val}$	The intra-consensus validation time of the k-th shard replica node
S_{B_H}	Block header of a local block
S_B	Block size of a local block

It is worth noting that transactions randomly generated in real time may lead to uneven throughput demands in different shards. Meanwhile, the computation capacity of validator groups is different due to the random grouping of consensus nodes. Therefore, the work load of different shards will be unbalanced if a shard is congested with a large number of transactions. Specifically, if a shard with a heavy work load has poor processing capacity (the local primary or replicas have poor computation capacity), the intra-consensus will be delayed and the throughput of the entire blockchain system will be affected. Therefore, the allocation between the validator groups and the IoT transaction pools should be reasonable to ensure the load balance among the shards.

2.2 Load balanced performance theoretical analysis

2.2.1 Intra-shard consensus validation time

Here, the detailed steps and theoretical analysis of the load balance between the shards and the transaction pool is given.

The whole time is partitioned into discrete time periods $T = \{1, \cdots, t, \cdots, \dot{T}\}$, and each time period t has a constant duration $\dot{T}$. In time period t, it is assumed that the k-th shard consists of $N_k (N_k = C = \frac{N}{K+1})$ consensus nodes, and generating or verifying one signature, generating or verifying one MAC require α and β CPU cycles, respectively. As shown in Fig. 2, based on PBFT, the primary node and replica nodes perform the following processing on the transactions submitted by IoT devices.

(1) Request: the IoT devices send requests for block validation to the primary, then the primary verifies one MAC for each transaction request, each request contains one signature that requires verification for each replica during the consensus process.

(2) Pre-Prepare: the primary node processes the requests (x_k requests operated from the k-th transactions pool in time slot) in a single pre-prepared message and forwards the message to all replica nodes. In this phase, the primary node generates $(N_k - 1)$ MACs to send the pre-prepared message, and each replica node needs to verify one MAC.

(3) Prepare: each replica node authenticates the pre-prepare message and generates $(N_k - 1)$ MACs to all the other replicas, and verifies $(N_k - 2)$ MACs when they receive them. Meanwhile, the primary node needs to verify $(N_k - 1)$ MACs received from all the replicas.

(4) Commit: all the validators, including the primary node, exchange the messages between each other, so the primary and any replica first send and then receive $(N_k - 1)$ commit messages, which need to generate $(N_k - 1)$ MACs and verify $(N_k - 1)$ MACs, respectively.

At the end of the commit phase, the primary and all the replica nodes reply their intra-shard consensus to the DC for the final consensus. At this time, the pri-

mary and replica nodes create C MACs for each request.

Note that the primary node performs total x_k signature checks and $x_k(C+1)+4(N_k-1)$ MAC operations, and replica nodes perform a total x_k signature checks and $x_kC+4(N_k-1)$ operations. The computing load of primary nodes and replica nodes are $x_kC\alpha+[x_k(C+1)+4(N_k-1)]\beta$ and $x_kC\alpha+[x_kC+4(N_k-1)]\beta$, respectively.

Thus, the validation time of primary node and replica nodes can be expressed as follows.

$$T_{k,p}^{\mathrm{val}}=\frac{x_kC\alpha+[x_k(C+1)+4(N_k-1)]\beta}{c_{k,p}}$$

and

$$T_{k,r}^{\mathrm{val}}=\frac{x_kC\alpha+[x_kC+4(N_k-1)]\beta}{c_{k,r}} \tag{1}$$

where $c_{k,p}$ and $c_{k,r}$ represent the computing capacity of the primary node and replica node in the k-th shard. Then, the intra-consensus validation time of the k-th shard (T_k^{val}) can be expressed as follows.

$$T_k^{\mathrm{val}}=\min\{T_{k,p}^{\mathrm{val}},T_{k,r}^{\mathrm{val}}\} \tag{2}$$

In addition, in each shard the nodes take turns to be the primary node producing blocks, taking the longest validation time of k-th shard into consideration, which can be calculated by the condition that the node with the least computation capacity in the shard exceeds the computing load of the primary node. Thus, the longest validation time ($\dot{T}_k^{\mathrm{val}}$) is presented by

$$\dot{T}_k^{\mathrm{val}}=\frac{x_kC\alpha+[x_k(C+1)+4(N_k-1)]\beta}{c_{k,\min}} \tag{3}$$

