IEICE TRANS. COMMUN., VOL.E107–B, NO.12 DECEMBER 2024

PAPER *Special Section on Emerging Communication Technologies in Conjunction with Main Topics of ICETC 2023*

Repeated Stochastic Game and Lyapunov Optimization for Mining Task Offloading in Decentralized Applications*

Kota YAMADA[†a)], **Takanori HARA**[†b)], ***Members***, ***and*** **Shoji KASAHARA**[†c)], ***Fellow***

SUMMARY Mining task offloading has been attracting users of decentralized applications (DApps) because they have only devices with the limited resource, which make it difficult to execute resource-intensive mining tasks. This allows the DApp users to offload the mining tasks to a cloud and/or a mobile edge computing server managed by a cloud service provider (CSP). To ensure the sustainable cloud/edge services and the integrity of blockchain, a CSP selection problem arises, which is a problem to assign the offloading requests to an appropriate CSP. In this paper, we propose a mining task offloading strategy for the CSP selection problem to maximize the miner's expected utility. More specifically, we formulate the CSP selection problem as a repeated stochastic game such that the coarse correlated equilibrium is achieved among miners (DApp users). In addition, we develop an online algorithm for efficiently solving the repeated stochastic game using the Lyapunov optimization and the drift-plus-penalty algorithm. Through the numerical experiments, we demonstrate the characteristics of the proposed strategy in terms of parameter sensitivity, utility, fairness, and execution time.

key words: *decentralized application (DApp), mining task offloading, coarse correlated equilibrium (CCE), repeated stochastic game, Lyapunov optimization, drift-plus-penalty algorithm*

1. Introduction

A *decentralized application* (DApp) is an application through smart contracts autonomously operating on a permissionless blockchain technology [2]. With increasing the demand for DApps, mining, which is one of consensus-building processes in a blockchain network, grows in importance from the viewpoint of the secure operation of DApps. The *proof-of-work* (PoW) based blockchain network motivates DApp users to become a consensus node (i.e., a miner) because they have an opportunity to acquire the mining reward. However, the DApp users become reluctant to solve a PoW puzzle because most of them have only devices with the limited resource such as mobile devices and internet of things (IoT) devices. In other words, their device makes it difficult to execute the resource-intensive mining task. To compensate for the limited resource, mining task offloading has been attracting such DApp users [3], [4]. This allows the DApp user to offload the computational task required for mining to a cloud and/or a *mobile edge computing* (MEC) server, managed by a *cloud service provider* (CSP).

There have been many studies for mining task offloading [3], [4]. Toward realizing sustainable mining task offloading, these studies have tackled a resource allocation problem by taking into account the interaction between miners and CSPs. If a lot of offloading requests by DApp users are concentrated on a particular CSP, the services provided by the CSP will become congested, resulting in high operational costs. On the other hand, if there exist only few requests by DApp users, the integrity of blockchain network may be compromised. Therefore, a *CSP selection problem*, which is a problem to assign the offloading request to the appropriate CSP, arises from the viewpoint of the service stability and the integrity of blockchain network. The offloading strategies for the CSP selection can mainly be categorized into three types of strategies: (1) the maximization of social welfare [5]–[7], (2) the maximization of CSP's utility [8]–[10], and (3) the maximization of miner's utility [11]–[14].

In this paper, we formulate the CSP selection problem as a *repeated stochastic game* [15] in order to maximize miner's utility, inspired by [16], [17]. The repeated stochastic game aims at realizing the CSP selection such that a *coarse correlated equilibrium* (CCE) [18] is achieved among the DApp users. Since the repeated stochastic game cannot be directly solved in terms of computational complexity, we propose an online algorithm to solve the repeated stochastic game, with the assistance of *Lyapunov optimization* and *drift-plus-penalty algorithm*. Through the numerical experiments, we demonstrate the characteristics of the proposed algorithm in terms of parameter sensitivity, utility, fairness, and execution time.

The rest of the paper is organized as follows. Section 2 gives the related work. Section 3 provides the system model assumed in this paper. In Sect. 4, we propose a repeated stochastic game and Lyapunov optimization for mining task offloading in decentralized applications. Section 5 shows the fundamental characteristics of the proposal. Finally, Sect. 6 gives the conclusion and future work.

2. Related Work

Mining task offloading related surveys can be found in [3], [4]. Table 1 presents the summary of representative related work on mining task offloading. The social welfare

Manuscript received February 2, 2024.
Manuscript revised May 28, 2024.
Manuscript publicized August 22, 2024.

[†]Division of Information Science, Nara Institute of Science and Technology, Ikoma-shi, 630-0192 Japan.

*This paper is an expanded version of the paper that was presented at 2023 International Conference on Emerging Technologies for Communications (ICETC2023) [1].

a) E-mail: yamada.kota.yi0@is.naist.jp
b) E-mail: hara@ieee.org
c) E-mail: kasahara@ieee.org

DOI: 10.23919/transcom.2024CEP0001

Table 1 Summary of representative related work on mining task offloading.

Strategy	Ref.	Objective	Algorithm/Model	# of CSPs/servers	Time transition
Maximization of social welfare	[5]	Maximizing the miners' and CSP's profit	Single-sided auction model	Single CSP	-
	[6]	Maximizing the miners' profit and the MEC servers' profit	Stackelberg game	Multiple MEC servers	-
	[7]	Maximizing the miners' profit and the CSPs' profit	Double-sided auction model	Multiple CSPs	-
Maximization of CSP's utility	[8]	Minimizing the power consumption and task response time	Lyapunov optimization	Single MEC server and single MCC server	✓
	[9]	Maximizing the fog servers' profit	Matching theory	Multiple fog servers	-
	[10]	Maximizing the CSP's profit	Single-sided auction and deep learning	Single CSP	-
Maximization of miner's (user's) utility	[11]	Maximizing the miner's profit by pool selection	Offloading a mining task to a nearby MEC server or a cloud	Single CSP or multiple MEC servers	✓
	[12]	Maximizing the miner's profit in a PoC consensus mechanism	Non-cooperative game	Single CSP	-
	[13]	Maximizing the miner's profit	ADMM algorithm	Multiple CSPs	-
	[14]	Minimizing the power consumption and the latency for local execution and task offloading	TOPSIS and RL	Multiple MEC servers	✓
	This work	Maximizing the miner's profit	Repeated stochastic game	Multiple CSPs	✓

maximization strategy aims at maximizing the total utility of both miners and CSPs combined [5]–[7]. Jiao et al. developed the auction mechanism, consisting of two-round valuation, in resource allocation and pricing [5]. In the first round, since miners could not know the total amount of resources and the number of winners, they bid according to the expected reward. In the next round, miners derive the post-valuation consisting of the expected reward and the network effect by considering the auction result. Zhang et al. formulated a CSP selection problem as a Stackelberg game to select the appropriate offloading strategy by taking into account the different preference toward the tradeoff between risk and reward [6]. Focusing on the interaction between miners and CSPs, Liu et al. formulated a double-sided auction game to determine the offloading strategy [7].

