Bandit Learning based Stable Matching for Decentralized Task Offloading in Dynamic Fog Computing Networks

Hoa Tran-Dang and Dong-Seong Kim

***Abstract*—This paper deals with the task offloading problem in the dynamic fog computing networks (FCNs) that involves the task and resource allocations between a set of task nodes (TNs) having task computation needs and a set of helper nodes (HNs) having available computing resources. The problem is associated with the presence of selfishness and rational nodes of these nodes, in which the objective of TNs is to minimize the task completion time by offloading the tasks to the HNs while the HNs tend to maximize their monetization of task offloading resources. To tackle this problem, we use the fairness and stability principle of matching theory to assign the tasks of TNs to the resources of HNs based on their mutual preferences in a decentralized manner. However, the uncertainty of computing resource availability of HNs as well as dynamics of QoS requirements of tasks result in the lack of preferences of TN side that mainly poses a critical challenge to obtain a stable and reliable matching outcome. To address this challenge, we develop the first, to our knowledge, Thompson sampling based multi-armed bandit (MAB) learning to acquire better exploitation and exploration trade-off, therefore allowing TNs to achieve the informed preference relations of HNs quickly. Motivated by the above considerations, this paper aims at design a bandit learning based matching model (BLM) to realize the efficient decentralized task offloading algorithms in the dynamic FCNs. Extensive simulation results demonstrate the potential advantages of the TS based learning over the ϵ-greedy and UCB based baselines.**

***Index Terms*—Decentralized task offloading, exploitation and exploration dilemma, fog computing network, multi-armed bandit, stable matching, Thompson sampling.**

I. Introduction

A. Context and Motivations

PRACTICALLY, the Internet of things (IoT) has become an integral element for realizing smart practical systems such as smart cities [1], smart factories [2], smart logistics, and supply chain [3], [4]. The fundamental aspect of IoT-based systems is to connect all devices through the Internet protocol to exchange high volume data and process them to create smart services and applications. Owing to limited computation resources, network, storage, and energy, IoT devices are inadequate for executing all computational tasks, especially tasks with huge volumes and complex data structures. So far, cloud computing still has been an essential solution to this problem because it provides powerful resources to support the task computation efficiently through different on-demand services (i.e., PaaS, IaaS, SaaS) [5], [6]. Nevertheless, cloud computing-based solutions may be inadequate to satisfy the expected quality of service (QoS) and quality of experience (QoE) requirements for certain types of latency-sensitive applications because of the long physical distance between the IoT devices and the remote cloud servers, scarce spectrum resources, and intermittent network connectivity.

This context has led to an integration of fog computing networks (FCNs) in middle between the end devices and cloud layer to process and offload most tasks on behalf of the cloud servers, thereby allowing for the QoS and QoE requirements to be satisfied [7], [8]. Besides providing the cloud like services to end devices, the fog computing potentially improves the performance of fog-based systems such as reduction of service delay [9] and energy saving [10].

However, to realize these benefits of fog computing paradigm, there requires efficient resource allocation strategies to perform task offloading operations [11] that is known as multi-task multi-helper (MTMH) problem in the FCNs [12]. A critical issue to be solved in this problem is how to efficiently map multiple tasks into multiple helper nodes in order to decrease the service latency. Indeed, there are many factors that challenge the design and development of effective offloading strategies such as the heterogeneity of fog nodes, various types of computational tasks with different QoS requirements [13], and time-varying nature of fog node capabilities. There are a large number of centralized optimization techniques and algorithms proposed in the literature to provide optimal solutions to the aforementioned resource allocation problems [14]. However, these solutions require a centralized control to gather the global system information, thus incurring a significant overhead and computation complexity of algorithms. This complexity is further amplified by the rapidly increase of density and heterogeneity of FCNs [15] when dealing with combination integer programming problems [16].

Obviously, the aforementioned limitations of global opti-

Manuscript received July 24, 2023; approved for publication April 7, 2024; approved for publication by Qiao, Daji Division 3 Editor, April 21, 2024.

This work was partly supported by Innovative Human Resource Development for Local Intellectualization program through the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (IITP-2024-2020-0-01612, 40%), Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education (2018R1A6A1A03024003, 20%) and Korea Research Fellowship Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (RS-2023-00249687, 40%).

H. Tran-Dang and D.-S. Kim are with department of IT Convergence Engineering, Kumoh National Institute of Technology, Korea, email: {hoa.trandang, dskim}@kumoh.ac.kr.

D.-S. Kim is the corresponding author.

Digital Object Identifier: 10.23919/JCN.2024.000017

1229-2370/24/$10.00 © 2024 KICS

mization approaches in such the complex environment leads to a fundamental shift from the traditional centralized mechanism toward decentralized one. The existing literature has acknowledged a class of game theory based decentralized task offloading (DTO) solutions developed to avoid the cost-intensive centralized resource management as well as substantially reduce the complexity of algorithms [17], [18]. Despite their potentials, these approaches pose several shortcomings. First, classical game theoretical algorithms such as best response require some information regarding actions of other players [19]. Correspondingly, many assumptions are introduced in the game theory based algorithms to simplify the system models that, in some case, are impractical. Second, most game-theoretic solutions, for example, Nash equilibrium, investigate one-sided stability notions in which equilibrium deviations are evaluated unilaterally per player [20]. In addition, the stability must be achieved by both sides of players (i.e., TNs and HNs) in the context of fog computing environment to handle the selfishness and rationals of individual players.