Note that the intra-shard consensus of different shards in parallel, therefore the longest intra-shard validation time of the system $\dot{T}^{\mathrm{val}}$ can be defined as the largest $\dot{T}_k^{\mathrm{val}}$ among K shards.

$$\dot{T}^{\mathrm{val}}=\max\{\dot{T}_1^{\mathrm{val}},\dot{T}_2^{\mathrm{val}},\cdots,\dot{T}_k^{\mathrm{val}},\cdots,\dot{T}_K^{\mathrm{val}}\} \tag{4}$$

2.2.2 Load balanced shard allocation

Based on Eqs(3) and (4), the intra-shard validation time will not be delayed if a transaction pool with a higher transaction load is allocated to a validator group with higher computation capability. In order to avoid the load imbalance, the allocation between validator groups and transaction pools should minimize the system's longest intra-shard validation time $\dot{T}^{\mathrm{val}}$.

The allocation for K validator groups and K transaction pools can be represented as

$$\Delta^K(t)=\{\delta_1(t),\delta_1(t),\cdots,\delta_i(t),\cdots,\delta_K(t)\} \tag{5}$$

and $\delta_i(t)=j(i,j\in\{1,2,\cdots,K\})$ means that the i-th transaction pool is allocated to j-th validators group at time slot t.

Since the nodes in the IoT usually have limited computation resources[20], here, the computation capability of k-th validator groups is modelled as a random variable $\mu^k=c_{k,\min}$ with the node with the least computation capability. Due to various tasks in the IoT system, it's hard to know the computation capability of each node at the next time instant. Assuming the value of computation capability can be partitioned into H discrete intervals, denoted as $\Psi=\{\Psi_0,\Psi_1,\cdots,\Psi_{H-1}\}$, the computation capability of shard k at time slot t can be expressed as $\mu^k(t)$.

Considering the time correlation of computation states in nodes, the transition of computation capability is modelled as a Markov chain. Thus, the $H\times H$ transition probability matrix is defined as $[p_\mu]_{H\times H}$, where $[p_\mu]_{h,h'}=\Pr[\mu^k(t+1)=\Psi_h\mid\mu^k(t)=\Psi_{h'}]$ and $\Psi_h,\Psi_{h'}\in\Psi$. Here, the set of computation capability of K validator groups is donated by $\mathfrak{U}(t)=\{\mu^1(t),\mu^2(t),\cdots,\mu^k(t),\cdots,\mu^K(t)\}$.

Similarly, the transactions generated in k-th transaction pool is also modelled as a random variable χ^k. Assuming the value of transaction load can be partitioned into L discrete intervals, denoted as $X=\{X_0,X_1,\cdots,X_{L-1}\}$, the transaction load of transaction pool k at time slot t can be expressed as $\chi^k(t)$.

Considering the time correlation of the IoT transactions pool, the transition of the number of generated transactions within the k-th transaction pool is modelled as a Markov chain. Thus, the $L\times L$ transition probability matrix is defined as $[p_\chi]_{L\times L}$, where $[p_\chi]_{l,l'}=\Pr[\chi^k(t+1)=X_l\mid\chi^k(t)=X_{l'}]$ and $X_l,X_{l'}\in X$. Here, the set of transaction load of K transaction pools is donated by $\boldsymbol{X}(t)=\{\chi^1(t),\chi^2(t),\cdots,\chi^k(t),\cdots,\chi^K(t)\}$. Algorithm 1 shows the detail of load balance allocation scheme for sharded-blockchain-enabled IoT.

Algorithm 1 Load balance allocation scheme for sharded-blockchain-enabled Internet of Things

Initialization:

Randomly partition of the IoT network into K transaction pools

Randomly grouping of the consensus nodes into K validator groups

Input the transaction load set of K transaction pools $\{\chi^1(t),\chi^2(t),\cdots,\chi^K(t)\}$

Input the computation capability set of K validator groups $\{\mu^1(t),\mu^2(t),\cdots,\mu^K(t)\}$

Input the number of consensus nodes N

Initialize the shard allocation:

$$\Delta^K(t)=\{\delta_1(t),\delta_1(t),\cdots,\delta_i(t),\cdots,\delta_K(t)\}$$

for $t = 1,2,\cdots,O$ do
  for $\delta_i(t) = j$ do
    Allocate the i-th transaction pool to j-th validator group
    Load balanced allocation by minimizing the system's longest intra-shard validation time $\dot{T}^{\text{val}}$
  End for
  Adjust the block sizes and block intervals by maximizing the throughput $\Theta(S_B, T^I)$
  DQN optimization of the allocation and the parameters adjustment
  Apply the training result to the sharded-blockchain enabled IoT system
End for