The CSP's utility maximization strategy is a strategy for energy and cost saving and its profit maximization [8]–[10]. In order to efficiently offload resource-intensive tasks to MEC and mobile cloud computing (MCC) servers, Wu et al. proposed the Lyapunov optimization to determine an appropriate offloading server such that the energy consumption and task response time are minimized [8]. Yang et al. proposed the distributed matching algorithm that allocates mining tasks to fog servers in order to maximize the fog servers' profit [9]. Luong et al. developed the single-sided auction between CSPs and miners using deep learning to maximize the CSPs' profit [10]. Different from these studies, we explore an offloading strategy that aims to maximize the miner's utility.

The miner's utility maximization strategy aims at maximizing the miner's profit indicating the difference between the mining reward obtained by solving a PoW puzzle and the payment cost to the CSP [11]–[14]. In [11], Mai et al. proposed a centralized mining pool selection algorithm using an evolutionary game to maximize the profit of each miner. In this work, they, however, assumed that each miner offloads a mining task to a nearby MEC server or a cloud. In addition, they proposed a distributed mining pool selection algorithm using a win or learn fast policy hill climbing (WoLF-PHC) algorithm [19], which is one of the distributed reinforcement learning (RL) algorithms, by considering the non-cooperative relationship among miners. In order to maximize the miner's profit, Jiang et al. formulated a problem of determining the ratio of hash power to allocate to self mining and cloud mining as the non-cooperative game in a proof-of-capacity (PoC) consensus mechanism [12]. Liu et al. proposed an offloading and caching strategy using an alternating direction method of multiplier (ADMM) algorithm [20] in the wireless blockchain network using MEC servers [13]. This algorithm selects where to offload mining tasks and caches both requested content and computation result to handle ever-increasing network traffic. In [14], the authors proposed a method for offloading tasks of IoT devices to MEC servers to minimize their power consumption and latency and adopts a double deep q-network (DDQN) [21] and a technique for order of preference by similarity to ideal solution (TOPSIS) algorithm [22] to optimize the multi-objective decision making. Note that this method is not directly related to mining task offloading.

The existing studies [5], [10], [12] assume only the ex-

istence of single CSP but the CSP may set excessive service costs in this situation. On the contrary, the studies [11], [14] to aim at the maximization of miner's (user's) utility take the existence of multiple CSPs into account but not mining task offloading. In [12], [13], the interaction between miners and CSPs over time is not considered. In other words, these studies do not sufficiently consider both the existence of multiple CSPs and the interaction between miners and CSPs over discrete time from the aspect of mining task offloading. To tackle these issues, we aim to formulate a CSP selection problem as a repeated stochastic game to maximize the miner's profit and develop an online algorithm using the Lyapunov optimization for mining task offloading by taking into account the existence of multiple CSPs and the interaction between miners and CSPs over discrete time, inspired by [16], [17]. In [16], the author formulated a repeated stochastic game that attains the CCE among users and developed an online algorithm for efficiently solving this game. Li et al. applied a repeated stochastic game to a network selection problem in a wireless network [17]. In this paper, we also apply a repeated stochastic game to a CSP selection problem for mining task offloading.

In the conference version [1], we cannot provide detailed explanation and numerical results of the proposed method due to the page limitation. In this paper, we elaborate on a system model, formulation of repeated stochastic game, and an online algorithm in Sects. 3 and 4. In Sect. 5, we provide the evaluation results from the viewpoint of utility, queue stability, and execution time by reconsidering the evaluation settings by using the actual data obtained from [23]–[25].

3. System Model

In this section, we elaborate on the system model. Table 2 presents the notations used in this paper.

3.1 Repeated Stochastic Game

A repeated stochastic game models interactions in which the environment changes according to the N users' actions over an infinite time slot $t \in \{0, 1, 2, \ldots\}$ [15]. Let $\mathcal{N} = \{1, \ldots, N\}$ be a set of users (i.e., miners). Note that terms user and miner will be used interchangeably. Each user $i \in \mathcal{N}$ chooses an action $\alpha_i(t) \in \mathcal{A}_i$ under the whole history states including the current state on time slot t. Given the action combinations among all users and the current state, the probability distribution of action vector $\boldsymbol{\alpha}(t) = (\alpha_1(t), \ldots, \alpha_N(t))$ and the utility $u_i(\boldsymbol{\alpha}(t))$ of each user on time slot t are determined. Note that symbols $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}(t)$ will be used interchangeably.

The definition of Nash equilibrium assumes that every user $i \in \mathcal{N}$ independently chooses an action. The probability mass function $\Pr[\boldsymbol{\alpha}]$ is a *(mixed) Nash equilibrium* (MNE) if it satisfies:

$$\sum_{\boldsymbol{\alpha} \in \mathcal{A}} \Pr[\boldsymbol{\alpha}] \hat{u}_i(\boldsymbol{\alpha}) \geq \sum_{\boldsymbol{\alpha} \in \mathcal{A}} \Pr[\boldsymbol{\alpha}] \hat{u}_i((\beta_i, \boldsymbol{\alpha}_{-i})),$$

Table 2 Notations.

