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/TCC.2019.2930259, IEEE Transactions on Cloud Computing

Cloud Computing Assisted Blockchain-Enabled Internet of Things

Chao Qiu, *Student Member, IEEE,* Haipeng Yao, *Member, IEEE,* Chunxiao Jiang, *Senior Member, IEEE,* Song Guo, *Senior Member, IEEE,* and Fangmin Xu.

This is a resubmitted manuscript, which is approved by the EIC.

***Abstract*—Recently, the term 'Internet of Things' (IoT) has garnered great attention. As a trusted, dependable, and decentralized approach, blockchain has already been used in IoT. However, the existing blockchain has a number of drawbacks that prevent it from being used as a generic platform for IoT. The nodes in IoT are heavily resource-limited, especially computing and networking resources. Unfortunately, they are necessary for the blockchain to solve complicated puzzles and propagate blocks. In this paper, we propose agent mining and cloud mining approaches to solve the above problem in the blockchain-enabled IoT. To be specific, miners act as mining agents for nodes in IoT, offload mining tasks to cloud computing servers, and use networking resources dynamically. Furthermore, in order to enhance the performance, the access selection of users, computing resources allocation, and networking resources allocation are formulated as a joint optimization problem. We then propose a dueling deep reinforcement learning approach to address this problem. Numerical results justify the effectiveness of our proposed scheme.**

***Index Terms*—Internet of Things (IoT), blockchain, cloud mining, proof of work, dueling deep reinforcement learning.**

I. Introduction

Recently, the term 'Internet of Things' (IoT) is an emerging paradigm to meet the demands of flexibility, agility, and ubiquitous accessibility through high-throughput and low-latency communications. The core idea of IoT is to sense and gather data from surroundings, as well as perform self-acting functions for users. Moreover, a sharp growth in the number of IoT devices will happen in the next few years, i.e., from 780 million devices in 2016 to 26 billion devices in 2021 [1]. A series of emerging applications aided by IoT will also appear, such as smart home [2], smart grid [3], smart transportation [4], and smart manufacturing [5].

The improvement of IoT needs innovative information and communication technologies (ICTs), including standardized protocols, proper architectures, etc. As shown in the work of [6], a centralized mainframe architecture was used in the past version of IoT. Currently, a cloud-based centralized architecture is being used. However, confidential and private information may be gathered by smart devices in IoT, and many safety problems have occurred to threaten the current IoT architecture. This is due to the fact that the current architecture is heavily restricted with the problem of single point of failure (SPOF), data privacy, security, robustness, etc. Existing methods for supporting secure, reliable, and private data handling need third-party entities. In such a scenario, users have to trust these entities to handle their data. Nevertheless, misusing and surveillance still exist. Therefore, a number of non-trivial problems in the current IoT architecture prevent it from being used as a generic platform for various applications. These factors have fueled the need to explore a trusted, dependable, and decentralized approach for the widespread development of IoT. The next step of IoT will be the distribution among multiple peers. Here, blockchain may be a viable option to help.

Since the advent of Bitcoin protocol in 2008 [7], a large wave of blockchain has emerged. It works as a decentralized public ledger to store and share the records of transactions, supporting the implementation of decentralized applications, and supervising the interactions in multiple peers systems. Additionally, blockchain relies on a heavy utilization of cryptography to support trusted systems without centralized authorities, where untrusted individuals can interact with each other in a reliable mode. Thus, smart devices in IoT can be interconnected with each other in a distributed blockchain manner, safely and reliably. Many researches have arisen in designing blockchain-enabled IoT, ranging from industrial communities to academic communities. For example, IBM has considered blockchain as a promising technology for future IoT [8]. Moreover, numerous works have used blockchain to enable peer-to-peer communication in IoT [9], [10]. In the blockchain, the participants (miners) need to solve a cryptographic puzzle. Only the winner could issue a block and reach consensus with the rest of miners. Such an approach is friendly scalable and secure, but at the cost of computing resources and consensus speed.

However, the existing blockchain has numerous drawbacks that prevent it from being used as a generic platform in IoT [11]. The nodes in IoT are heavily resource-limited, especially computing resources and networking resources. It is challenging for them to solve such complicated puzzles and propagate blocks among all miners, which may result in the poor performance of the system. Inspired by cloud/fog mining in [12] and band renting in [13], [14], we consider agent mining and cloud mining in this paper. Here, miners act as mining agents for nodes in IoT, offload mining tasks

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/TCC.2019.2930259, IEEE Transactions on Cloud Computing

to the nearby cloud computing servers, and rent networking resources dynamically. In this way, the utility of miners can be improved, so as to increase the performance of blockchain-enabled IoT.

In this paper, we study a blockchain-enabled IoT. Given the limited resources in IoT, we use agent mining and cloud mining in the blockchain-enabled IoT. We consider the performance of the system is constrained with respect to users' service demands, computing capabilities, and networking resources. Accordingly, in order to enhance the performance, the access selection of users, computing resources allocation, and networking resources allocation are formulated as a joint optimization problem. We then propose a dueling deep reinforcement learning approach to solve this problem. Finally, simulations are shown to demonstrate the effectiveness of our proposed scheme.

The rest of this paper is organized as follows. Related work is presented in Section II. System description and model are shown in Section III. In Section IV, we formulate the joint problem as a Markov decision process, with the definition of reinforcement learning tuples. Additionally, the dueling deep reinforcement learning approach is used to solve this problem. We show simulation results in Section V. Finally, Section VI provides some concluding remarks.

II. Related Work

In this section, we review the journey of IoT, and blockchain, along with two types of consensus protocols used in blockchain. We then give a brief survey about the current researches of consensus protocols.

A. Overview of IoT

IoT is the network to expand the interconnection among physical devices, embedded with sensors, electronics, and software, enabling these devices to connect, collect, and exchange data [15].

We then take a look at the evolutions of IoT from 1982 until now in Fig. 1. With the connection between a modified Coke machine and the Internet, the first IoT appeared at 1982 [16]. In the early years of IoT, a centralized and closed mainframe architecture was used to connect each physical device. With the emergence of social network site (SNS), such as Facebook and Twitter, a centralized and open accessed cloud architecture is being used currently. In the future, the age of the smart IoT will come. Some open problems are introducing unprecedented challenges to the current IoT, including security, privacy, and extensible architecture [17]. Lots of researches have advocated that blockchain will rise to meet these new challenges [9], [10], [18].

B. Overview of Blockchain

With the emergence of Bitcoin in late 2008 [7], a plenty of attention has been attracted to the blockchain. It is a

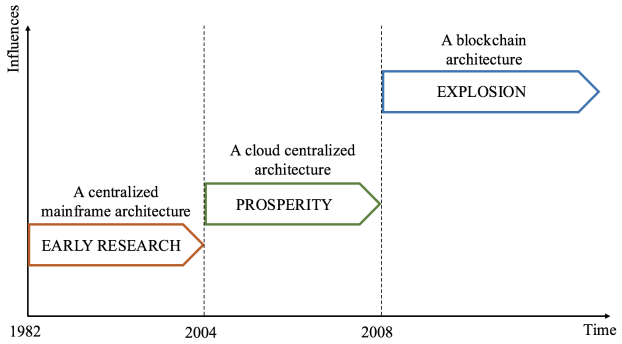

Fig. 1: Evolutions of IoT.

distributed system to support reliable services to numerous nodes that do not fully trust each other. All nodes collect transactions, forming a block, encrypted by public key, message authentication codes (MACs), signatures, and collision-resistant hashing to provide high-level security protection. After validated by the whole nodes, the authenticated block is appended into the chain. Here, consensus protocols ensure that the whole nodes agree on a unique order where this block is appended [19]. Blockchain enables to achieve consensus among nodes without a central supervisor, with the features of dependability, security, and large system scale. Therefore, blockchain may be a potential approach to solve the above problems in the current cloud-centric IoT.