2.2.3 Scalability performance

The scalability of a sharded-blockchain system can be evaluated by system throughput, which is the number of transactions packed into local blocks in a unit time. Usually, the throughput of the traditional blockchain is affected by two performance parameters: block size and block interval. Noting that the sharded-blockchain produces blocks in a parallel manner, the throughput is K times increased, which can be expressed as[14]

$$\Theta(S_B, T^I) = \frac{K[(S_B - S_{B_H})/\lambda]}{T^I} \tag{6}$$

where S_B represents the local block with a maximum size (bytes) for each block interval period T^I, S_{B_H} and λ are the block header of the local block and the average transaction size respectively. Considering the fixed shard number, it can be found from Eq. (6) that increasing the block size or reducing the block interval can increase the throughput.

3 Problem formulation

In order to keep the load balance and improve the throughput of the sharded blockchain, it's necessary to jointly optimize shard allocation, block size, and block interval. To implement the DRL approach, the system performance optimization problem is formulated as Markov decision process (MDP).

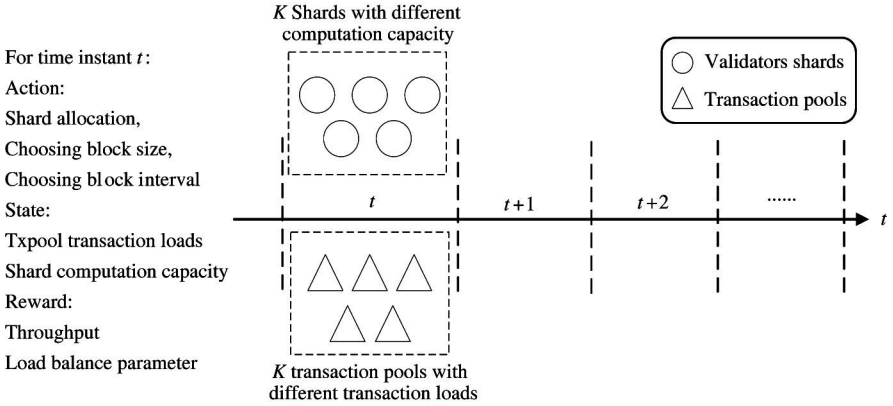

Fig. 3 MDP of load balanced sharded blockchain

3.1 MDP formulation

3.1.1 State space

Due to the fact that the learning agent makes decisions about the allocation between the validator groups and the transaction pools, the system state $S(t)$ at time instant $t(t = 1,2,3\cdots)$ can be expressed as

$$S(t) = [\mathfrak{U}(t), \boldsymbol{X}(t)] \tag{7}$$

where $\mathfrak{U}(t)$ represents the set of computation capability of K shards and $\boldsymbol{X}(t)$ is the set of transaction load of K transaction pools.

3.1.2 Action space

In order to optimize the shard allocation and maximize the throughput, several parameters of the blockchain system should be adjusted to adapt to the dynamic environment, which includes the shard allocation $\Delta(t)$, block size S_B and block interval T^I. Thus, the action space at decision epoch t is expressed by

$$A(t) = [\Delta^K(t), S_B, T^I] \tag{8}$$

where $\Delta^K(t)$ represents the allocations between transaction pools and shards at time slot t. Additionally, using limited fractional methods, block size space $S_B \in \{2,4,\cdots,\dot{S}\}$ and block interval $T^I \in \{0.5,1,\cdots,\dot{T}\}$ have a maximum block size limit $\dot{S}$ and maximum block interval $\dot{T}$.

3.1.3 Reward function

The reward function is defined to maximize the blockchain throughput while minimizing the longest system intra-shard validation time, and a decision should be made in each epoch to solve the following problem.

$$\text{Objective}: \max_{A} Q(S, A) \tag{9}$$

where $Q(S, A)$ is the action-value function calculated by

$$Q(S, A) = \mathbb{E}\left[\sum_{t=0}^{\infty} \sigma^t R(t)(S(t), A(t)) \mid S^0 = S, A^0 = A\right] \tag{10}$$

with the discount factor $\sigma \in (0, 1]$ that reflects the trade-off between the immediate and future rewards, and the immediate reward is defined as

$$R(t)(S(t), A(t)) = \omega_L \frac{1}{\dot{T}^{\text{val}}} + \omega_T \Theta(S_B, T^t) \tag{11}$$

where the $\omega_L, \omega_T \in (0, 1]$ are the weights coefficient for the trade-off between throughput rewards and the load balance rewards in order to get a reasonable training result. And there is $\omega_L + \omega_T = 1$.