Symbol	Description
$\mathcal{N}$	Set of users (miners)
$\mathcal{M}$	Set of CSPs
$\mathcal{A}_i$	Set of miner i's actions
$\mathcal{A}$	Cartesian product set of miners' actions
α_i	Miner i's action
$\boldsymbol{\alpha}$	Action vector
Ω_i	Set of events observed by miner i
Ω	Cartesian product set of events
ω_0	Event vector observed by a game manager
ω_i	Event vector observed by miner i
ω	Event matrix, $\omega = [\omega_1, \ldots, \omega_N]$
d_i	Computing demand required by miner i
c^j	Congestion state of CSP j
C^j	Computing resource of CSP j
p^j	Unit service price of CSP j
R	Mining reward
r	Transaction fee
$\Pr[\cdot]$	Probability mass function
$u_i(t)$	Utility function
$\mathcal{B}_i$	Pure strategies for miner i
$b_i^{(\beta)}(\cdot)$	Pure strategy function for miner i
$\phi(\cdot)$	α-proportional fairness function
$Q_i^{(\beta)}, Z_i$	Virtual queues
$\boldsymbol{\gamma}$	Auxiliary vector
$u_i^{\max}$	Maximum utility of user i
$\overline{u}_i$	Time average utility of user i
V	Controllable parameter of a drift-plus-penalty algorithm

$$\forall i \in \mathcal{N}, \forall \beta_i \in \mathcal{A}_i, \tag{1}$$

$$\Pr[\boldsymbol{\alpha}] = \prod_{k \in \mathcal{N}} \Pr[\alpha_k(t) = \alpha_k], \tag{2}$$

where $\boldsymbol{\alpha}_{-i} = (\alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_N)$ $(\boldsymbol{\alpha}_{-i} \in \mathcal{A}_{-i})$ stands for an action vector except user i. Note that $\mathcal{A} = \mathcal{A}_1 \times \ldots \times \mathcal{A}_N$ and $\Omega = \Omega_1 \times \ldots \times \Omega_N$. Equation (1) indicates that every user $i \in \mathcal{N}$ cannot improve his/her utility when deviating from the action α_i. Equation (2) represents that every user $i \in \mathcal{N}$ independently takes an action. It is well-known that there exists at least one MNE in every game but the high computational complexity is imposed to derive the MNE.

Different from an MNE, the definition of CCE assumes that a game manager who provides the suggestion S_i to each user $i \in \mathcal{N}$ exists and non-participating users do not receive any suggestions from the game manager [18]. In other words, the CCE is an equilibrium where every user $i \in \mathcal{N}$ cannot improve his/her utility when deviating from the suggestion S_i, assuming that all users follow the suggestions from the game manager. Therefore, the probability mass function $\Pr[\boldsymbol{\alpha}]$ is a CCE if it satisfies Eq. (1). Let $\mathcal{E}_{\mathrm{MNE}}$ and $\mathcal{E}_{\mathrm{CCE}}$ be the solution spaces of MNE and CCE, respectively. Since $\mathcal{E}_{\mathrm{MNE}} \subseteq \mathcal{E}_{\mathrm{CCE}}$ is established, the CCE may reduce the solution optimality compared with the MNE but can derive solutions faster [16].

3.2 CSP Selection Problem

In this section, we formulate the CSP selection problem as the repeated stochastic game. We assume that each user $i \in \mathcal{N}$

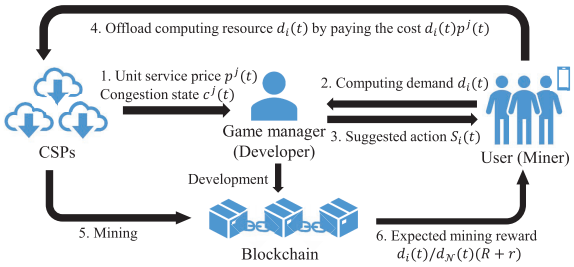


Fig. 1 System model.

first observes a random event (i.e., a current state) $\omega_i \in \Omega_i$ and then chooses a CSP $j \in \mathcal{M}$ from a set $\mathcal{M} = \{1, \ldots, M\}$ of M CSPs for the mining task offloading at each time slot t. Let Ω_i be a set of states that can be observed by the user i. The current states observed by all users on time slot t is defined as a matrix $\boldsymbol{\omega}(t) = [\omega_1(t), \ldots, \omega_N(t)]$. We assume that $\boldsymbol{\omega}$ is independent and identically distributed (i.i.d.) over time slots. Let $\boldsymbol{\omega}_0(t)$ be the current state observed by the game manager. In what follows, we assume that each user $i \in \mathcal{N}$ receives the current state $\boldsymbol{\omega}_0(t)$ from the game manager (i.e., $\boldsymbol{\omega}_i(t) = \boldsymbol{\omega}_0(t)$).

Figure 1 illustrates the system model assumed in this paper. For each time slot t, each CSP $j \in \mathcal{M}$ notifies its congested state $c^j(t)$ and the unit service price $p^j(t)$ to the game manager (i.e., a DApp developer) (Step 1 in Fig. 1). Each user $i \in \mathcal{N}$ receives the congested states $\boldsymbol{\omega}_i(t) = (c^1(t), \ldots, c^M(t))$ of M CSPs from the game manager and subsequently notifies the computing demand (i.e., hash power) $d_i(t)$ required for the mining task to the game manager (Step 2 in Fig. 1). Here, $c^j(t) \in \{0, 1\}$. $c^j(t) = 0$ (resp. $c^j(t) = 1$) means that the computing resource of CSP j is available (resp. congested). The congestion probability $q^j(t)$ of CSP j is given by

$$q^j(t) = \begin{cases} \frac{d^j(t)}{C^j(t)}, & \text{if } d^j(t) \le C^j(t), \\ 1, & \text{otherwise,} \end{cases}$$

where $d^j(t) = \sum_{i \in \{k \in \mathcal{N} | \alpha_k(t) = j\}} d_i(t)$. $d^j(t)$ denotes the total sum of the computing demands required to the CSP j on time slot t. C^j denotes the available computing capacity of CSP j. On time slot t, the CSP j is available (resp. congested) with the probability $1 - q^j(t)$ (resp. $q^j(t)$).

The game manager derives the suggestion $S_i(t) \in \mathcal{A}_i$ of user $i \in \mathcal{N}$ such that the CCE is achieved among the users and then notifies it to the user i (Step 3 in Fig. 1). Here, $\mathcal{A}_i (= \{0\} \cup \mathcal{M})$ denotes a set of actions (a set of available CSPs) for user i and 0 means no selection. Each user $i \in \mathcal{N}$ takes an action $\alpha_i(t) = S_i(t)$ according to the suggestion from the game manager (Step 4 in Fig. 1) and offloads the mining task to the CSP $j = \alpha_i(t)$ (Step 5 in Fig. 1). If the user succeeds in mining, he/she can obtain the mining reward R and the transaction fee r (Step 6 in Fig. 1). We assume that each user can choose a single CSP.