Recently, matching theory (MT) has emerged as a promising technique for solving offloading problems in the fog computing because it can alleviate the shortcomings of game theory and optimization-based schemes [21], [22]. Basically, matching theory provides mathematically tractable solutions for the combination problem of matching players in two disjoint sets, depending on the individual information and preference of each player. Following the deferred acceptance (DA) procedure [23], the matching game between the two sides of players achieves the stability in a distributed manner. However, most of MT-based algorithms assume that the full preferences of agents are known $a\ priori$, which is not realistic in the context of the dynamic FCNs. For instance, TNs are likely to be uncertain about the computing capability (i.e., CPU frequency, queuing delay) of HNs at time of offloading decision making since it is time varying. As a results, the preference relation of TN to HNs are unpredictable. Therefore, to address this issues, the TNs must interact iteratively with the HNs to learn their unknown preferences.

Multi-armed bandit (MAB) is a common strategy to model this type of learning process, which aims to solve the exploitation and exploration dilemma [24]. Several algorithms including ϵ-greedy, upper confidence bound (UCB), and Thompson sampling (TS) are proposed for the player (i.e., learner) to select the optimal arm, which offers the highest cumulative reward [25]. Basically, the ϵ-greedy algorithm is simple to implement with low complexity but heavily relying on the random choice, thus occurring the long convergence. TS is Bayesian method [26], in which the player keeps a posterior distribution over the expected arm rewards, and at each round takes a sample from each arm's posterior, and then, plays the arm with the largest sample. Reward observed from the played arm is then used to update its posterior. This sampling strategy allows the arm to frequently select the arms whose probabilities of being optimal are the highest based on their posteriors and to occasionally explore inferior arms to refine their posteriors. Meanwhile, the UCB strategy is to play the arm with the highest UCB index to tradeoff exploration and exploitation, which is usually composed of sample mean reward of an arm plus an exploration bonus that accounts for the uncertainty in the arm's reward estimates [27]. Unlike TS, performance of this type of policies heavily rely on the confidence sets used to compute the exploration bonus [28]. This together with the superior performance of TS evaluated in numerous applications [29] motivate us to investigate a TS based learning for our problem.

B. Paper Contributions

In consideration of the aforementioned issues, this paper aims at developing a DTO algorithm for the heterogeneous dynamic FCNs that combines the matching theory and TS-based bandit learning method. Generally, the main contributions of this work are summarized as follows:

- We model the task offloading problem of MTMH scenarios as a two-sided matching game between TNs and HNs, in which the preference relations are unknown a prior to TNs.
- To deal with the uncertain information of HNs (i.e., computing capability, queuing waiting), we propose TS-based bandit learning that allows TNs to obtain the preference relation quickly.
- Based on the achieved preference relations of TNs, we propose a bandit learning-based matching model called BanLeMat to assign the tasks to the HNs in a distributed manner.
- Through extensive simulations, we present the feasibility and advantages of proposed BanLeMat model compared to existing state-of-the-art solutions.

C. Paper Organization

The rest of this paper is organized as follows. Section II highlights the typical related works regarding the bandit learning-based DTO algorithms in different fog computing environments. Section III introduces the system model and formulates the task offloading problem for the heterogeneous dynamic FCNs. Section IV proposes and describes the design of BanLeMat model. Section V shows the simulation results and comparative analysis. Section VI concludes the paper and discusses the open directions to extend the current work.

II. Backgrounds and Related Works

A. TS for MAB

TS is a popular probabilistic algorithm used in the context of multi-armed bandit problems. In the context of MAB problem, a player is faced with a row of K arms, each with an unknown probability distribution θ_k of providing a reward. The goal is to maximize the total reward over a series of pulls.

TS introduces a Bayesian approach to the exploration-exploitation dilemma inherent in such problems. The algorithm maintains a probability distribution over the possible reward distributions for each arm. After each pull, the probability distribution is updated based on the observed reward. The next arm to be pulled is then chosen based on a sample from the

Algorithm 1: TS Algorithm for MAB Problem

Input: K arms
Output: Sequential arms to pull
Initialization: $\forall k \in \{0, \cdots, K\}$, $a_k = b_k = 1$.
for $t = 1, 2, \cdots,$ **do**
  for $k = 1, 2, \cdots, K$ **do**
    Sample $\theta_k \approx$ Beta(α_k, β_k)
  Pull arm k if it has maximal value of θ_k
  Observe the reward $r_k(t)$
  Update: $(\alpha_k, \beta_k) \leftarrow (\alpha_k + r_k(t), \beta_k + 1 - r_k(t))$

updated distribution. Algorithm [**?**] shows the procedure code to implement TS for MAB as following steps.