3.2 DQN-based optimization

Based on the formulation in Section 3.1, a DRL approach is used to optimize the reward. Specifically, a DNN-based Q-learning approach is applied to perform complicated function approximation, which is known as DQN technology[21].

DQN is an improved version of Q-learning that uses deep networks to approximate the action-value function $Q(S, A)$ instead of using a Q-table to store $Q(S, A)$, which is able to approximate the value function accurately while addressing a large volume data dimension. Algorithm 2 shows the optimization framework of system performance based on DQN.

Algorithm 2 DQN-based optimization framework for load-balanced sharded blockchain

Initialization:

Initialize the system state of the sharded blockchain $S(t)$

Initialize replay memory D with the capacity Z

Initialize main Q network of $Q(S, A; \theta)$ with random weights θ

Initialize target Q network of $Q^-(S, A; \theta^-)$ with weights $\theta = \theta^-$

Load initial state space $S(t)$ data and use it as the input of the main deep Q network

Input maximum training episode M, maximum training step $\dot{T}$

DQN learning process:

For episode $= 1, \cdots, M$ do

for $t = 1, \cdots, \dot{T}$ do

Select a random probability p

if $p < \varepsilon$ then do

Select a random action

else

$A(t) = \arg\max_{A} Q(S, A; \theta)$

end if

Decrease exploration probability ε

Execute action $A(t)$ to select allocation and adjust block size and block interval, and observe reward $R(t)$ and proceed to next state $S(t+1)$

Store the experience $(S(t), A(t), R(t), S(t+1))$ into the replay memory D

Randomly sample a mini-batch of state transition $(S(i), A(i), R(i), S(i+1))$ from the replay memory D

Set $y_i = r_i + \gamma \max_{A'} Q(S_{i+1}, A'; \theta^-)$ to compute the target Q^- value from the target Q network

Update target Q network by performing the gradient descent of loss function

$L(\theta) = (y_i - Q(S_i, A_i; \theta))^2$ for every $\mathbb{G}$ step

End for

End for

4 Simulation results and discussions

In this section, the computer simulation is used to demonstrate the effectiveness of the proposed scheme. The simulation settings are presented. The related parameter settings are illustrated in Table 2.

4.1 Simulation setting

The computation capability of the validator is formulated as a Markov model. Assuming that the state of the computation capability can be very high, high, medium, and low, whose transition probability matrix $[p_\mu]_{H \times H}$ is set as

$$p_\mu = \begin{bmatrix} 0.45 & 0.3 & 0.15 & 0.1 \\ 0.3 & 0.45 & 0.15 & 0.1 \\ 0.15 & 0.3 & 0.45 & 0.1 \\ 0.15 & 0.1 & 0.3 & 0.45 \end{bmatrix} \tag{12}$$

Table 2 Parameters of the system simulation environment

Parameter name	Description
The number of nodes, N	100
The number of shards, K	4
Maximum block size[14], $\dot{S}$	8 MB
Limit block interval[13], $\dot{T}$	16 s
The transaction requests of transaction pool, x_k, $(X_l, X_{l'} \in X)$	100 – 300
The computing capacity of nodes[13], μ^k, $(\Psi_h, \Psi_{h'} \in \Psi)$	10 – 30 GHz
Block header[14], S_{B_H}	80 Bytes
Computing cost for verifying signatures[13], α	2 MHz
Computing cost for generating/verifying MACs[13], β	1 MHz

Similarly, assuming the state of transaction load of transaction pool can be very high, high, medium, and low, the transition probability matrix $[\boldsymbol{p}_\chi]_{L\times L}$ is set by

$$\boldsymbol{p}_\chi = \begin{bmatrix} 0.45 & 0.32 & 0.16 & 0.07 \\ 0.32 & 0.45 & 0.16 & 0.07 \\ 0.16 & 0.32 & 0.45 & 0.07 \\ 0.16 & 0.07 & 0.32 & 0.45 \end{bmatrix} \tag{13}$$

For comparison, three baseline schemes are considered in the simulation as follows.