4. Proposed Method

4.1 Formulation of CSP Selection Problem

In this section, we formulate the CSP selection problem as an optimization problem. On time slot t, the utility $\hat{u}(\boldsymbol{\alpha}(t), \boldsymbol{\omega}(t))$ of user i who chooses the CSP j is defined as follows:

$$\hat{u}_i(\boldsymbol{\alpha}(t), \boldsymbol{\omega}(t)) = \begin{cases} \frac{d_i(t)}{d_{\mathcal{N}}(t)}(R + r) - d_i(t)p^j(t), \\ \qquad \text{with probability } 1 - q^j(t), \\ 0, \quad \text{with probability } q^j(t), \end{cases}$$

where $d_{\mathcal{N}}(t) = \sum_{j \in \mathcal{N}} d_j(t)$ denotes the total sum of the computing demands of all users. $d_i(t)/d_{\mathcal{N}}(t)$ means the probability of successfully mining a block, which is represented by the ratio of the total computing demand to the computing demand of user i. $d_i(t)p^j(t)$ means the payment cost to CSP j.

For each time slot t, since the utility $u_i(t)$ of user i depends on both actions $\boldsymbol{\alpha}(t)$ and congested states $\boldsymbol{\omega}(t)$, $u_i(t)$ is defined as a real-value function of $\boldsymbol{\alpha}(t)$ and $\boldsymbol{\omega}(t)$ (i.e., $u_i(t) = \hat{u}_i(\boldsymbol{\alpha}(t), \boldsymbol{\omega}(t))$):

$$u_i(t) = \begin{cases} \frac{d_i(t)}{d_{\mathcal{N}}(t)}(R + r) - d_i(t)p^j(t), \\ \qquad \text{if } \alpha_i(t) = j,\ j \neq 0,\ c^j(t) = 0, \\ 0, \quad \text{if } \alpha_i(t) = j,\ j \neq 0,\ c^j(t) = 1, \\ 0, \quad \text{if } \alpha_i(t) = j,\ j = 0. \end{cases}$$

The conditional probability mass function $\Pr[\boldsymbol{\alpha} \mid \boldsymbol{\omega}]$ is defined as follows:

$$\Pr[\boldsymbol{\alpha} \mid \boldsymbol{\omega}] \ge 0, \quad \forall \boldsymbol{\alpha} \in \mathcal{A}, \forall \boldsymbol{\omega} \in \Omega, \tag{3}$$

$$\sum_{\boldsymbol{\alpha} \in \mathcal{A}} \Pr[\boldsymbol{\alpha} \mid \boldsymbol{\omega}] = 1, \quad \forall \boldsymbol{\omega} \in \Omega. \tag{4}$$

We assume that the action vector $\boldsymbol{\alpha}$ is selected independently over time slots according to the i.i.d. $\boldsymbol{\omega}$ and the same conditional probability mass function $\Pr[\boldsymbol{\alpha} \mid \boldsymbol{\omega}]$. Thanks to the large sample approximation, the time average utility $\overline{u}_i$ of user i is given by

$$\overline{u}_i = \sum_{\boldsymbol{\omega} \in \Omega} \sum_{\boldsymbol{\alpha} \in \mathcal{A}} \pi[\boldsymbol{\omega}] \Pr[\boldsymbol{\alpha} \mid \boldsymbol{\omega}] \hat{u}_i(\boldsymbol{\alpha}, \boldsymbol{\omega}), \quad \forall i \in \mathcal{N}, \tag{5}$$

where $\pi[\boldsymbol{\omega}] = \Pr[\boldsymbol{\omega}(t) = \boldsymbol{\omega}]$.

Each user aims at maximizing his/her time average utility over time slots while the game manager aims at providing fair opportunities for mining task offloading among users. Considering these points, we introduce the α-proportional fairness function $\phi(\cdot)$ ($\alpha = 1$).

$$\phi(\overline{u}_1, \ldots, \overline{u}_N) = \sum_{i \in \mathcal{N}} \log \overline{u}_i. \tag{6}$$

The CSP selection problem can be formulated as the following optimization problem P_1.

$$\max_{\Pr[\boldsymbol{\alpha} | \boldsymbol{\omega}]} \quad (6),$$

$$\text{s.t.} \quad (3), (4), (5),$$
$$\sum_{\omega \in \Omega} \sum_{\alpha \in \mathcal{A}} \pi[\omega] \Pr[\alpha \mid \omega] \hat{u}_i(\alpha, \omega) \geq \sum_{\omega \in \Omega} \sum_{\alpha \in \mathcal{A}} \pi[\omega] \Pr[\alpha \mid \omega] \hat{u}_i((b_i^{(\beta)}, \alpha_{-i}), \omega), \quad \forall i \in \mathcal{N}, \forall \beta \in \mathcal{B}_i, \tag{7}$$

Constraint (7) represents the CCE constraint. $\mathcal{B}_i$ denotes the pure strategies for user i and $b_i^{(\beta)} = b_i^{(\beta)}(\omega_i)$ represents the pure strategy function for user i.

4.2 Online CSP Selection Algorithm

It is difficult to directly solve the optimization problem P_1 in terms of computational complexity because (1) P_1 includes the unknown conditional probability mass function $\Pr[\boldsymbol{\alpha} \mid \boldsymbol{\omega}]$ and (2) the objective function is a non-linear function of time average utility among users. In this section, we propose an online algorithm to solve P_1 with the help of Lyapunov optimization, inspired by [16].

4.2.1 Reformulation of CSP Selection Problem

Since Eq. (5) includes the unknown $\Pr[\boldsymbol{\alpha} \mid \boldsymbol{\omega}]$, we redefine the time average utility $\overline{u}_i$ of user i until time slot t as follows:

$$\overline{u}_i(t) = \frac{1}{t} \sum_{\tau=0}^{t-1} \mathrm{E}[u_i(\tau)]. \tag{8}$$

If only the user i switches the action from $\alpha_i(t)$ to the available action $b_i^{(\beta)}(t)$, the utility $u_i^{(\beta)}(t)$ is defined as $\hat{u}_i((b_i^{(\beta)}(\omega_i), \boldsymbol{\alpha}_{-i}), \boldsymbol{\omega})$.