Thus, as we can see, the consensus protocol is an important process to provide a guaranteed ordering of transactions and blocks. Consensus protocols implemented in blockchain vary, which can be classified into the working-based protocols and the replica-based protocols, as shown in the following.

- In the working-based protocols, lots of miners use their computing power to solve a hard computing puzzle. Only the winner of them could issue a block and reach consensus with the rest of participants. These protocols always appear coupled to cryptocurrencies, e.g., Bitcoin [7] and Ether in Ethereum [20]. The typical protocols include PoW, PoS, DPoS, and PoET. They are beneficial of that they could scale to a large number of nodes, at the cost of a large amount of computing resource and time. Therefore, they are widely used in permissionless blockchain, such as Bitcoin and Ethereum.
- In the replica-based protocols, *Byzantine* nodes, which are subverted by adversaries and against the common target of achieving agreements maliciously, are tolerated. State machine replication mechanisms are utilized to deal with the *Byzantine* nodes, by transferring the replica of a block to others. And when a certain number of nodes validate it, consensus and finality will exist and occur. The typical protocols include PBFT, RBFT, and Paxos. Their advantages are low costs and low latency. However, due to the fact that these consensus

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/TCC.2019.2930259, IEEE Transactions on Cloud Computing

protocols rely on a *primary* node to order a block, which can be smartly malicious, they are not enough robust. Thus, they are always used in the partially trusted environment, e.g., permissioned blockchain, such as Hyperledger Fabric [21].

For a comprehensive perspective, we offer an at-a-glance view of the main considerations, pros, and cons of these two types of consensus protocols in Table I.

C. A Brief Survey about Current Researches

Recently, some excellent works have used blockchain in IoT. However, blockchain is still in its nascent stage and being explored to adopt in the best possible ways. There are a number of drawbacks that prevent it from being used as a generic platform in IoT. The most intractable one is that using the working-based consensus protocols in IoT imposes too many costs in term of computing resources and time. Unfortunately, IoT is heavily resource-limited, and there are lots of tasks that need to be computed and delivered in the working-based consensus protocols. Currently, a number of excellent works have been done to solve the issue.

Generally speaking, a majority of researches have advocated offloading the heavy computing tasks to cloud servers to relieve IoT nodes, where different approaches have been used to solve the problem of where to offload computing tasks. Specifically, Xiong *et al.* in [12] used a game theoretic method to solve the optimal revenue of cloud/fog servers and blockchain nodes. Pan *et al.* in [22] considered the integration of a permissioned blockchain with currency system, called EdgeChain. Here, a credit-based approach was utilized to allocate computing resources. Additionally, Liu [23] used an alternating direction method of multipliers based approach to solve the problem of offloading strategy.

Although the above works have solved the problem of computing resources allocation, they all ignore the access selection of users, and networking capabilities among miners, which are also viable ways to improve the utility of miners, i.e., the users with the higher service demands have the higher successful probability of mining a block, and miners using the better networking resources have the lower block propagation delay. Therefore, in this paper, we integrate cloud mining with agent mining in IoT, where computing tasks are offloaded to cloud nodes, and miners act as mining agents for users in IoT. In order to improve the expected utility of miners, we jointly consider access selection of users, computing resources allocation, as well as networking resources allocation, and formulate them as a joint optimization problem.

III. System Description and Model

In this section, we present system description. Based on the system description, we think the performance of the system is constrained with respect to users' service demands, computing capabilities, and networking resources. We then give mining reward model, computing model, and networking model.

A. System Description

We consider a permissionless blockchain without any centralized authorities, and it uses PoW protocol to reach consensus. Due to the fact that the nodes in IoT are heavily resource-limited, we consider a cloud mining approach in the blockchain-enabled IoT. On the other hand, there are three reasons why agent mining is necessary: 1) IoT nodes do not need peer-to-peer communications by the blockchain all the time. Mining agents guarantee the effectiveness when IoT nodes can join in the blockchain dynamically; 2) If a user really wants to be served by a mining agent, it will compete with others and increase its mining service demand. In this way, suspicious nodes avoid accessing into the blockchain; 3) IoT nodes increase their mining service demands to compete for the privilege of mining, which also increases the mining rewards that miners get.

Thus, we consider agent mining and cloud mining approaches in the blockchain-enabled IoT. We assume there are M miners, which are denoted by $\mathcal{M} = \{1, 2, ..., m, ..., M\}$. They work as agents that receive service demands from users to mine for them. There are U IoT nodes (called users in the following) that submit their service demands to miners, indexed by $\mathcal{U} = \{1, 2, ..., u, ...U\}$. Additionally, there are N computing servers in the cloud computing system, represented by $\mathcal{N} = \{1, 2, ..., n, ..., N\}$. There are B networking bands that can be allocated to miners to propagate blocks, expressed by $\mathcal{B} = \{1, 2, ..., b, ...B\}$.

The workflow of agent mining and cloud mining can be depicted in Fig. 2, where each step means:

(1) The users want to find mining agents, and issue mining service demands to an orchestrator. The bigger mining service demand means more mining rewards that the user is intended to pay, which determines the successful probability of mining a block.

(2) According to a certain policy that will be presented in Section IV, the orchestrator makes decisions about which user can be served by a mining agent. Generally speaking, the user with bigger mining service demand is more likely to be served.

(3) In order to improve the performance of the system, cloud computing system and dynamic networking system assist miners to solve cryptographic puzzles and propagate blocks.

(4) One miner solves the cryptographic puzzle successfully, replies its user to encapsulate data in block's payloads, and produces a block.

From the above workflow, we can see that in order to enhance the performance, the user with bigger mining demand should be selected, the cloud server with more computing resources should be offloaded, and the networking band with

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/TCC.2019.2930259, IEEE Transactions on Cloud Computing

TABLE I: Comparison of the working-based consensus algorithms and the replica-based consensus algorithms.

items / Algorithm	Speed	Scalability	Finality	Typical approach	Application
The working-based consensus algorithms	Poor	Good	Poor	PoW, PoS, PoET, etc.	Permissionless blockchain (Bitcoin, Ethereum, etc.)
The replica-based consensus algorithms	Good	Moderate	Good	PBFT, RBFT, Paxos, etc.	Permissioned blockchain (Hyperledger Fabric, etc.)

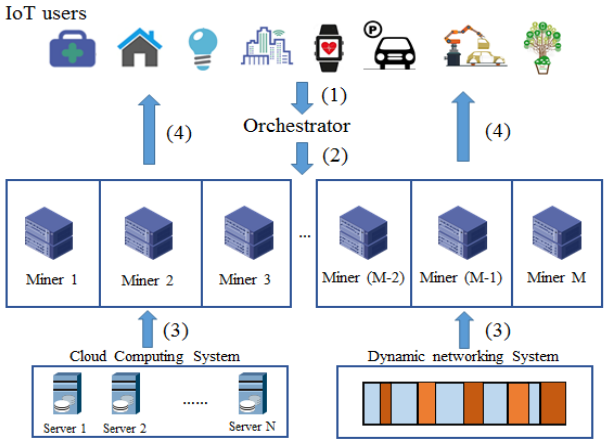

Fig. 2: The workflow of agent mining and cloud mining in the blockchain-enabled IoT.

more networking resources should be used. Accordingly, the access selection of users, computing resources allocation, and networking resources allocation should be considered. Therefore, we then present mining reward model, computing model, and networking model in the following. The notations that are used in this Section are rearranged in Table II.