(1) Proposed scheme without load balanced allocation: the K transaction pools are allocated to K shards in a random manner.

(2) Proposed scheme with fixed block size: the blocks generated by the block producers at different intervals with the same size (2 MB).

(3) Proposed scheme with fixed block interval: the frequency of issuing blocks is fixed (every 10 s).

4.2 Simulation results and discussion

4.2.1 Convergence trend analysis

Fig. 4 shows the convergence performance of the long term reward under the proposed DQN-based load balanced sharded blockchain optimization scheme, where the y-axis denotes the long term reward and the x-axis denotes the training episodes. From Fig. 4 it can be observed that the convergence trend of the long term reward with the learning rate of 0.01 is not obvious. With the learning rate decreasing to 0.001, the convergence speed of the network is accelerated. The result is ideal when the learning rate is reduced to 0.0001, where the convergence is evident and the long term reward is higher than the others.

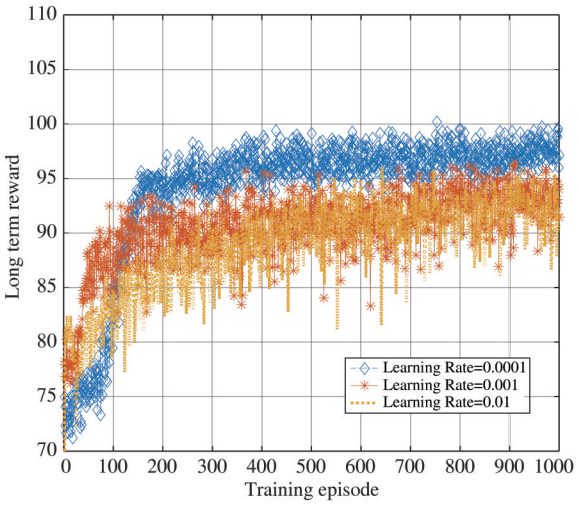

Fig. 4 Long term reward under different learning rates

In addition, it can observed that the long term reward is lower at the beginning of the learning process. However, with the increase of episodes, the long term reward increases and reaches a stable state after around 2000 episodes, which means that the agent has learned the optimal strategy to maximize long-term reward. The convergence verifies the effectiveness of the proposed scheme.

4.2.2 Load balance performance analysis

Fig. 5 describes the load balance performance with the proposed DRL-based performance optimization scheme. The longest intra-shard validation time of the proposed scheme decreases and reaches stable after 2000 episodes, which means the shard allocation became reasonable. Compared with the traditional random scheme, the proposed scheme can be verified to receive a more reasonable allocation according to the simulation result.

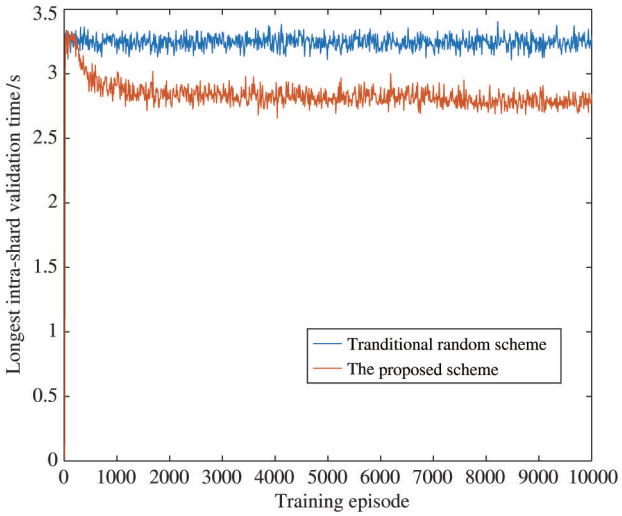

Fig. 5 Comparison of longest intra-shard validation time

Fig. 6 shows the relationship between system performance and the number of validators. One observation is that the validation time increases with the validators number increasing. With more validators joining the consensus process, the computing load of each transaction increases, which naturally leads to a decrease in system performance. In addition, focusing on the comparison with the traditional scheme, the proposed scheme obtains less validation time with the variation of validator number. The reasons are described as before.