A stationary and randomized algorithm observes the congested state $\boldsymbol{\omega}(t)$ and independently selects suggested actions $\boldsymbol{S}(t) = \boldsymbol{\alpha}(t)$ according to the same conditional probability mass function $\Pr[\boldsymbol{\alpha} \mid \boldsymbol{\omega}]$. If P_1 can be solved by some conditional probability mass function $\Pr[\boldsymbol{\alpha} \mid \boldsymbol{\omega}]$, the following optimization problem P_2 can also be solved. In addition, any solution of P_2 has time average expectations that are arbitrarily close to conditional probability mass functions $\Pr[\boldsymbol{\alpha} \mid \boldsymbol{\omega}]$ that solve P_1 [16]. Therefore, the optimization problem P_1 can be transformed into the following optimization problem P_2.

$$\max_{\Pr[\boldsymbol{\alpha}|\boldsymbol{\omega}]} \quad \liminf_{t \to \infty} \phi(\overline{u}_1(t), \ldots, \overline{u}_N(t)), \tag{9}$$
$$\text{s.t.} \quad \liminf_{t \to \infty} [\overline{u}_i(t) - \overline{u}_i^{(\beta)}(t)] \geq 0, \quad \forall i \in \mathcal{N}, \forall \beta \in \mathcal{B}_i, \tag{10}$$
$$\boldsymbol{\alpha}(t) \in \mathcal{A}, \forall t \in \{0, 1, \ldots\}. \tag{11}$$

The objective function (9) aims to maximize the inferior limit of the proportional fairness function of time average utility. Constraint (10) represents the CCE constraint satisfying that the time average utility based on the suggestion is greater than or equal to that based on deviating behavior.

4.2.2 Jensen's Inequality

It is difficult to directly maximize Eq. (9) because Eq. (9) is represented as a non-linear function of time average utility among users. With the assistance of the Jensen's inequality and the auxiliary variable technique [26], the maximization of the non-linear function of time average can be transformed into the maximization of the time average of non-linear function. Let $\boldsymbol{\gamma}(t) = (\gamma_1(t), \ldots, \gamma_N(t))$ be an auxiliary vector tuned by the game manager on time slot t. Note that $0 \leq \gamma_i(t) \leq u_i^{\max}$ and $u_i^{\max}$ is maximum utility of user i. Applying the auxiliary vector to Eq. (6), we can define $g(t) = \phi(\overline{\gamma}_1(t), \ldots, \overline{\gamma}_N(t))$. From the Jensen's inequality, we have

$$\overline{g}(t) \leq \phi(\overline{\gamma}_1(t), \ldots, \overline{\gamma}_N(t)). \tag{12}$$

As a result, the optimization problem P_2 can be transformed into the optimization problem P_3 that the game manager observes $\boldsymbol{\omega}(t)$ and chooses $\boldsymbol{\alpha}(t)$ and $\boldsymbol{\gamma}(t)$ over time slots. The optimization problem P_3 is defined as follows:

$$\max \quad \liminf_{t \to \infty} \overline{g}(t),$$
$$\text{s.t.} \quad (10), (11),$$
$$\lim_{t \to \infty} |\overline{\gamma}_i(t) - \overline{u}_i(t)| = 0, \forall i \in \mathcal{N}, \tag{13}$$
$$0 \leq \gamma_i(t) \leq u_i^{\max}, \forall i \in \mathcal{N}, \forall t \in \{0, 1, \ldots\}. \tag{14}$$

Constraint (14) represents the domain of auxiliary variables. Suppose that all limits exist in P_3 so that the constraint (13) guarantees $\overline{\gamma}_i(t) = \overline{u}_i(t)$. The objective value of P_3 is less than or equal to that of P_2 according to Eq. (12).

4.2.3 Lyapunov Optimization

The optimization problem P_3 can be solved by using a drift-plus-penalty algorithm [26]. In order to meet the constraints (10) and (13), we define virtual queues $Q_i^{(\beta)}(t)$ and $Z_i(t)$ with the following update rules.

$$Q_i^{(\beta)}(t+1) = \max[Q_i^{(\beta)}(t) + u_i^{(\beta)}(t) - u_i(t), 0], \tag{15}$$
$$Z_i(t+1) = Z_i(t) + \gamma_i(t) - u_i(t). \tag{16}$$

Equation (15) (resp. Eq. (16)) represents a queueing process with arrival rate $u_i^{(\beta)}(t)$ and service rate $u_i(t)$ (resp. arrival rate $\gamma_i(t)$ and service rate $u_i(t)$). Therefore, the constraints (10) and (13) are satisfied if these queues are *mean rate stable* as follows:

$$\lim_{t \to \infty} \frac{\mathrm{E}[Q_i^{(\beta)}(t)]}{t} = 0, \tag{17}$$
$$\lim_{t \to \infty} \frac{\mathrm{E}[|Z_i(t)|]}{t} = 0. \tag{18}$$

The Lyapunov function $L(t)$ can be defined as the sum of the squared virtual queue lengths on time slot t.

$$L(t) = \frac{1}{2}\sum_{i=1}^{N} Z_i(t)^2 + \frac{1}{2}\sum_{i=1}^{N}\sum_{\beta\in\mathcal{B}_i} Q_i^{(\beta)}(t)^2.$$

The Lyapunov drift is given by $\Delta(t) = L(t+1) - L(t)$. The drift-plus-penalty algorithm runs by greedily choosing actions every time slot t so that the upper bound of $\Delta(t) - Vg(t)$ is minimized. In general, V indicating a non-negative controllable parameter controls the convergence time and the solution optimality. $-g(t)$ means a penalty.

Every time slot t, the drift-plus-penalty expression $\Delta(t) - Vg(t)$ satisfies the following property.

$$\begin{aligned}\Delta(t) - Vg(t) \le B - Vg(t) &\\ + \frac{1}{2}\sum_{i\in\mathcal{N}}\sum_{\beta\in\mathcal{B}_i} Q_i^{(\beta)}(t)(u_i^{(\beta)}(t) - u_i(t)) &\\ + \frac{1}{2}\sum_{i\in\mathcal{N}} Z_i(t)(\gamma_i(t) - u_i(t)), &\qquad (19)\end{aligned}$$

where $B = \frac{1}{2}\sum_{i\in\mathcal{N}}(u_i^{\max})^2 + \frac{1}{2}\sum_{i\in\mathcal{N}}\sum_{\beta\in\mathcal{B}_i}(u_i^{\max})^2$.

The drift-plus-penalty algorithm can greedily solve the optimization problem P_3. This algorithm first splits the right-hand side of Eq. (19) into the term relating to $\boldsymbol{\gamma}$ and the term relating to $\boldsymbol{\alpha}$ and greedily optimizes each of the terms. As a result, the algorithm can maximize the inferior limit of time average while holding Eqs. (15) and (16).