B. Mining Reward Model

Each user u has different and variable service demands, which determine the successful probability of mining a block. Due to their diversity and dynamics, we can't exactly know service demand in user u submitted to the blockchain at the next time slot. Therefore, we model the service demand in user u as a random variable s_u. In order to discretize the values of s_u, we partition s_u into L discrete intervals, represented by $\mathcal{L} = \{\mathcal{L}_0, \mathcal{L}_1, ..., \mathcal{L}_{L-1}\}$. It means:

If $s'_0 \leq s_u < s'_1$, represented by $\mathcal{L}_0$;
If $s'_1 \leq s_u < s'_2$, represented by $\mathcal{L}_1$;
.....
If $s'_{L-2} \leq s_u < s'_{L-1}$, represented by $\mathcal{L}_{L-2}$;
If $s_u \leq s'_{L-1}$, represented by $\mathcal{L}_{L-1}$

Service demand s_u at time slot t can be represented by $s_u(t)$. Considering the time correlation of users' service demands, a Markov chain is used to model the transition of users' service demands. Based on a certain transition probability, $s_u(t)$ changes from one state $x_{\bar{s}}$ to another $y_{\bar{s}}$. Thus, $L \times L$ state transition probability matrix of service demand in user u can be denoted as:

$$\mathcal{S}_u(t) = [\gamma_{x_{\bar{s}} y_{\bar{s}}}(t)]_{L \times L}, \tag{1}$$

where $\gamma_{x_{\bar{s}} y_{\bar{s}}}(t) = Pr(s_u(t+1) = y_{\bar{s}} | s_u(t) = x_{\bar{s}})$, and $x_{\bar{s}}, y_{\bar{s}} \in \mathcal{L}$.

In PoW consensus protocol, miners use their computing capabilities to work on an hard puzzle. The winner of them could issue a block and reach consensus with the rest of miners. According to the work in [12], there are two factors that have effects on the successful probability of mining a block, including relative computing capacity and block propagation delay. In the mining step, the successful probability is directly relational with its relative computing capacity. In other word, a miner with more computing capacities is more likely to be the winner of this mining. In the propagation step, one miner solves the puzzle successfully, if its mined block is too large, the block will be more possible to be discarded due to long propagation delay, known as orphaning [24].

In the mining step, at time slot t, miner m has a relative computing capacity with other miners because of user u's service demand, which can be expressed as:

$$r_m^u(t) = a_m^u(t) \frac{s_u(t)}{\sum_{k=1}^{U} s_k(t)}, \tag{2}$$

where $r_m^u(t)$ is the relative computing capacity of user u's agent miner m at time slot t. $a_m^u(t)$ determines whether miner m serves user u or not. $a_m^u(t) = 1$ means user u is served by miner m; otherwise $a_m^u(t) = 0$. Note that each miner only serves one user at one time slot, thus $\sum_{u=1}^{U} a_m^u(t) = 1, \forall m, t$.

In the propagation step, we model the appearance of solving the puzzle as a Poisson process with parameter λ. Thus, the orphaning probability p_o can be approximated as [25]:

$$p_o = 1 - e^{-\lambda f(T_m)}, \tag{3}$$

where T_m is the number of transactions in a block mined by miner m. $f(T_m)$ is the block propagation delay, which has relationship with the number of transactions in a block and median bandwidth [26]. It can be expressed as follows:

$$f(T_m) = \frac{T_m}{Band}, \tag{4}$$

where $Band$ is the median bandwidth between all nodes. As we can see, the better bandwidth (networking) resources leads to the lower block propagation delay, and better

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/TCC.2019.2930259, IEEE Transactions on Cloud Computing

TABLE II: The notations used in Section III.

Notations	Description
s_u	The random variable that denotes the service demand of user u.
$s_u(t)$	The service demand of user u at time slot t.
$\mathcal{L}$	The set of discrete intervals of s_u.
$\mathcal{S}_u(t)$	The state transition probability matrix of service demand in user u at time slot t.
$\gamma_{x_{\bar{s}}y_{\bar{s}}}(t)$	The transition probability of service demand in user u.
$x_{\bar{s}}$, $y_{\bar{s}}$, $a_{\bar{s}}, b_{\bar{s}}$, $\iota_{\bar{s}}\vartheta_{\bar{s}}$	The states in $\mathcal{L}$, $\mathcal{H}$, $\mathcal{K}$.
$r_m^u(t)$	The relative computing capacity of user u's agent miner m at time slot t.
$a_m^u(t)$	The flag denoting whether miner m serves user u or not.
p_o	The orphaning probability.
λ	The parameter of a Poisson process when solving the puzzle.
T_m	The number of transactions in a block mined by miner m.
$p_m^u(t)$	The successful probability of mining a block at time slot t.
R	The fix mining reward.
φ	The variable mining reward.
$\Upsilon_m^u(t)$	The mining reward of miner m.
t_c	The computing task.
n_c	The size of task t_c.
q_c	The required number of CPU cycles to perform task t_c.
ρ_n	The random variable that denotes the computing resources in cloud server n.
$\rho_n(t)$	The computing resources in cloud server n at time slot t.
$\mathcal{H}$	The set of discrete intervals of ρ_n.
$\mathcal{P}_n(t)$	The state transition probability matrix of computing resources in cloud server n at time slot t.
$\theta_{a_{\bar{s}}b_{\bar{s}}}(t)$	The transition probability of computing resources in cloud server n.
$\eta_c(t)$	The execution time of task t_c.
$\mathcal{R}_m^n(t)$	The computing rate of miner m using the computing resources in cloud server n.
$a_m^n(t)$	The flag denoting whether cloud server n serves miner m or not.
ϱ_b	The random variable that denotes the networking resources in band b.
$\varrho_b(t)$	The networking resources in band b at time slot t.
$\mathcal{K}$	The set of discrete intervals of ϱ_b.
$\Pi_b(t)$	The state transition probability matrix of networking resources in band b at time slot t.
$\varpi_{\iota_{\bar{s}}\vartheta_{\bar{s}}}(t)$	The transition probability of networking resources in band b.

miners' utility. Thus, in the following subsection, networking model is formulated.

According to these two steps, the successful probability of mining a block $p_m^u(t)$ can be expressed as:

$$p_m^u(t) = r_m^u(t)(1-p_o) = a_m^u(t)\frac{s_u(t)}{\sum_{k=1}^{U} s_k(t)} e^{-\lambda \frac{T_m}{Band}}. \quad (5)$$

Additionally, the winner enables to obtain the reward after mining a block, which includes a fixed reward R and a variable reward φ. We assume variable reward φ follows a linear relationship with the number of transactions in the block, i.e., $\varphi = kT_m$. Thus, the mining reward of miner m can be represented as:

$$\Upsilon_m^u(t) = (R + kT_m)a_m^u(t)\frac{s_u(t)}{\sum_{k=1}^{U} s_k(t)} e^{-\lambda \frac{T_m}{Band}}. \quad (6)$$

C. Computing Model

In order to relieve the limitation of computing capacities in miners, offloading the tasks in cloud computing servers is a viable method. Let $t_c = \{n_c, q_c\}$ represent a computing task, where n_c is the size of task t_c, and q_c is the required number of CPU cycles to perform task t_c.