Fig. 7 shows the longest intra-shard validation time of the proposed scheme under different shard number K. It can be found that latency of intra-shard consensus decreases with the increase of shard number. The reason is that with the increasing number of shards, the consensus nodes within a shard decreases, which reduces the computation cost for the intra-shard consensus validation. In addition, the transaction load of each transaction pools decreases with the increase of shard

number. Therefore, the load balanced shard allocation can be got faster with more shard number.

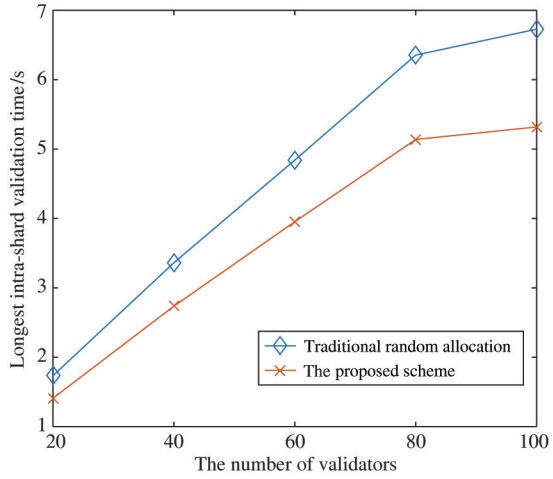

Fig. 6 Longest intra-shard validation time vs the number of validators

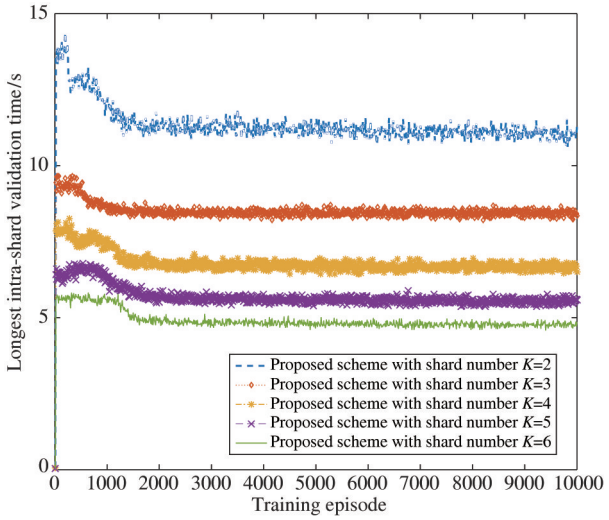

Fig. 7 Longest intra-shard validation time under different shard numbers

4.2.3 Throughput performance analysis

Fig. 8 shows the throughput performance of the proposed scheme under different baselines. It can be observed that the throughput is lower at the beginning of the learning process. However, with the increasing number of episodes, the throughput increases and reaches a stable state after around 2000 episodes, which verifies the convergence performance of the proposed scheme. In addition, it can be also found that the proposed scheme can receive higher throughput than that of the other baselines, which shows the advantage of the proposed framework.

Fig. 9 shows the throughput performance of the proposed scheme under different shard number K. It is obvious that the convergence of throughput slows down as the number of shards increases. The reason is that with the increasing number of shards, the action space of MDP becomes larger. For instance, when the shard number is 4, which means that 4 transaction pools should be allocated to 4 validator groups, thus it has totally 24 ($4 \times 3 \times 2 \times 1$) allocation methods. Then, the action space for the shard allocation is 24. When the shard number is 5, the action space of the shard allocation becomes 120. Therefore, the DQN agent needs more training episode to explore a more reasonable action.