- Step 1 (Initialization): The algorithm starts with a prior distribution for each arm. Commonly, a Beta distribution is used as a conjugate prior for Bernoulli-distributed rewards. For each arm k, it maintains two parameters α_k (number of successes) and β_k (number of failures).
- Step 2 (Sample): For each arm, sample a value from its current distribution.
- Step 3 (Select arm): Choose the arm with the highest sampled value θ_k.
- Step 4 (Pull arm): Pull the chosen arm and observe the reward.
- Step 5 (Update): : Update the parameters of the distribution for the chosen arm based on the observed reward. For a Bernoulli reward, increment α_k for success and β_k for failure.
- Step 6 (Repeat): Go back to step 2 and repeat the process.

For non-stationary bandit with time changing reward distribution θ_k, the TS algorithm is applicable to achieve the improved performance compared to UCB or greedy method [29].

B. Related Works

This section presents a review of recent literature on using the bandit learning method to design the task offloading algorithms in the fog computing environment.

The principle of bandit learning techniques can be used in a family of bandit convex optimization (BCO) algorithms to solve convex optimization problems in which the objective functions and constraints are time varying [30], [31]. These similar features are prevalent in the computation offloading optimization problems of fog-based IoT systems where the function of accumulated network delay (need to be minimized) and long-term workload balancing constraints are variant over time. Based on this investigation, the studies [32], [33] proposes a method for managing the task offloading problems in the large-scale and dynamic IoT systems using BCO. Concretely, a family of online bandit saddle-point (BanSaP) schemes are developed, which adaptively adjust the online operations based on (possibly multiple) bandit feedback of the loss functions, and the changing environment. The authors demonstrate the effectiveness of the proposed method through simulations, showing that it can simultaneously yields sublinear dynamic regret and fit in cases that the best dynamic solutions vary slowly over time. In particular, numerical experiments in the fog computing offloading tasks corroborate that the proposed BanSap approach offers competitive performance relative to existing approached based on gradient feedback.

Concerning the execution of hard real-time tasks within fixed deadlines in the IoT systems, the paper [34] introduces a two-tiered framework to offload the tasks using the fog and cloud computing instead of sensor nodes (SNs). To facilitate the task processing, a directed acyclic task graph (DATG) is formed by breaking down high-level tasks into smaller subtasks. The tasks are initially offloaded to a nearby FN using a greedy selection to avoid the combinatorial optimizations at the SNs, thus saving time and energy. As IoT environments are dynamic, adaptive solutions are necessary. An online learning scheme called ϵ-greedy nonstationary MAB-based scheme (D2CIT) is proposed for task allocation among FNs. D2CIT enables the selection of a set of FNs for subtask distribution, parallel execution with minimum latency, energy, and resource usage. Simulation results show that D2CIT reduces latency by 17% and offers a speedup of 59% compared to existing online learning-based task offloading solutions in fog environments due to the induced parallelism.

The paper [35] develops an online task offloading strategy that minimizes the long-term cost that takes into account factors such as latency, energy consumption, and switching cost in dynamic FCNs characterized by the abrupt change of system parameters at unknown times. Additionally, the fact that queried nodes can only provide feedback on processing results after task completion is considered. To address these challenges, an effective bandit learning algorithm called BLOT is proposed to solve the non-stationary stochastic programming problem under a bandit model. The research also demonstrates the asymptotic optimality of BLOT in a non-stationary fog-enabled network and presents numerical experiments to justify the superior performance of proposed algorithm compared to the baseline approaches. However, the performance of algorithm is only evaluated in the small-scale network with one TN and nine HNs, thus inapplicable for large-scale networks where the matching conflicts can occur.

The paper [36] proposes a learning-based approach for task offloading in fog networks with the goal of reducing latency for delay-sensitive applications. The approach integrates Combinatorial MAB (CMAB) which is a generalization of the classical MAB problem to find the best set of arms to pull together, rather than finding the best single arm to pull [37]. Initially, the algorithm being suggested acquires knowledge of the shared computing resources of fog nodes, with minimal computational expenses. Next, the objective is to reduce the time taken for task offloading by simultaneously refining the task allocation decision and spectrum scheduling. Ultimately, simulation outcomes reveal that the proposed approach surpasses the conventional UCB algorithm with regards to delay performance and maintains extremely low offloading delays in a dynamically evolving system.

The authors in [38] also tackle the task offloading issues but consider the case of vehicular fog computing (VFC). The VFC

environment with diverse modes of mobility introduces unpredictability with regards to the availability of resources and their demand, which create unavoidable obstacles in making optimal decisions for offloading. Moreover, these uncertainties pose additional challenges for task offloading in the face of an oblivious adversary attack and the risks associated with data privacy. The authors then have developed a novel algorithm for adversarial online learning with bandit feedback that leverages the principles of the adversarial MAB theory. This algorithm is designed to facilitate efficient and simple decision making for offloading by optimizing the selection of FNs, with the goal of minimizing costs associated with offloading services such as delay and energy usage. Fundamentally, the proposed approach involves implicitly adjusting the exploration bonus during selection, and incorporating assessment rules that account for the volatile nature of resource supply and demand. Theoretically, the input-size dependent selection rule allows for the selection of an appropriate FN without the need to explore sub-optimal actions. Additionally, the appropriate score patching rule facilitates quick adaptation to changing circumstances, reducing variance and bias, and ultimately achieving a better balance between exploitation and exploration. Simulation results demonstrate the effectiveness and robustness of the proposed algorithm.