We assume there are multiple computation modules running. There are a number of computing servers and the other computing tasks that also utilize the computing resources in computing servers. Therefore, we can't exactly know the computing resources for the blockchain system at the next time slot. We model the computing capabilities in cloud computing server n as a random variable ρ_n. To discrete ρ_n, we partition it into H discrete intervals, indexed by $\mathcal{H} = \{\mathcal{H}_1, \mathcal{H}_2, ..., \mathcal{H}_{H-1}\}$.

In addition, we consider that computing resources ρ_n at time slot t can be expressed as $\rho_n(t)$. Similarly, considering the time correlation in computing servers, a Markov chain is used to model the transition of computing state. Based on a certain transition probability, $\rho_n(t)$ changes from one state to another. Therefore, $H \times H$ state transition probability matrix of computing state in computing server n can be denoted as:

$$\mathcal{P}_n(t) = [\theta_{a_{\bar{s}}b_{\bar{s}}}(t)]_{H\times H}, \quad (7)$$

where $\theta_{a_{\bar{s}}b_{\bar{s}}}(t) = Pr(\rho_n(t+1) = b_{\bar{s}} | \rho_n(t) = a_{\bar{s}})$, and $a_{\bar{s}}, b_{\bar{s}} \in \mathcal{H}$.

The execution time of task t_c can be expressed as:

$$\eta_c(t) = \frac{q_c}{\rho_n(t)}. \quad (8)$$

Thus, the computing rate of miner m using the computing resources in server n is denoted as:

$$\mathcal{R}_m^n(t) = a_m^n(t)\frac{n_c}{\eta_c(t)} = a_m^n(t)\frac{\rho_n(t)n_c}{q_c}, \quad (9)$$

where $a_m^n(t)$ determines whether computing server n serves miner m or not. $a_m^n(t) = 1$ means that miner m is served by computing server n; otherwise $a_m^n(t) = 0$. Note that each miner is served by one computing server at one time slot, thus $\sum_{n=1}^{N} a_m^n(t) = 1, \forall m, t$.

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/TCC.2019.2930259, IEEE Transactions on Cloud Computing

D. Networking Model

Blocks need to be delivered by enough bandwidth among miners. The concept of band renting has been proposed by Mitola [27], which helps the efficient utilization of networking resources and relieves the problem of scarcity. Thus, given that IoT has limited networking resources and the band renting in [13], [14], we consider to allocate networking resources (bandwidth resources) dynamically to the blockchain system, so as to reduce the block propagation delay, and improve the expected utility of miners.

We model networking resources used in blockchain system as a random variable ϱ_b, which can be divided into K discrete intervals, i.e. $\mathcal{K} = \{\mathcal{K}_0, \mathcal{K}_1, ..., \mathcal{K}_{K-1}\}$. And networking resources at time slot t can be denoted as $\varrho_b(t)$. Similarly, we use a Markov chain to model the transition of networking state. Based on a certain transition probability, $\varrho_b(t)$ changes from one state $\iota_{\bar{s}}$ to another $\vartheta_{\bar{s}}$, where $\iota_{\bar{s}}, \vartheta_{\bar{s}} \in \mathcal{K}$. Thus, $K \times K$ networking resources state transition probability matrix can be expressed as:

$$\Pi_b(t) = [\varpi_{\iota_{\bar{s}}\vartheta_{\bar{s}}}(t)]_{K\times K}, \tag{10}$$

where $\varpi_{\iota_{\bar{s}}\vartheta_{\bar{s}}} = Pr(\varrho_b(t+1) = \vartheta_{\bar{s}} | \varrho_b(t) = \iota_{\bar{s}})$.

In order to improve the expected utility of miners, we jointly consider the access selection of users, computing resources allocation, and networking resources allocation. Due to the fact that this joint problem is highly dynamic and dimensional, a dueling deep reinforcement learning approach can be used to solve the problem.

IV. The Learning Method

We have presented mining reward model, computing model, and networking model. In order to improve the expected utility of miners, it is necessary to jointly select the most valuable mining task, the best cloud computing server, and the best networking resources among miners, which is the joint optimization problem of users' access selection, computing allocation, and networking allocation. In this section, the joint problem is formulated as a Markov decision process, with the definition of state space, action space, and reward function. Then we present the dueling deep reinforcement learning approach to solve the problem.

A. The Learning Model

We define three tuples as $\{S, A, R\}$. S is the set of states that the learning agent senses from the environment. A denotes the possible action set of the learning agent. At time slot t, the learning agent senses state $S(t)$, and outputs action $A(t)$. This action then is executed, i.e., one user will be served by miner m, one computing server will be used by miner m, and one band will be utilized by miner m. In order to let the learning agent act better at next time, the immediate reward $R(t)$ will be fed back to the learning agent. Meanwhile, the users' service demands, the computing resources of computing servers, and the networking resources transfer to next state $S(t+1)$. The learning agent then does the same steps as above. Fig. 3 shows the learning framework of the joint problem. The goal of the learning is to find the optimal access selection strategy of nodes, computing resources allocation strategy, and networking resources allocation.

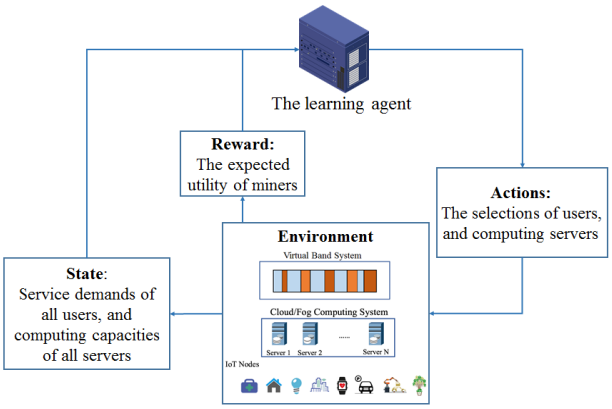

Fig. 3: The learning framework of the joint problem.

1) State Space: The learning agent interacts with the environment, including the service demands of all users, the computing capabilities of all computing servers, and the networking capabilities among all miners, which are denoted by matrix $S_U(t)$, matrix $S_N(t)$, and matrix $S_B(t)$ respectively. Matrix $S_U(t)$ can be denoted as

$$S_U(t) = [s_1(t), s_2(t), ..., s_u(t), ..., s_U(t)], \tag{11}$$

where $s_u(t)$ means the service demand of user u at time slot t. Meanwhile, matrix $S_N(t)$ can be expressed as

$$S_N(t) = [\rho_1(t), \rho_2(t), ..., \rho_n(t), ..., \rho_N(t)], \tag{12}$$

where $\rho_n(t)$ is the computing resources in cloud computing server n at time slot t. And matrix $S_B(t)$ can be represented as

$$S_B(t) = [\varrho_1(t), \varrho_2(t), ..., \varrho_b(t), ..., \varrho_B(t)], \tag{13}$$

where $\varrho_b(t)$ is the networking capabilities of band b at time slot t.