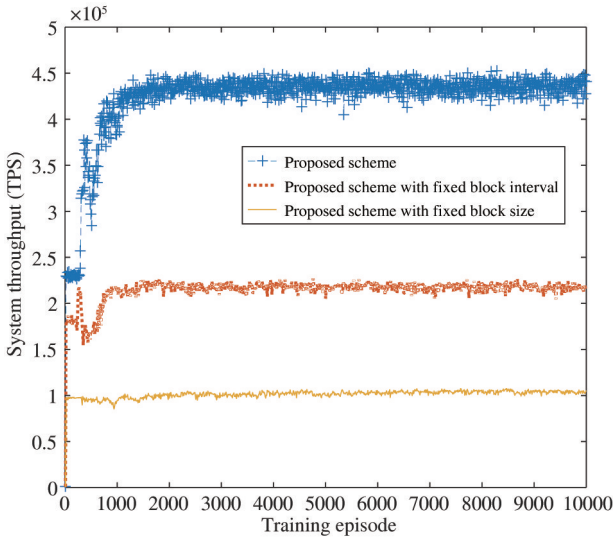

Fig. 8 Throughput performance under different baseline

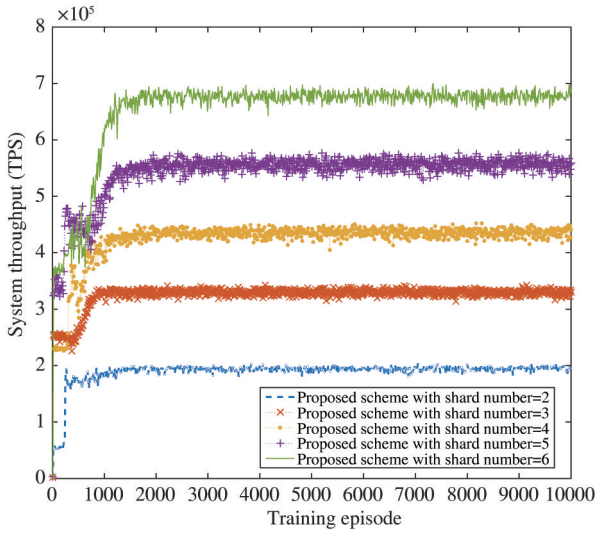

Fig. 9 Throughput performance under different shard numbers

Fig. 10 shows the relationship between system performance and the average transaction size. It can be observed that with the average transaction size increasing, the throughput of the system decreases. The reason is that the number of transactions contained in one

block decreases when the average transaction size increases. Focusing on the comparison of different schemes, the proposed scheme maintains the highest throughput with the different average transaction size, followed by the proposed scheme with fixed interval and block size. The reason is that the proposed scheme without any limiting factors is able to adjust a reasonable blockchain parameters to improve throughput.

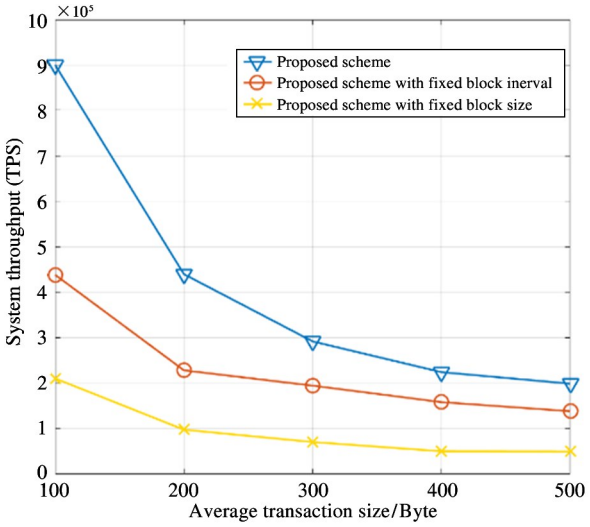

Fig. 10 Throughput vs. transaction size

5 Conclusions

In this paper, a DRL-based load-balanced sharded-blockchain for the IoT network is proposed, where the allocation between the validator clusters and the transaction pools is load balanced and the scalability is improved. With sharded blockchain, the large scale data sharing of the IoT network is stored immutably and managed reliably. In the proposed framework, a theoretical analysis of the performance of sharded blockchain system is provided first. Then the load balanced allocation is optimized by the constraint of the shard's longest validation time and by maximizing the load balanced parameter using the DQN approach. In addition, the throughput is maximized by adjusting the block size and block interval with DQN. Simulation results demonstrate the proposed framework can achieve reasonable load balanced allocation and higher throughput than the baselines with various system parameters. Future work is in progress to consider the optimization of shard number with other DRL approaches based on the proposed framework.