The authors in [39] consider the task offloading in the fog computing network with time-varying stochastic time of arrival tasks and channel conditions. Due to the unavailability of global knowledge of all fog nodes in practice, the problem is modeled as a CMAB problem, without prior information about channel conditions and stochastic task arrival characteristics. To address this problem, the paper proposes the WFCUCB algorithm, which extends the classical CMAB problem to include one *i.i.d.* variable and one non-stationary random variable. The paper's numerical results demonstrate that the WFCUCB algorithm is capable of fast learning and achieves superior performance compared to other possible strategies.

Due to the highly dynamic environment of the vehicular network, it is challenging to ensure that task offloading delay is minimized. To address this issue, task replication is introduced into the VEC system as proposed in the study [40], where multiple replicas of a task are offloaded simultaneously to several vehicles, and the task is considered completed once the first response among the replicas is received. The impact of the number of task replicas on the offloading delay is examined, and the optimal number of task replicas is determined through a closed-form approximation. Using these findings, a learning-based task replication algorithm (LTRA) is developed using CMAB theory. The LTRA algorithm is designed to operate in a distributed manner and can adapt automatically to the VEC system's dynamics. The proposed algorithm's delay performance is evaluated using a realistic traffic scenario. The results show that, under the simulation settings, the optimized LTRA algorithm with a specific number of task replicas can decrease the average offloading delay by more than 30% compared to the benchmark without task replication while also improving the task completion ratio from 97% to 99.6%.

The authors in [41] also study the task offloading in the VFC systems with uneven workload distribution and the reliability of the communication between the FNs. In the work, they utilized the concept of CMAB to facilitate the selection of task offloading destinations in a distributed manner, without overburdening system resources. This is achieved by replicating tasks across multiple destination nodes and selecting the optimal number of replicating nodes to ensure reliability and minimize delay in a vehicular resource-sharing environment. This approach also reduces overall system residence time and enhances task delivery ratio by reducing task failures. Additionally, redundant tasks are eliminated from node queues after receiving the first response from candidate nodes in the surrounding area. They compared their solution with other baseline approaches based on several performance metrics, such as task residence time, end-to-end delays, delivery rate, and utilization ratio. The simulation results demonstrate that the proposed learning-based task offloading solution effectively utilizes resources and ensures its effectiveness over other approaches compared to the algorithms presented in [40].

The study presented in [42] addresses the problem of data offloading in heterogeneous and dynamic fog computing-enabled wireless sensor networks. The authors model the data offloading problem as a contextual MAB problem that uses the heterogeneity of sensor nodes (SNs) as contextual information. The proposed algorithm for dynamic node movement in urban environments has been enhanced to ensure stable performance of the collaborative system despite the complexities and changes of the urban environment. By analyzing and simulating human movement data in such settings, the proposed approach can effectively minimize offloading delay and increase the success rate of offloading.

The work [43] presents the combination of learning and matching to design a learning-matching algorithm for task offloading and resource allocation in the VFC systems. Considering the uncertainty of information of systems, the algorithm proposes a pricing based model iterated over time slot to learn the uncertainty as well as handle the matching conflict. Evaluated by extensive simulations, the proposed algorithm can achieve bounded deviation from the optimal performance without the availability of global information.

III. System Model

A. Fog Computing Network

In the general system architecture, a FCN consists of M TNs and N HNs co-existing in an area to support computing various types of tasks as shown in Fig. [**?**]. Without the loss of generality, this paper considers the FCN with an equal number M of TNs and HNs to investigate the performance of distributed task offloading in form of one-to-one matching with bandit learning integration. Notably, scenarios with unequal number of TNs and HNs can be deduced or extended from the current network configuration. For example, as $M > N$, a part of tasks that are unable to be processed by HNs are managed by the cloud. Indeed, as $M < N$, there exist M HNs with the most ample computing capability selected for offloading the tasks from M TNs. Furthermore, as a HN is able to process multiple tasks via, for example, parallel virtual machines, it is

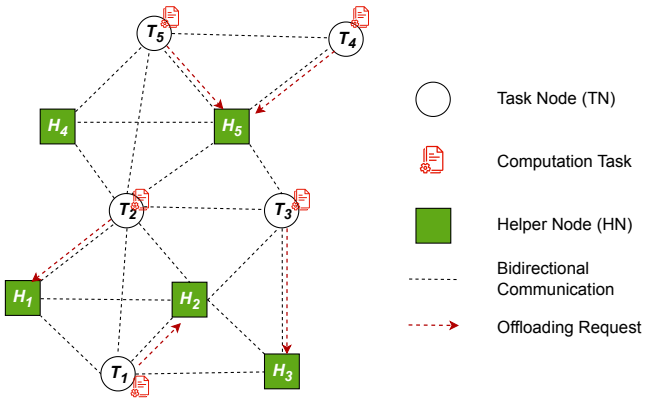

Fig. 1. An illustrative model of FCN with two types of FNs including TNs and HNs.

TABLE I
NOTATIONS AND VARIABLES.