Thus, state space $S(t)$ of the joint problem can be represented as

$$S(t) = [S_U(t), S_N(t), S_B(t)]. \tag{14}$$

2) Action Space: The learning agent needs to decide which user can be served by miners (access selection of users), which computing server should be allocated to miners (computing resources allocation), and which networking resources should be allocated to miners (networking resources allocation). Thus, action space can be represented 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/TCC.2019.2930259, IEEE Transactions on Cloud Computing

$$A(t) = \begin{bmatrix} A_m^1(t) & A_m^2(t) & ... & A_m^u(t) & ... & A_m^U(t) \\ A_m^1(t) & A_m^2(t) & ... & A_m^n(t) & ... & A_m^N(t) \\ A_m^1(t) & A_m^2(t) & ... & A_m^b(t) & ... & A_m^B(t) \end{bmatrix}, \tag{15}$$

where $A_m^u(t) \in \{0,1\}$ determines which user is served by miner m. $A_m^u(t) = 1$ means that user u is served by miner m; otherwise $A_m^u(t) = 0$. Note that there is only one user served by miner m at one time slot, thus $\sum_{u=1}^{U} A_m^u(t) = 1, \forall m, t$. $A_m^n(t) \in \{0,1\}$ determines which computing server is allocated to miner m. $A_m^n(t) = 1$ represents that miner m is served by computing server n; otherwise $A_m^n(t) = 0$. Similarly, $\sum_{n=1}^{N} A_m^n(t) = 1, \forall m, t$. Additionally, $A_m^b(t) \in \{0,1\}$ denotes which band is allocated to miner m. $A_m^b(t) = 1$ means that band b is allocated to miner m; otherwise $A_m^b(t) = 0$. And $\sum_{b=1}^{B} A_m^b(t) = 1, \forall m, t$.

3) Reward Function: On one hand, miner m charges the price from user u because of being a mining agent for it. On the other hand, miner m needs to pay the computing fee and networking fee, as long as it uses cloud computing servers to mine, and band resources to propagate blocks. In order to improve the expected utility of miner m, we define the expected utility of miner m during unit time as reward function:

$$R_m(t) = \tau_u \Upsilon_m^u(t) \mathcal{R}_m^n(t) - \omega_n g_n q_c - \varepsilon_b \varrho_b(t), \tag{16}$$

where τ_u is the unit charging price for being mining agent of user u. ω_n is the unit paying price for energy consumption at server n. g_n is the energy cost of one CPU cycle in computing server n. ε_b is the unit paying price of using networking resources in band b.

In order to maximize the utility of each miner, there are some challenges remaining to be solved:

- The environment states and rewards cannot be achieved ahead of time by step by step control. Therefore, the conventional optimization methods that only consider the current state are not practicable.
- Taking a joint consideration of users' service demands, computing resources allocation, and networking resources allocation, this optimization problem is highly dimensional, highly dynamical, and highly complex. It is hard to make joint decisions by conventional methods.
- According to the work in [28], there are some regularities of users' preferences and network features. With these regularities, mining strategies and resources allocation strategies can be made ahead of time.

Therefore, in this article, we consider using the dueling deep reinforcement learning approach to address the above challenges.

B. Dueling Deep Reinforcement Learning Approach

Then we introduce the dueling deep reinforcement learning approach that finds optimal access selection strategy, computing resources allocation strategy, and networking resources allocation strategy.

In traditional reinforcement learning, the learning agent senses the environment. It denotes the feedback from each interaction by a state-action value function $Q(s,a)$, and records $Q(s,a)$ into Q-table. However, with the explosion of data complexity and dimension, it is hardly possible to obtain all $Q(s,a)$ and put them into Q-table.

The development of deep networks is a novel approach to solve the problem. Deep networks enable to be used to approximate $Q(s,a)$, that is to say, $Q(s,a,\omega,b) \approx Q(s,a)$, where ω and b are the sets of weights and biases in deep networks. This is the core methodology of deep reinforcement learning (DRL).

In order to solve the issue of instable training in DRL, two mechanisms are used in DRL:

- Experience replay: it is to store $\{S(t), A(t), R_m(t), S(t+1)\}$ into a finite-sized cyclic buffer. In each training step, the learning agent selects batches of them randomly, rather than only the current ones. This mechanism could break up the correlations that affect the training process.
- Fixed target deep networks: there are two identical deep networks in DRL, including evaluated deep networks, and fixed deep networks. The evaluated deep networks are trained in each step, and evaluate real $Q(s,a)$. The target deep networks are frozen for some time, and regarded as "real $Q(s,a)$". Additionally, the target deep networks update with the evaluated networks periodically.

In our problem formulation, the choice of actions has no relationship with the transition of states. For instance, the learning agent decides user x to be served by miner m. The next service demand of user x still transfers according to its state transition matrix, rather than whether it is served by miner m or not. Similarly, the computing resources in computing servers and the networking resources in bands also do not change with the selection of actions. Therefore, learning which action has better results is more efficient.

According to the research in [29], dueling DRL is an approach that enables to learning the relative advantage of an action. It uses $A(s,a)$ to denote the advantage of each action. There are two separate streams following the output of deep networks, one evaluating state value $V(s)$, and another evaluating action value $A(s,a)$, known as dueling architecture. Finally, these two evaluations aggregate as a one output $Q(s,a)$, which can be represented 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/TCC.2019.2930259, IEEE Transactions on Cloud Computing

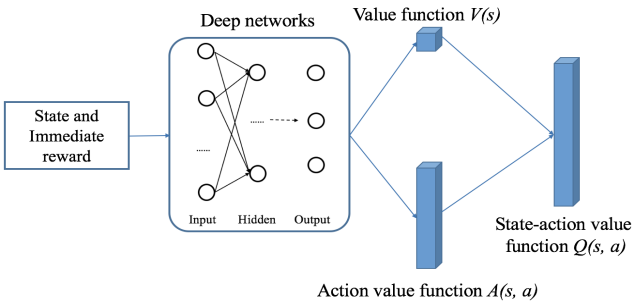

Fig. 4: The process of dueling deep Q-learning.

$$Q(s,a;\omega,b,\nu,\sigma) = V(s;\omega,b,\nu) + (A(s,a;\omega,b,\sigma) + \frac{1}{|A|}\sum_{a'} A(s,a';\omega,b,\sigma)), \quad (17)$$

where ν and σ are the learning parameters of these two separate streams. And we present the process of dueling deep reinforcement learning is Fig. 4. The pseudo-code of dueling DRL is presented in Algorithm 1.

V. Simulation Results and Discussions

In this section, computer simulation is used to evaluate the performance of our proposed scheme. To be specific, simulation settings are presented firstly, then we give the simulation results and discussions.

A. Simulation Settings

In this simulation, hardware environment is a CPU-based computer, with 8GB 1867MHz LPDDR3, 2GHz Intel Core i5, and 256G memory. Software environment is Python 3.6.8 with TensorFlow 1.4.0 [30].

We assume that there are six users that need to be served by miners, and there are five cloud computing servers that can be allocated to miners. Due to the fact that the CPU-based computer is hard to train too much data, we only use three discrete levels to represent the different service demands, the different computing capabilities, and the different networking capabilities. Also, there are 200,000 storing history samples trained in the simulation. By these history samples, deep networks are trained to decide the allocations for the following time slots.