References

[1] LI S, LI D X, ZHAO S. The Internet of Things: a survey [J]. *Information Systems Frontiers*, 2015, 17(2): 243-259

[2] KANG J, RONG Y, HUANG X, et al. Blockchain for secure and efficient data sharing in vehicular edge computing and networks[J]. *IEEE Internet of Things Journal*, 2019, 6(3): 4660-4670

[3] HONG L, ZHANG Y, TAO Y. Blockchain-enabled security in electric vehicles cloud and edge computing[J]. *IEEE Network*, 2018, 32(3): 78-83

[4] KANG J, RONG Y, HUANG X, et al. Enabling localized peer-to-peer electricity trading among plug-in hybrid electric vehicles using consortium blockchains[J]. *IEEE Transactions on Industrial Informatics*, 2017, 13(6): 3154-3164

[5] TONG W, DONG X, SHEN Y, et al. A hierarchical sharding protocol for multi-domain IoT blockchains[C]// 2019 IEEE International Conference on Communications (ICC), Shanghai, China, 2019: 1-6

[6] QIU C, YAO H, YU R F, et al. A service-oriented permissioned blockchain for the Internet of Things[J]. *IEEE Transactions on Services Computing*, 2020, 13(2): 203-215

[7] NOVO O. Blockchain meets IoT: an architecture for scalable access management in IoT[J]. *IEEE Internet of Things Journal*, 2018, 5(2): 1184-1195

[8] WOOD G. Ethereum: a secure decentralised generalised transaction ledger[J]. *Ethereum Project Yellow Paper*, 2014, 151(1): 1-32

[9] NAKAMOTO S. Bitcoin: a peer-to-peer electronic cash system[EB/OL]. http://www.bitcoin.org/bitcoin.pdf: Bitcoin, [2020-06-03]

[10] XIAO K, GAO Z, SHI W, et al. EdgeABC: an architecture for task offloading and resource allocation in the Internet of Things[J]. *Future Generation Computer Systems*, 2020, 107(6): 498-508

[11] HOSSEINIAN H, SHAHINZADEH H, GHAREHPETIAN G B, Z. et al. Blockchain outlook for deployment of IoT in distribution networks and smart homes[J]. *International Journal of Electrical and Computer Engineering*, 2020, 10(3): 2787-2796

[12] ZILLIQA Team. The Zilliqa technical whitepaper[EB/OL]. https://zilliqa.com: Zilliqa, 2019,[2020-04-20]

[13] LIU M T, YU F R, TENG Y, et, al. Performance optimization for blockchain-enabled industrial Internet of Things (IIoT) systems: a deep reinforcement learning approach[J]. *IEEE Transactions on Industrial Informatics*, 2019, 15(6): 3559-3570

[14] YUN J, GOH Y, CHUNG J M. DQN-based optimization framework for secure sharded blockchain systems[J]. *IEEE Internet of Things Journal*, 2021, 8(2): 708-722

[15] JAMEEL F, JAVED M A, ZEADALLY S, et al. Efficient mining cluster selection for blockchain-based cellular V2X communications[J]. *IEEE Transactions on Intelligent Transportation Systems*, 2020, 22(7): 4064-4072

[16] YIN H, ZHANG Z, ZHU L, et al. A blockchain-based

storage system with financial incentives for load-balancing [J]. *IEEE Transactions on Network Science and Engineering*, 2020,8(2): 1178-1188

[17] WANG J, WANG H. Monoxide: scale out blockchains with asynchronous consensus zones [C] // The 16th USENXI Symposium on Networked Systems Design and Implementation (NSDI 19), Boston, USA, 2019: 95-112

[18] LUU L, NARAYANAN V, ZHENG C, et al. A secure sharding protocol for open blockchains [C] // 2016 ACM Special Interest Group on Security, Audit and Control Conference (SIGSAC), Vienna, Austria, 2016: 17-30

[19] MIGUEL O, BARBARA L. Practical Byzantine fault tolerance [C] // Proceedings of the 3rd Symposium on Operating Systems Design and Implementation (OSDI), New Orleans, USA, 1999: 1-172

[20] ZHU C, LI X, LEUNG V, et al. Towards pricing for sensor-cloud [J]. *IEEE Transactions on Cloud Computing*, 2020,8(4): 1018-1029

[21] MNIH V, KAVUKCUOGLU K, SILVER D, et al. Playing Atari with deep reinforcement learning [C] // Proceedings of Neural Information Processing Systems, Lake Tahoe, USA, 2013:1-9

YANG Zhaoxin, born in 1993. He is currently pursuing a Ph. D degree at Beijing University of Technology. From October 2019 to April 2020, he visited Carleton University, Ottawa, ON, Canada, as a visiting Ph. D student funded by Beijing University of Technology. He received his B. S. degree and M. S. degree from Beijing University of Technology in 2015 and 2018, respectively. His current research interests include big data, blockchain, and machine learning.