Symbols	Definitions
FN, TN, HN	Fog Node, task node, helper node
$\mathcal{T}$, $\mathcal{H}$	Set of TNs (tasks), set of HNs
M	Number of TNs, number of HNs in FCN
T_i, H_j	TN i, HN j
t	The t^{th} time slot
$T_i(t)$	The task generated by TN i in time slot t
$f_j(t)$	CPU frequency of H_j in time slot t (cycles/s)
$\rho_j(t)$	Processing density of H_j in time slot t (cycles/bit)
$L_i(t)$	The task size of $T_i(t)$ (bits)
$\Gamma_i(t)$	Computation complexity of $T_i(t)$ (cycles/bit)
$D_i^{max}(t)$	Maximum permitted latency for $T_i(t)$
$D_{ij}(t)$	Latency for executing $T_i(t)$ by HN H_j
$\delta_{ij}^{tx}(t)$	Transmission delay from T_i to H_j
$R_{ij}(t)$	Data rate between T_i and H_j
$\delta_j^w(t)$	Waiting delay in the queue of H_j
$Q_j(t)$	Queue length of H_j in time slot t (bits)
$\delta_{ji}^p(t)$	Delay to process T_i by H_j

worth to investigate the many-to-one matching model for task offloading as extension works.

In this work, we define the set of TNs and HNs as $\mathcal{T} = \{T_1, \cdots, T_i, \cdots, T_M\}$ and $\mathcal{H} = \{H_1, \cdots, H_j, \cdots, H_M\}$, respectively. FNs are heterogeneous in terms of computing capability, storage, and connectivity technology. In addition, we assume that all the FNs can connect together through wireless links. We adopt a time-slot model where the total time period is divided into K discrete intervals, the set of which is denoted as $\mathcal{S} = \{1, \cdots, t, \cdots, K\}$. At the t^{th} slot, each TN i generates a corresponding task $T_i(t)$, which is represented by a tuple $T_i(t) = \{L_i(t), \Gamma_i(t), D_i^{max}(t)\}$, where $L_i(t)$ is the task size (bits), $\Gamma_i(t)$ is the computation complexity of task (CPU cycles/bit), and $D_i^{max}(t)$ is the maximum permitted latency for the task T_i at time slot t. The specification of task varies across different slots. In addition, the tasks are assumed to be split arbitrarily, thus each task T_i should be either allocated as one whole piece to one neighboring HN or processed locally.

The unfinished tasks are summed to be cached in a first-in first-out (FIFO) queue at each FN. Due to the limited computation and storage resource within one individual FN, the tasks processed locally usually experience high latency, which degrades QoS and QoE. To enable low-latency processing, TNs may offload some of its computation burdens to the nearby HNs. These HNs typically posses more computation and storage resources and are deployed to help other TNs on demand.

In accounting for the dynamic of fog computing environment, the computing capability of a HN H_j is represented through CPU frequency $f_j(t)$ (cycles/s), and CPU processing density $\rho_j(t)$ (cycles/bit), which are assumed to vary over different time slots.

Table I summarizes the key variables and notations used in this paper.

B. Problem Formulation

Denote the set of task offloading decisions between M TNs and M HNs as $\alpha(t) = \{\alpha_{ij}(t)\}$, where each element is a binary variable, i.e., $\alpha_{ij}(t) \in \{0, 1\}$. $\alpha_{ij}(t) = 1$ means that the task T_i is decided to be offloaded by H_j in the time slot t; $\alpha_{ij}(t) = 0$, otherwise. $D_{ij}(t)$ is the total delay when H_j is assigned to process T_i and is calculated as follow:

$$D_{ji}(t) = \delta_{ij}^{tx}(t) + \delta_j^w(t) + \delta_{ji}^p(t), \tag{1}$$

where $\delta_{ij}^{tx}(t)$ is the transmission delay, $\delta_j^w(t)$ is the waiting delay in queue of $H_j(t)$, and $\delta_{ji}^p(t)$ is the processing delay by $H_j(t)$. Given the data rate $R_{ij}(t)$ (bits/s) between T_i and H_j at time slot t, we can achieve $\delta_{ij}^{tx}(t) = L_i(t)/R_{ij}(t)$. The processing delay is derived as:

$$\delta_{ji}^p(t) = \frac{L_i(t)\Gamma_i(t)}{f_i(t)}. \tag{2}$$

The waiting delay is measured by H_j as follow:

$$\delta_j^w(t) = \frac{Q_j(t)\rho_j(t)}{f_j(t)}, \tag{3}$$

where $Q_j(t)$ is the queue length (bits) of H_j in the time slot t.

For TNs, the objective of offloading their tasks to HNs is to minimize the long-term average delay $\overline{D}$, which is defined as follow:

$$\overline{D} = \lim_{K\to\infty} \frac{1}{K} \sum_{t=1}^{K} \sum_{i=1}^{M} \sum_{j=1}^{M} \alpha_{ij}(t) D_{ij}(t). \tag{4}$$

Practically, the FNs are managed by different service providers which aim at maximizing the revenue by providing the best services (PaaS, IaaS, and SaaS) for task offloading operations. Therefore, the preference relation of HN is based on the pay-off that it receives to process the tasks. For each $H_j \in \mathcal{H}$, the cost to process a bit of task T_i is denoted as c_{ji}. The total pay-off received by H_j when offloading T_i is expressed as $C_{ji}(t) = \frac{D_{ij}(t)}{D_i^{max}} c_{ji} L_i(t)$. For HNs, the objective of offloading their tasks from TNs is to maximize the average pay-off $\overline{C}$ over time slots, which is defined as follow:

$$\overline{C} = \lim_{K\to\infty} \frac{1}{K} \sum_{t=1}^{K} \sum_{i=1}^{M} \sum_{j=1}^{M} \alpha_{ij}(t) C_{ji}(t). \tag{5}$$