Specially, the service demand of each user is high, medium, low, whose transition probability matrix is:

$$\mathcal{S}_u = \begin{bmatrix} 0.5 & 0.4 & 0.1 \\ 0.1 & 0.5 & 0.4 \\ 0.4 & 0.1 & 0.5 \end{bmatrix}. \quad (19)$$

Similarly, the computing capacity of each computing server is high, medium, low. We set the transition probability matrix as

$$\mathcal{P}_n = \begin{bmatrix} 0.5 & 0.3 & 0.2 \\ 0.2 & 0.5 & 0.3 \\ 0.3 & 0.2 & 0.5 \end{bmatrix}. \quad (20)$$

Additionally, the networking capability of each band is high, medium, and low, whose transition probability matrix is as follows

$$\mathcal{P}_n = \begin{bmatrix} 0.45 & 0.35 & 0.2 \\ 0.2 & 0.45 & 0.35 \\ 0.35 & 0.2 & 0.45 \end{bmatrix}. \quad (21)$$

Actually, the learning objective of deep networks is to obtain the transition probability matrices. Thus the learning agent knows what the probability of service demand, computing capability, and networking capability, transiting from one state to another. Then the learning agent makes decisions about access selection, computing allocation, and

Algorithm 1 Dueling DQN

1: Initialization:
Initialize evaluated deep networks with ω and b .
Initialize target deep networks with ω' and b'.
2: **for** $k = 1 : K$ **do**
3: Reset the environment with a randomly initial state S_{ini}, and $S(t) = S_{ini}$.
4: **while** $S(t)! = S_{terminal}$ **do**
5: Select action $A(t)$ based on $\epsilon-$greedy policy.
6: Get immediate reward $R_m(t)$ and next observation $S(t+1)$.
7: Store experience $\{S(t), A(t), R_m(t), S(t+1)\}$ into experience replay memory.
8: Randomly select some batches of $\{S(i), A(i), R_m(i), S(i+1)\}$ from experience replay memory.
9: Calculate two streams of evaluated deep networks, including $V(s;\omega,b,\nu)$ and $A(s,a;\omega,b,\sigma)$, and combine them as $Q(s,a;\omega,b,\nu,\sigma)$ using (17).
10: Calculate target Q-value $Q_{target}(s)$ in target deep networks:
if $S(i)$ is $S_{terminal}$
$Q_{target}(i) = R_m(i)$,
else
$Q_{target}(i) = R_m(i) + \gamma max_{a'} Q(s', a'; \omega', b', \nu', \sigma')$.
11: Train evaluated deep networks to minimize loss function $L(\omega, b)$

$$L(\omega,b,\nu,\sigma) = E[(Q_{target}(i) - Q(s,a;\omega,b,\nu,\sigma))^2]. \quad (18)$$

12: Every some steps, update target deep networks.
13: $S(t) \leftarrow S(t+1)$
14: **end while**
15: **end for**

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/TCC.2019.2930259, IEEE Transactions on Cloud Computing

networking allocation in advance. If deep networks can be converged in one group of specific transition probability matrices, it can be determined that they can be used and converged in other situation. Therefore, these three transition probability matrices are examples to show the performance of our proposed scheme.

The values of the rest of parameters are rearranged in Table III.

For the performance comparison, five schemes are simulated. The comparison of the five schemes is shown in Table IV.

1) DuelingDRL-based scheme. In this scheme, more valuable user can be served, as well more networking and computing resources can be used to improve the system performance. Thus, it can be anticipated that this scheme would have the best performance.

2) DuelingDRL-based scheme without networking allocation. In this scheme, more valuable user can be served, and more computing resources can be used, but only using miners' local networking resources. The advantage of allocating networking resources can be presented by this scheme.

3) DuelingDRL-based scheme without users choice. In this scheme, more networking and computing resources can be used, but selecting users randomly. The advantage of selecting the valuable user can be shown by this scheme.

4) DuelingDRL-based scheme without computing allocation. In this scheme, more valuable user can be served, and more networking resources can be used, but only using miners' local computing resources. The advantage of allocating computing resources can be indicated by this scheme.

5) Existing scheme. This simulation allows the mining agent serves users randomly without considering the mining reward, uses the local networking and computing resources.

B. Simulation Results

Fig. 5 shows the relationship between training episodes and the expected utility of miners. The learning agent runs in AdamOptimizer [31] with the learning rate of 1×10^{-4}. As we can see that with the increase of training episodes, the expected utility of miners increases, until keeps in stable values. With the joint consideration of users' service demands, computing resources in cloud computing servers, and networking resources allocation, our proposed DRL-based scheme has the better utility. The reason is that the miner enables to be a mining agent for the user with high service demand to increase the successful probability of mining a block, offload the mining task to the cloud computing server to have more computing power, and use the best networking resources to propagate blocks. Additionally, Fig. 5 shows the convergence performance of deep networks.

Fig. 6 shows the relationship between training steps and the expected utility of miners under different learning rates in duelingDRL-based scheme. As we can see, the learning rate affects the convergence performance of deep networks.

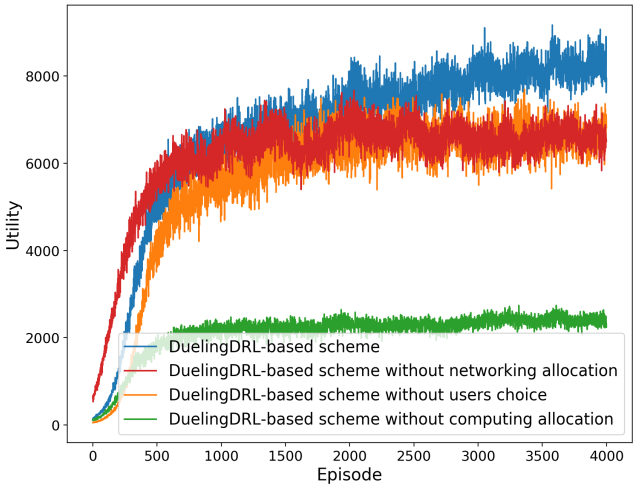

Fig. 5: Training curves tracking the expected utility of miners under different schemes.

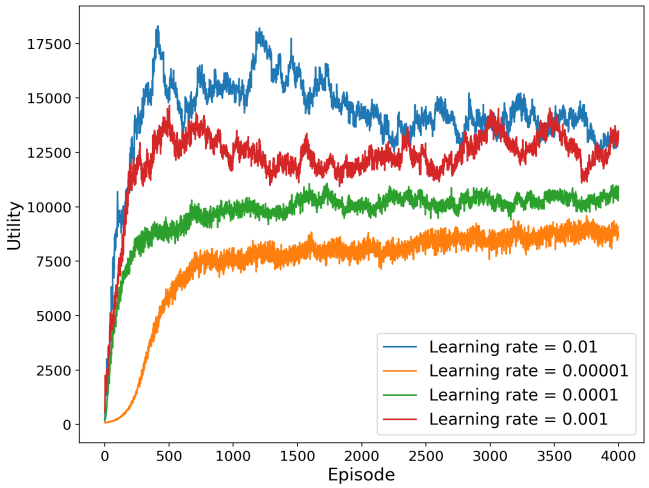

Fig. 6: Training curves tracking the expected utility of miners under different learning rates.

The bigger learning rate means the longer learning step. The longer learning step is likely to miss the global optimum, which leads to the highly scaled curves, such as when learning rates are 0.01 and 0.001. The smaller learning rate means the shorter learning step. The shorter learning step may result in the slower convergence speed, because more steps are needed to reach the global optimum, such as when learning rate is 1×10^{-5}. Therefore, considering the convergence stability and the convergence speed, we choose the learning rate as 1×10^{-4} in the simulation.

After training the deep networks, we use them in 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/TCC.2019.2930259, IEEE Transactions on Cloud Computing

TABLE III: Parameters setting in the simulation.