More precisely, the algorithm initializes the virtual queues $\{Q_i^{(\beta)}(0)\}_{\forall i\in\mathcal{N},\forall\beta\in\mathcal{B}}$ and $\{Z_i(0)\}_{\forall i\in\mathcal{N}}$ with zeros on time slot $t = 0$. Every time slot t, the algorithm first observes $\boldsymbol{\omega}(t)$ on time slot $t-1$. The algorithm then derives $\boldsymbol{\gamma}(t)$ by solving the following the optimization problem.

$$\begin{aligned}\max\quad & Vg(t) - \sum_{i\in\mathcal{N}} Z_i(t)\gamma_i(t),\\ \text{s.t.}\quad & 0 \le \gamma_i \le u_i^{\max}, \quad \forall i \in \mathcal{N}.\end{aligned}$$

Similarity, the algorithm derives suggested actions $\boldsymbol{S}(t) = \boldsymbol{\alpha}(t)$ such that

$$\begin{aligned}\min\quad & -\sum_{i\in\mathcal{N}} Z_i(t)u_i(t)\\ & + \sum_{i\in\mathcal{N}}\sum_{\beta\in\mathcal{B}} Q_i^{(\beta)}(t)[u_i^{(\beta)}(t) - u_i(t)].\end{aligned}$$

The algorithm provides each user $i \in \mathcal{N}$ with the suggestion $\alpha_i(t)$ and subsequently updates the virtual queues according to Eqs. (15) and (16). The algorithm does not require knowledge of the probabilities $\pi[\boldsymbol{\omega}]$.

5. Numerical Results

5.1 Evaluation Scenario

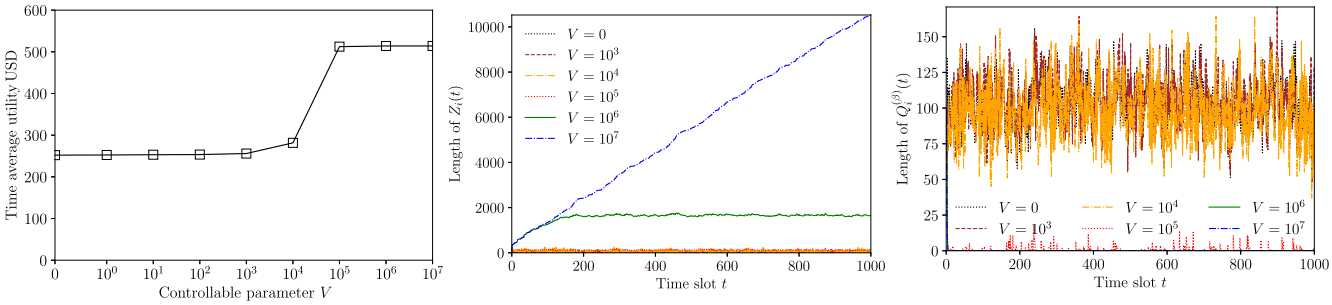
Table 3 presents the parameters used in the numerical experiments. There exist $N = 100$ DApp users and $M = 10$ CSPs. We assume that the total hash power over the blockchain network is equivalent to the total computing demands of N users. The computing demand of each user i is calculated by the total hash power over the blockchain network on 25 Apr. 2024 [23]. The service price (i.e., the payment cost) p^j per hash power is calculated by multiplying the service fee per power consumption and the maximum and minimum energy efficiency (i.e., the power consumption per hash power) of the top-10 application-specific integrated circuits (ASICs) in the expected daily profit, released until Mar. 2024 [24]. The service fee per power consumption is defined as the household average of electricity charge in the United States (i.e., 0.06 USD/kWh) plus the cloud service fee per power consumption (i.e., 0.12 USD/kWh). The mining reward R is set to 6.25 BTC as of Apr. 2024 [25]. The transaction fee r is the average of the transaction fees in 5000 blocks generated between 23 Mar. 2024 at 5:11:32 a.m and 26 Apr. 2024 at 12:03:58 a.m. [25]. It is assumed that the BTC to US dollar exchange rate is 1 BTC = 67259 USD, which is the average value between 23 Mar. 2024 and 25 Apr. 2024. We run each evaluation for $T = 1000$ time slots. We assume that the duration of a single time slot t is 10 minutes.

Table 3 Parameters used in the numerical experiments.

Parameter	Value
N	100
M	10
$d_i(t)$	$[5.0\times10^6, 6.0\times10^6]$ Peta hashes (PH)/time slot
C^j	$[6.0\times10^7, 7.0\times10^7]$ PH/time slot
$p^j(t)$	$[1.12\times10^{-8}, 1.41\times10^{-8}]$ BTC/PH
R	6.25 BTC/time slot
r	0.84 BTC/time slot
T	1000

It is difficult to directly compare the performance of the proposed method with those of the existing methods, as shown in Sect. 2. To confirm the effectiveness of the proposed method, we prepare two simple methods (i.e., a random method and a TOPSIS method [22] adopted in [14]). In the random method, each user randomly chooses a CSP from the available CSPs without any strategy. Recall that the existing method [14] does not support mining task offloading. Therefore, we slightly modify the TOPSIS method as follows: Each user first sets the relative importance (i.e., the weighted value) of each attribute, where the attributes are the computing resource capacity and the payment cost, provided by each CSP. Next, each user chooses the CSP with the highest weighted sum of the evaluations for each attribute from the available CSPs. It is assumed that the relative importance is given randomly for each user.

For the evaluation, we use the server with Intel(R) Xeon(R) Gold 6326 CPU (32 cores) and 256 GB memory. We use the two solvers: (1) an interior point optimizer (IPOPT) solver [27] for the non-linear problem and (2) a COIN-OR branch-and-cut (CBC) solver [28] for the linear problem. As for performance metrics, we use virtual queue lengths (i.e., $|Z_i(t)|$ and $|Q_i^{(\beta)}(t)|$), time average utility (Eq. (8)), proportional fairness (Eq. (6)), and execution time, respectively. The execution time is defined as the average of the time it takes to derive the actions of miners for each time

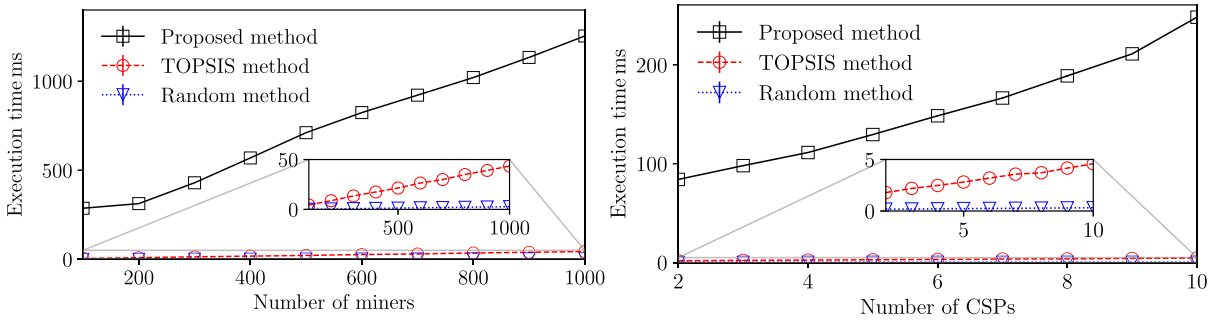


(a) Time average utility.