Definitely, the task offloading optimization problem is formulated as follows:

$$\begin{aligned} \textbf{P:} \quad & \min_{\alpha_{ij}(t)} \overline{D} \quad \& \quad \max_{\alpha_{ij}(t)} \overline{C} \\ \text{s. t.} \quad & C_1 : \alpha_{ij}(t) \in \{0,1\}, \forall \{i,j,t\} \in \{\mathcal{T}, \mathcal{H}, \mathcal{S}\}, \\ & C_2 : \sum_{i=1}^{\mathcal{T}} \alpha_{ij}(t) \leq 1, \forall j \in \mathcal{H}, \forall t \in \mathcal{S}, \\ & C_3 : \sum_{j=1}^{\mathcal{H}} \alpha_{ij}(t) \leq 1, \forall i \in \mathcal{T}, \forall t \in \mathcal{S}. \end{aligned} \tag{6}$$

Here, the constraints C_1, C_2, and C_3 guarantee that at each time slot t, a TN's task can be offloaded to only one HN, and a HN can process at most one task of TN.

There are two difficulties in solve the above problem. First, it is a stochastic programming problem. The exact information about the delay $D_{ij}(t)$ is not available before the task $T_i(t)$ is completed. In addition, event if $D_{ij}(t)$ is estimated a priori, this problem is still a combinatorial optimization problem and the complexity is in the order of $\mathcal{O}(M^{2K})$. This is due to the fact that the previous offloading decisions determine the queue length in each HN and further affect the decisions of future tasks.

IV. BLM Model Description

A. OTO Matching Model for MTMH Problems

Given the constraints C_2 and C_3 of problem **P**, the task offloading problem can be modeled as an one-to-one (OTO) matching game between players of two sets: $\mathcal{T}$ and $\mathcal{H}$, which is defined as follow.

Definition 4.1: The OTO matching is a *function* $\mathcal{M}$: $\mathcal{T} \cup \mathcal{H} \mapsto \mathcal{T} \cup \mathcal{H}$ such that the three following constraints are satisfied:

- For any $T_i \in \mathcal{T}$, $\mathcal{M}(T_i) \in \mathcal{H} \cup \{T_i\}$,
- For any $H_j \in \mathcal{H}$, $\mathcal{M}(H_j) \in \mathcal{T} \cup \{H_j\}$,
- For any $T_i \in \mathcal{T}$ and $H_j \in \mathcal{H}$, $T_i = \mathcal{M}(H_j)$ if and only if $H_j = \mathcal{M}(T_i)$.

In the OTO matching model, each agent T_i can only be matched with one agent H_j, and T_i remains unmatched if $\mathcal{M}(T_i) = T_i$. The objective of matching is to reach the stable status for all pairs.

Definition 4.2: A matching $\mathcal{M}$ is pairwise stable if there is no block pair (T_i, H_j).

Definition 4.3: (T_i, H_j) is a block pair for a matching $\mathcal{M}$ if three following conditions are satisfied:

- $\mathcal{M}(T_i) \neq H_j$,
- $H_j \succ_{T_i} \mathcal{M}(T_i)$,
- $T_i \succ_{H_j} \mathcal{M}(H_j)$,

where $\succ_{T_i}$ and $\succ_{H_j}$ represent two preference relations allowing the TNs and HNs from each of the two sets to express preferences over the opposite nodes.

Each node builds a ranking of other nodes in the opposite side individually using this preference relation, and then creating its own preference list (PL). With the full connected FCNs, each node has a complete PL over the nodes on the opposite side. Assume that each $T_i \in \mathcal{T}$ has a PL denoted as $\mathcal{P}(T_i) = (H_2, H_4, \cdots, H_j, \cdots, H_M, T_i)$. This means that T_i prefers agent H_2 to H_4 (i.e., $H_2 \succ_{T_i} H_4$).

To achieve the PL, each node is based on its preference values when matching with all the nodes of opposite set. These values are usually determined by utility functions taking account the objective of matching game. In this paper, for each T_i, its preference value for H_j is quantified by an unknown value $x_{ij} \in [0,1]$. For two different H_j and $H_{j'}$, $x_{ij} > x_{ij'}$ implies that $H_j \succ_{T_i} H_{j'}$. For the objective of delay minimization, the preference value taking into account the capability of HNs to processing the tasks serve as the reward offered by HN. Furthermore, to limit the rewards to the range (0,1), we use the sigmoid function and is calculated as follow:

$$x_{ij}(t) = \frac{1}{1 + e^{-(D_i^{max}(t) - D_{ij}(t))}}. \tag{7}$$

Similarly, denote y_{ji} as the preference value of H_j for T_i. For two different player T_i and $T_{i'}$, $y_{ji} > y_{ji'}$ implies that H_i prefers T_i to $T_{i'}$. The ranking for the reference of each H_j is known after it receives all the offloading requests sent from players (TNs). We use $C_{ji}(t)$ to express the preference values of T_i for H_j (i.e., $y_{ji}(t) = C_{ji}(t)$). For any two tasks T_i and $T_{i'}$ requested to be offloaded by a HN H_j, the preference relation is as follow: $T_i \succ_{H_j} T_{i'}$ if and only if $y_{ji} > y_{ji'}$.