Parameter	Value	Description
λ	0.5 times/s	The arrival rate of solving mining puzzle.
T_m	1M	The number of transactions in a block (block size).
α	2 ms/Mb	The linear coefficient between propagation delay and block size.
R	200 units	The fixed reward of mining a block.
k	100 units	The linear coefficient between variable reward and block size.
n_c	20 Mbit	The size of mining task.
q_c	10 Mcycles	The required number of CPU cycles to perform the task.
τ_u	3 units/Mbps	The charging price of being a mining agent for each user.
g_n	1 W/Mcycle	The energy price of one CPU cycle in computing server n.
ω_n	50 units/W	The paid price for unit energy consumption of computing server n.
ε_b	3 units/Mbps	The paid price of using networking resources in band b.

TABLE IV: The comparison of the five schemes.

Item	Access selection of users	Computing resources allocation	Networking resources allocation
DuelingDRL-based scheme	✓	✓	✓
DuelingDRL-based scheme without networking allocation	✓	✓	×
DuelingDRL-based scheme without users choice	×	✓	✓
DuelingDRL-based scheme without computing allocation	✓	×	✓
Existing scheme	×	×	×

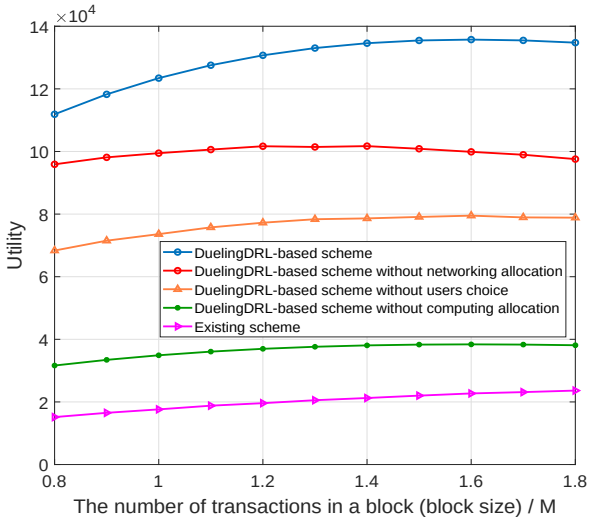

Fig. 7: The utility versus the number of transactions in a block under different schemes.

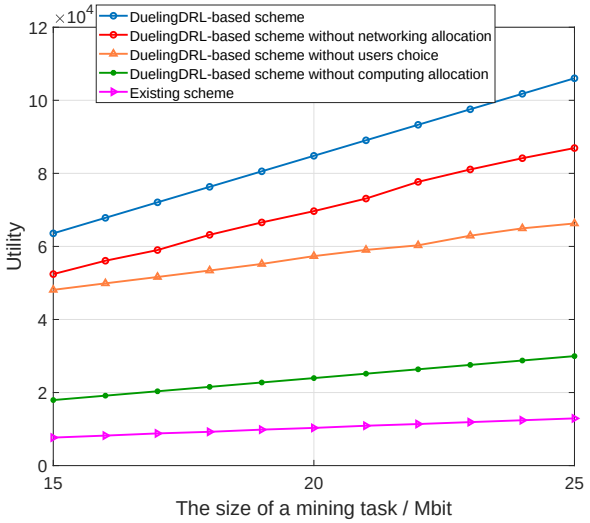

Fig. 8: The utility versus the size of a mining task under different schemes.

following simulations. Fig. 7 shows the relationship between the number of transactions in a block T_m and miners' utility. With the increase of the number of transactions, miners' utility increases. The reason is that more transactions lead to more variable reward, although they need more time to propagate, as shown in (6). However, with the joint consideration of users choice, computing resources allocation, and networking allocation, our proposed scheme has better performance.

Fig. 8 shows the relationship between the size of mining task n_c and miners' utility. With the increase of task's size, miners' utility increases. According to (9), the bigger task's size leads to the bigger computing rate, which increases miners' utility. Additionally, the performance of our proposed scheme is still the best.

Fig. 9 shows the relationship between the required number of CPU cycles to perform a task q_c and miners' utility. The increase of CPU cycles leads to the decrease of computing rate, and the increase of energy consumption, resulting in the decrease of the miners' utility. However, with the joint

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/TCC.2019.2930259, IEEE Transactions on Cloud Computing

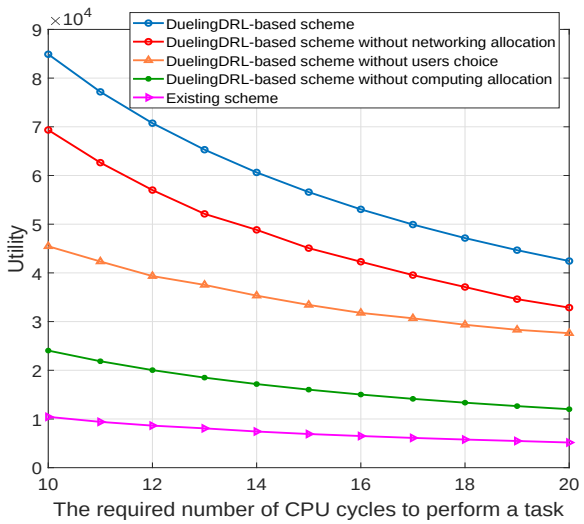

Fig. 9: The utility versus the required number of CPU cycles to perform a task under different schemes.

consideration of users choice, computing allocation, and networking allocation, our proposed scheme has the best performance.

VI. Conclusions and Future Work

In this paper, we studied a blockchain-enabled IoT. In order to solve the problem of deploying blockchain in IoT, we proposed agent mining and cloud mining approaches. We considered that the performance of the system was related to users' service demands, computing capabilities, and networking capabilities. Accordingly, we formulated the access selection of users, computing resources allocation, and networking resources allocation as a joint optimization problem. Then we used a dueling deep reinforcement learning approach to solve the problem. Simulation results showed the effectiveness of our proposed scheme. The reliability of the cloud servers has significant effects on the performance of the system. If the cloud servers cannot be trusted, they do nothing or do not compute as proposed. Thus, the more trusted cloud servers should be utilized. In addition, due to the fact that the learning results are trained from history samples, the robustness is an inevitable issue. The increases of training samples and state space can be solutions. Some future works are in progress to solve the above problems.