(b) Length of virtual queue $Z_i(t)$.

(c) Length of virtual queue $Q_i^{(\beta)}(t)$.

Fig. 2 Impact of V on time average utility and virtual queue lengths.



(a) Impact of N ($M = 10$).

(b) Impact of M ($N = 100$).

Fig. 3 Execution time.

slot. In what follows, we show the results in the average of 10 independent experiments.

5.2 Impact of Controllable Parameter V

Recall that the controllable parameter V affects the time average utility and the convergence speed of the virtual queue. To find the appropriate V that stabilizes the virtual queues fast while enhancing the time average utility, we first focus on the impact of V on the time average utility and the virtual queue lengths, as illustrated in Fig. 2. We observe from Fig. 2(a) that the time average utility rises with increase of V and subsequently stabilizes at $V \geq 10^5$. Focusing on the queue stability, we also observe from Fig. 2(b) that the lengths of virtual queues for all V except $V = 10^7$ stabilize until $t = 1000$. This result indicates that the virtual queue $Z_i(t)$ is mean rate stable (i.e., Eq. (16) is established) and thus the constraint (13) is satisfied. On the other hand, we confirm from Fig. 2(c) that $V = 10^6$ and $V = 10^7$ exhibit rapid convergence to zero, compared with the others. In other words, the virtual queue $Q_i^{(\beta)}(t)$ is mean rate stable (i.e., Eq. (15) is established) when $V = 10^6$ and $V = 10^7$. This indicates that the constraint (10) is satisfied. In what follows, we adopt $V = 10^6$ to balance both time average utility and queue stability.

5.3 Time Average Utility and Proportional Fairness

Table 4 presents the time average utility and the proportional fairness among three methods. We observe that the proposed method exhibits the highest time average utility of all methods. More precisely, the proposed scheme exhibits 1.46 and 5.08 times higher time average utilities compared with the TOPSIS and random methods, respectively. The proposed method has 6.5% and 35.2% higher proportional fairness than the TOPSIS and random methods. This indicates that the proposed method more fairly gives an opportunity for mining to miners than the others. The 6.5% difference in the proportional fairness may be negligible in the short term, but not in the long term. This indicates that the proposed method is a sophisticated strategy for mining task offloading that aims to maximize the proportional fairness of the long-term utility of each miner.

Table 4 Time average utility and proportional fairness.

Scheme	Time average utility USD	Std. of time average utility	Proportional Fairness	Std. of proportional fairness
Proposed method	519.5	2.4	625.3	0.5
Random method	102.2	1.5	462.5	1.4
TOPSIS method	354.8	5.0	587.1	1.4

5.4 Scalability

Finally, we turn attention to the scalability. Figure 3(a) (resp. Fig. 3(b)) illustrates the relationship between the number N of miners (resp. the number M of CSPs) and the execution

time. We confirm that the proposed and TOPSIS methods experience an increase in the execution time with N and M, respectively. In particular, the proposed method has a higher increase ratio of execution time than the others. However, we also confirm from Fig. 3(a) that the execution time of the proposed method is 1254 ms in case of $N = 1000$ and $M = 10$. In the mature PoW-based blockchain network (i.e., Bitcoin), there are 18589 nodes distributed across the world as of 28 Apr. 2024[†]. In particular, the top-3 countries (Germany, United States, and France) with their respective number of reachable nodes have 1926, 1835, and 421 nodes, respectively. The proposed method can support 51.9% of reachable nodes in Germany (i.e., 1000 nodes) within 1254 ms.

6. Conclusion

In this paper, we have proposed the mining task offloading strategy for the cloud service provider (CSP) selection problem in decentralized applications (DApps) to maximize the miner's expected utility. More specifically, we have formulated the CSP selection problem as the repeated stochastic game such that coarse correlated equilibrium (CCE) is achieved among miners. We have additionally developed the online algorithm to efficiently solve the repeated stochastic game with the assistance of the Lyapunov optimization and drift-plus-penalty algorithm. Through the numerical experiments, we have first confirmed how the controllable parameter of the drift-plus-penalty algorithm affects the time average utility and queue stability. Next, we have observed that the proposed algorithm exhibits higher time average utility and proportional fairness than the other methods. Finally, we have demonstrated that the proposed algorithm works well in an online fashion even when the number of miners increases to a thousand.

Acknowledgments

This work was supported in part by the Japan Society for the Promotion of Science (JSPS) Grant-in-Aid for Challenging Research (Exploratory) under Grant 22K19776, the JSPS KAKENHI (B) under Grant 24K02931, and the JSPS Grant-in-Aid for Young Scientists under Grant 23K16869, Japan.

References

[1] K. Yamada, T. Hara, and S. Kasahara, "A repeated stochastic game approach for offload mining in distributed applications in a permissionless blockchain network," Proc. International Conference Emerging Technologies for Communications, Nov. 2023.
[2] K. Yue, Y. Zhang, Y. Chen, Y. Li, L. Zhao, C. Rong, and L. Chen, "A survey of decentralizing applications via blockchain: The 5G and beyond perspective," IEEE Commun. Surveys Tuts., vol.23, no.4, pp.2191–2217, 2021.
[3] Z. Xiong, Y. Zhang, D. Niyato, P. Wang, and Z. Han, "When mobile blockchain meets edge computing," IEEE Commun. Mag., vol.56, no.8, pp.33–39, Aug. 2018.