B. BLM Model and Algorithm

Due to the dynamic nature of fog computing environment characterized by the time-varying capabilities of HNs and QoS requirements of tasks, $x_{ij}(t)$ are unknown a priori and must be learned by MAB learning. In this context, TNs and HNs plays roles as players and arms, respectively.

At each round $r = 1, 2, \cdots, R$ of time slot $t = 1, 2, \cdots, K$, each player T_i attempts to pull an arm H_j. When multiple player attempt to pull the same arm, a matching conflict occurs and only the player preferred most by this arm is accepted. If a player T_i wins the matching conflict, it will receive a random reward $x_{ij}(t, r)$ derived from (7). Otherwise, if failing the conflict, T_i is unmatched in this round and receive $x_{ij}(t, r) = 0$. Over R round, the average reward $\overline{X}_{ij}(t)$ received by T_i when attempting an arm H_j is estimated as follow:

$$\overline{X}_{ij}(t) = \frac{\sum_{r=0}^{R} \gamma^r x_{ij}(t, r)}{\sum_{r=0}^{R} \gamma^r}, \tag{8}$$

where $\gamma \in (0, 1)$ serves as the discount factor.

Fig. 2 illustrates BLM model including the MAB learning and resource competition for matching.

Algorithm 2 presents the procedures to implement the preference learning and matching conflict handling.

The algorithm takes the player set $\mathcal{T}$ and arm set $\mathcal{H}$ as input. For each player T_i and arm H_j, the algorithm maintains a Beta distribution $Beta(a_{ij}, b_{ij})$ for the preference value. Initially, the distribution is $Beta(1, 1)$ corresponding to the uniform distribution on $[0, 1]$. It will be later updated based on observed feedback and tend to concentrate on the mean value $\overline{X}_{ij}(t)$.

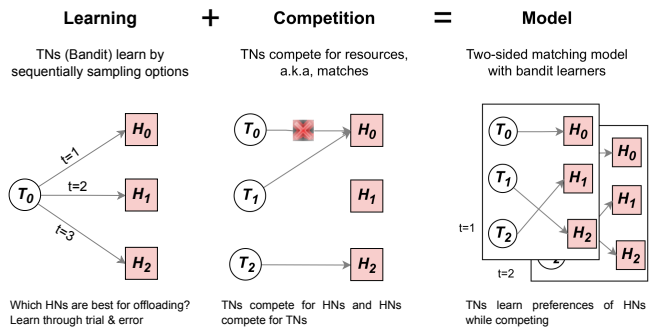

Fig. 2. Bandit learning based two-sided matching.

Algorithm 2: BLM algorithm

Input: Player set $\mathcal{T}$, arm set $\mathcal{H}$, parameter $\lambda \in (0,1)$.
Output: A stable matching $\mathcal{M} = \{(T_i(K), H_j(K))\}$ where $T_i(K) = \mathcal{M}(H_j(K))$

Initialization: $\forall\{T_i, H_j\} \in \{\mathcal{T}, \mathcal{H}\}$, random matching at $t = 0$: $\mathcal{M}(T_i(0)) = H_j(0)$, $a_{ij} = b_{ij} = 1$.
for $t = 1, 2, \cdots, K$ **do**
  for $r = 1, 2, \cdots, R$ **do**
    for $T_i \in \mathcal{T}$ **do**
      $\forall H_j \in \mathcal{H}$, sample $\theta_{ij}(t) \sim Beta(a_{ij}, b_{ij})$
      Independently draw $\pi_i(t) \sim Bernoulli(\lambda)$
      if $\pi_i(t) = 0$ **then**
        Construct potential matching set of T_i
        $P_i(t) := \{H_j : y_{ji}(t) \geq y_{ji'}(t)\}$ where $\mathcal{M}(T_{i'}(t-1)) = H_j$
        Pull $H_j(t) \in \arg\max_{H_j \in P_i(t)} \theta_{ij}(t)$
      else
        Pull $H_j(t) = \mathcal{M}(T_i(t-1))$
      if p_i *wins matching confict* **then**
        $H_j(t) = \mathcal{M}(T_i(t))$
        T_i received an instant reward $x_{ij}(t,r)$
        dervied from (7)
        Update $\overline{X}_{ij}(t)$ as (8)
        Draw $\omega(t) \sim Bernoulli(\overline{X}_{ij}(t))$
        Update $a_{ij}(t) \leftarrow a_{ij}(t) + \omega(t)$
        Update $b_{ij}(t) \leftarrow b_{ij}(t) + (1 - \omega(t))$
      else
        $\mathcal{M}(T_i(t)) = T_i(t)$
        T_i received a reward $x_{ij}(t,r) = 0$
        Update $\overline{X}_{ij}(t)$ as (8)

In round r of slot t, the algorithm samples an index $\theta_{ij}(r)$ from $Beta(a_{ij}, b_{ij})$ to represent the current estimation.