References

[1] S. Gartner, "Forecast: The internet of things, worldwide, 2013," 2013.
[2] W. Li, T. Logenthiran, V. Phan, and W. L. Woo, "Implemented IoT-based self-learning home management system (SHMS) for singapore," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 2212–2219, June 2018.
[3] W. Chin, W. Li, and H. Chen, "Energy big data security threats in iot-based smart grid communications," *IEEE Comm. Mag.*, vol. 55, no. 10, pp. 70–75, OCTOBER 2017.
[4] W. Ejaz, M. Naeem, A. Shahid, A. Anpalagan, and M. Jo, "Efficient energy management for the internet of things in smart cities," *IEEE Communications Magazine*, vol. 55, no. 1, pp. 84–91, January 2017.
[5] B. Chen, J. Wan, A. Celesti, D. Li, H. Abbas, and Q. Zhang, "Edge computing in IoT-based manufacturing," *IEEE Communications Magazine*, vol. 56, no. 9, pp. 103–109, SEPTEMBER 2018.
[6] T. M. Fernndez-Carams and P. Fraga-Lamas, "A review on the use of blockchain for the internet of things," *IEEE Access*, vol. 6, pp. 32 979–33 001, 2018.
[7] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.
[8] IBM, "Device democracy: Saving the future of the internet of things," 2015.
[9] C. Qiu, F. R. Yu, H. Yao, C. Jiang, F. Xu, and C. Zhao, "Blockchain-based software-defined industrial internet of things: A dueling deep q-learning approach," *IEEE Internet of Things Journal*, 2018.
[10] Z. Xiong, Y. Zhang, D. Niyato, P. Wang, and Z. Han, "When mobile blockchain meets edge computing," *IEEE Comm. Mag.*, vol. 56, no. 8, pp. 33–39, 2018.
[11] F. R. Yu, J. Liu, Y. He, P. Si, and Y. Zhang, "Virtualization for distributed ledger technology vDLT)," *IEEE Access*, vol. 6, pp. 25 019–25 028, 2018.
[12] Z. Xiong, S. Feng, W. Wang, D. Niyato, P. Wang, and Z. Han, "Cloud/fog computing resource management and pricing for blockchain networks," *IEEE Internet of Things Journal*, 2018.
[13] T. Brczes and . Horvth, "A finite-source queuing model for spectrum renting in mobile cellular networks," in *2014 ELEKTRO*, May 2014, pp. 26–30.
[14] M. Pan, H. Yue, Y. Fang, and H. Li, "The X loss: Band-mix selection for opportunistic spectrum accessing with uncertain spectrum supply from primary service providers," *IEEE Trans. on Mobile Comp.*, vol. 11, no. 12, pp. 2133–2144, Dec 2012.
[15] S. Yu, G. Wang, X. Liu, and J. Niu, "Security and privacy in the age of the smart internet of things: An overview from a networking perspective," *IEEE Comm. Mag.*, vol. 56, no. 9, pp. 14–18, 2018.
[16] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.
[17] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (IoT): A vision, architectural elements, and future directions," *Future Generation Comp. Sys.*, vol. 29, no. 7, pp. 1645–1660, 2013.
[18] L. Zhou, L. Wang, Y. Sun, and P. Lv, "Beekeeper: A blockchain-based IoT system with secure storage and homomorphic computation," *IEEE Access*, vol. 6, 2018.
[19] A. Dorri, M. Steger, S. S. Kanhere, and R. Jurdak, "Blockchain: A distributed solution to automotive security and privacy," *IEEE Comm. Maga.*, vol. 55, no. 12, pp. 119–125, 2017.
[20] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, vol. 151, pp. 1–32, 2014.
[21] C. Cachin, "Architecture of the Hyperledger blockchain fabric," in *Workshop on Distributed Cryptocurrencies and Consensus Ledgers*, 2016.
[22] J. Pan, J. Wang, A. Hester, I. Alqerm, Y. Liu, and Y. Zhao, "Edgechain: An edge-IoT framework and prototype based on blockchain and smart contracts," *IEEE Internet of Things Journal*, pp. 1–1, 2018.
[23] M. Liu, R. Yu, Y. Teng, V. C. M. Leung, and M. Song, "Computation offloading and content caching in wireless blockchain networks with mobile edge computing," *IEEE Trans. on Veh. Tech.*, pp. 1–1, 2018.
[24] N. Houy, "The bitcoin mining game," 2014.
[25] https://gist.github.com/gavinandresen/5044482, "Orphan probablity approximation."
[26] U. Klarman, S. Basu, A. Kuzmanovic, and E. G. Sirer, "bloxroute: A scalable trustless blockchain distribution network whitepaper," *IEEE Internet of Things Journal*.
[27] J. Mitola, "Cognitive radio for flexible mobile multimedia communications," in *Proc. Conf. on Mobile Multimedia Comm.*, Nov 1999, pp. 3–10.
[28] C. Qiu, H. Yao, R. Yu, F. Xu, and C. Zhao, "Deep q-learning aided networking, caching, and computing resources allocation in software-defined satellite-terrestrial networks," *IEEE Trans. on Veh. Tech.*, pp. 1–1, 2019.

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/TCC.2019.2930259, IEEE Transactions on Cloud Computing

[29] Z. Wang, T. Schaul, M. Hessel, H. Van Hasselt, M. Lanctot, and N. De Freitas, "Dueling network architectures for deep reinforcement learning," *arXiv preprint arXiv:1511.06581*, 2015.
[30] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.
[31] T. M. Chilimbi, Y. Suzue, J. Apacible, and K. Kalyanaraman, "Project Adam: Building an efficient and scalable deep learning training system." in *OSDI*, vol. 14, 2014, pp. 571–582.

Chao Qiu received the B.S. degree from China Agricultural University, Beijing, China in 2013 in communication engineering. She is currently pursuing the Ph.D. degree with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications. From September 2017 to September 2018, she visited Carleton University, Ottawa, ON, Canada, as a Visiting Scholar. Her current research interests include machine learning, software defined networking, and blockchain.

Haipeng Yao received the MSc and PhD degrees in Circuits and Systems from Beijing University of Posts and Telecommunications, in 2008 and 2011, respectively. He is currently an associated professor with the School of Information and Communication Engineering, and the State Key Laboratory of Networking and Switching Technology in Beijing University of Posts and Telecommunications. His main research interests include network artificial intelligence,future network architecture, big data for networking, the architecture and key technology of the new generation mobile communication system.

Chunjiao Jiang (S'09-M'13-SM'15) received the B.S. degree (Hons.) in information engineering from Beihang University in 2008, and the Ph.D. degree (Hons.) in electronic engineering from Tsinghua University in 2013. From 2013 to 2016, he holds a post-doctoral position with the Department of Electronic Engineering, Tsinghua University, during which he visited the University of Maryland at College Park and the University of Southampton. He is currently a Research-Track Faculty Member and an Assistant Research Fellow with the Tsinghua Space Center, Tsinghua University. He was a recipient of the IEEE Globecom Best Paper Award in 2013, the IEEE GlobalSIP Best Student Paper Award in 2015, and the IEEE Communications Society Young Author Best Paper Award in 2017.

Song Guo (M'02-SM'11) is a Full Professor at Department of Computing, The Hong Kong Polytechnic University. He received his Ph.D. in computer science from University of Ottawa and was a professor with the University of Aizu. His research interests are mainly in the areas of big data, cloud computing and networking, and distributed systems with over 400 papers published in major conferences and journals. His work was recognized by the 2016 Annual Best of Computing: Notable Books and Articles in Computing in ACM Computing Reviews. He is the recipient of the 2017 IEEE Systems Journal Annual Best Paper Award and other five Best Paper Awards from IEEE/ACM conferences. Prof. Guo was an Associate Editor of IEEE Transactions on Parallel and Distributed Systems and an IEEE ComSoc Distinguished Lecturer. He is now on the editorial board of IEEE Transactions on Emerging Topics in Computing, IEEE Transactions on Sustainable Computing, IEEE Transactions on Green Communications and Networking, and IEEE Communications. Prof. Guo also served as General, TPC and Symposium Chair for numerous IEEE conferences. He currently serves as a Director and Member of the Board of Governors of ComSoc.

Fangmin Xu received the M.S. and Ph.D. degrees in Communication Engineering from Beijing University of Posts and Telecommunication(BUPT), China, in 2003 and 2008, respectively. He is currently associate professor in the School of Information and Communicaiton Engineering, BUPT, China. From 2008 to 2014, he was with Samsung Electronics where he actively contributed to 3GPP LTE/LTE-A and IEEE 802.16m. He is the author of 2 books, 20 peer-reviewed international research papers, 50 standard contributions and the inventor of 15 issued or pending patents among which 4 have been adopted in the specifications of 4G (3GPP LTE/LTE-A and IEEE 802.16m) standards. His research interests include advance technologies in wireless networks, especially Internet of things(IoT) field.