[†]https://bitnodes.io/

[4] K. Gai, J. Guo, L. Zhu, and S. Yu, "Blockchain meets cloud computing: A survey," IEEE Commun. Surveys Tuts., vol.22, no.3, pp.2009–2030, 2020.
[5] Y. Jiao, P. Wang, D. Niyato, and K. Suankaewmanee, "Auction mechanisms in cloud/fog computing resource allocation for public blockchain networks," IEEE Trans. Parallel Distrib. Syst., vol.30, no.9, pp.1975–1989, Sept. 2019.
[6] K. Zhang, J. Cao, S. Leng, C. Shao, and Y. Zhang, "Mining task offloading in mobile edge computing empowered blockchain," Proc. IEEE International Conference on Smart Internet of Things (SmartIoT), pp.234–239, Aug. 2019.
[7] X. Liu, J. Wu, L. Chen, and C. Xia, "Efficient auction mechanism for edge computing resource allocation in mobile blockchain," Proc. International Conference on High Performance Computing and Communications; IEEE International Conference on Smart City; IEEE International Conference on Data Science and Systems (HPCC/SmartCity/DSS), pp.871–876, Aug. 2019.
[8] H. Wu, K. Wolter, P. Jiao, Y. Deng, Y. Zhao, and M. Xu, "EEDTO: An energy-efficient dynamic task offloading algorithm for blockchain-enabled IoT-edge-cloud orchestrated computing," IEEE Internet Things J., vol.8, no.4, pp.2163–2176, Feb. 2021.
[9] L. Yang, M. Li, H. Zhang, H. Ji, M. Xiao, and X. Li, "Distributed resource management for blockchain in fog-enabled IoT networks," IEEE Internet Things J., vol.8, no.4, pp.2330–2341, Feb. 2021.
[10] N.C. Luong, Z. Xiong, P. Wang, and D. Niyato, "Optimal auction for edge computing resource management in mobile blockchain networks: A deep learning approach," Proc. IEEE International Conference on Communications (ICC), pp.1–6, May 2018.
[11] T. Mai, H. Yao, N. Zhang, L. Xu, M. Guizani, and S. Guo, "Cloud mining pool aided blockchain-enabled Internet of Things: An evolutionary game approach," IEEE Trans. Cloud Comput., vol.11, no.1, pp.692–703, Jan. 2023.
[12] S. Jiang and J. Wu, "A game-theoretic approach to storage offloading in PoC-based mobile blockchain mining," Proc. International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing, MOBIHOC'20, New York, NY, USA, pp.171–180, Association for Computing Machinery, 2020.
[13] M. Liu, F.R. Yu, Y. Teng, V.C.M. Leung, and M. Song, "Joint computation offloading and content caching for wireless blockchain networks," Proc. IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp.517–522, April 2018.
[14] K. Moghaddasi and M. Masdari, "Blockchain-driven optimization of IoT in mobile edge computing environment with deep reinforcement learning and multi-criteria decision-making techniques," Cluster Comput., vol.27, pp.4385–4413, 2024.
[15] L.S. Shapley, "Stochastic games," Proc. National Academy of Sciences, vol.39, no.10, pp.1095–1100, Oct. 1953.
[16] M.J. Neely, "A Lyapunov optimization approach to repeated stochastic games," Proc. Annual Allerton Conference on Communication, Control, and Computing, pp.1082–1089, Oct. 2013.
[17] X. Li, Q. Huang, and D. Wu, "A repeated stochastic game approach for strategic network selection in heterogeneous networks," Proc. IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Honolulu, HI, USA, pp.88–93, IEEE, April 2018.
[18] H. Moulin and J.P. Vial, "Strategically zero-sum games: The class of games whose completely mixed equilibria cannot be improved upon," Int. J. Game Theory, vol.7, no.3-4, pp.201–221, Sept. 1978.
[19] M. Bowling and M. Veloso, "Multiagent learning using a variable learning rate," Artificial Intelligence, vol.136, no.2, pp.215–250, April 2002.
[20] E. Wei and A. Ozdaglar, "Distributed alternating direction method of multipliers," Proc. IEEE Conference on Decision and Control (CDC), pp.5445–5450, Feb. 2012.
[21] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," Proc. AAAI Conference on Artificial

Intelligence, vol.30, no.1, March 2016.
[22] B. Uzun, M. Taiwo, A. Syidanova, and D. Uzun Ozsahin, "The technique for order of preference by similarity to ideal solution (topsis)," Application of Multi-Criteria Decision Analysis in Environmental and Civil Engineering, D. Uzun Ozsahin, H. Gökçekuş, B. Uzun, and J. LaMoreaux, eds., Professional Practice in Earth Sciences, pp.25–30, Springer International Publishing, Cham, 2021.
[23] Luxor Technology, "Network hashrate," https://data.hashrateindex.com/chart/bitcoin-network-hashrate/, 2023. Accessed 18 Oct. 2023.
[24] Luxor Technology, "Bitcoin mining asics|hashrateindex," https://hashrateindex.com/rigs/, 2023. Accessed 18 Oct. 2023.
[25] blockchain.com, "Blockchain.com," https://www.blockchain.com/explorer/assets/btc/, 2023. Accessed 18 Oct. 2023.
[26] M.J. Neely, Stochastic Network Optimization with Application to Communication and Queueing Systems, Synthesis Lectures on Learning, Networks, and Algorithms, Springer International Publishing, Cham, 2010.
[27] COIN-OR Foundation, "Ipopt," https://github.com/coin-or/Ipopt/, 2023. Accessed 23 Dec. 2023.
[28] COIN-OR Foundation, "Cbc," https://github.com/coin-or/Cbc/, 2023. Accessed 23 Dec. 2023.

Kota Yamada received the B.Ec. degree from Hokkaido University, Hokkaido, Japan, in 2022 and the M.Eng. degree from Nara Institute of Science and Technology, Nara, Japan, in 2024. His research interests include game-theoretic approaches.

Takanori Hara received the M.Eng. and Ph.D. degrees from Nara Institute of Science and Technology, Nara, Japan, in 2018 and 2021. He is currently an Associate Professor with the Division of Information Science, Graduate School of Science and Technology, Nara Institute of Science and Technology, Japan. His research interests include AI/ML empowered networking, network virtualization, eBPF/XDP, pedestrian navigation, and game-theoretic approaches.

Shoji Kasahara received the B.Eng., M.Eng., and Dr. Eng. degrees from Kyoto University, Kyoto, Japan, in 1989, 1991, and 1996, respectively. Currently, he is a Professor of Division of Information Science, Nara Institute of Science and Technology, Nara, Japan. His research interests include stochastic modeling and analytics of large-scale complex systems based on computer/communication networks.