To avoid the frequent conflicts, each player constructs a potential matching set to exclude arms that already reject it in the previous round. We assume that the successful matched players are public at the end of each round. However, players can still simultaneously pull same arms for next round. To address this issues, we incorporate a random delay mechanism with hyper-parameter λ. Accordingly, each player first draws a Bernoulli random variable $\pi_i(t)$ with expectation λ. If $\pi_i(t) = 0$, T_i still attempts to pull the arms with the largest index in the potential set; otherwise, it follows the last-round choice.

When all players decide which arm to pull in this round, arms will determine which player to accept according o their rankings. If player T_i wins the conflict, it updates the average reward and the corresponding Beta distribution.

TABLE II
PARAMETERS FOR SIMULATION.

Parameters	Values
Network size $(2 \times M)$	10, 20
Task size, $L(t)$	U[1,15] KB
Maximum permitted latency, $D^{max}(t)$	$\{0.1, 0.2, 0.3, 0.4\}$ s
Data rate r_j	U[0.5,1] MBps
Processing density of FNs, $\gamma(t)$	U[500,1000] cycles/bit
CPU frequency of FNs, $f(t)$	$\{1.0, 1.5, 2.0, 2.5\}$ GHz

C. Complexity Analysis

The proposed BLM algorithm is based on iteration learning mechanism for obtain the optimal solution within each time slot, thus incurring an additional delay to reach the global optimum (i.e., stable matching). Regarding the computational complexity of the algorithm, in each iteration, the preference list of TNs and temporary fairing between TNs and HNs is updated and their stable matching is evaluated based on the DA algorithm. The number of rounds R and the network dimension $M * N$ determine the number of calculations required to update the TN-HN matching outcome. Given that $M = N$ in our proposition, the proposed algorithm has an overall complexity of order $O(R * M^2)$ per time slot.

V. Simulation Results and Evaluation Analysis

A. Simulation Environment Configuration

We evaluate the proposed algorithms in the fog environments where the network size $(2 \times M)$ includes 10 and 20 nodes. Table II summarizes the important parameters and values for the simulation scenario, where U[x,y] indicates the uniform distribution on interval [x, y]. In each scenario, we run all algorithms for $K = 1000$ time slots and all results averaged over 100 independent runs. Given the network configuration, we performed a large number of trial processes (10000 trials) to measure the response delays (offloading delays) offered by HNs. With the range of obtained offloading delays (0.087 to 0.45 s), we accordingly selected the maximum permitted latency for all tasks as in the set $\{0.1, 0.2, 0.3, 0.4\}$ (s) to evaluate the proposed algorithms.

B. Comparative Algorithms

Since the most of the existing solutions focus on bandit learning algorithm use ϵ-greedy and UCB, we compare our results with these two techniques used in the works [34]–[36]. Notably, the OTO algorithm in [36] use UCB based CMAB learning concept for selecting the efficient arms (i.e., HNs). In addition, the DTO-based matching solutions in the literature assume that the preference lists of two sides of matching game

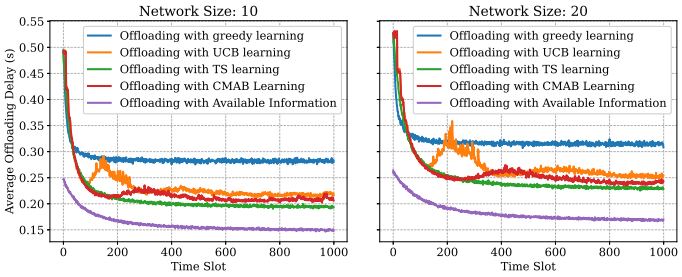

Fig. 3. The average delay offered by three different bandit learning based task offloading algorithms.

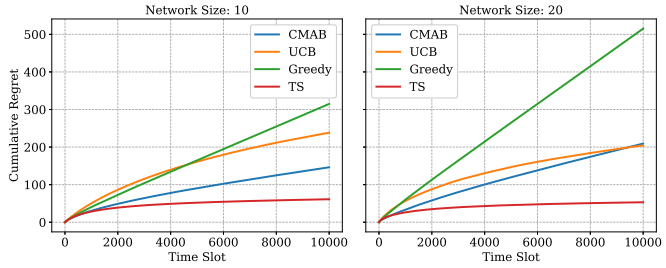

Fig. 4. Cumulative regrets obtained by different bandit learning policies used in the three task offloading solutions.

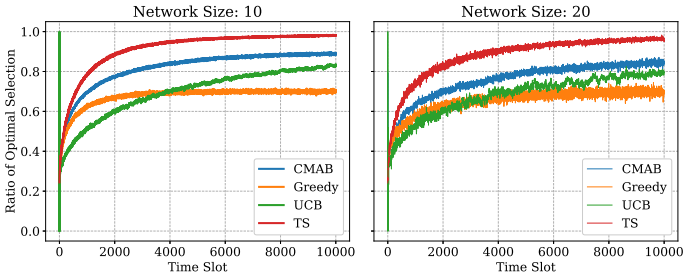

Fig. 5. Ratio of optimal selection.

are unknown, thus easily to produce the stable matching by the DA mechanism. We also compare our algorithms with the matching with available global information.

C. Evaluation Metrics

We evaluate the performance of proposed algorithms in three key metrics including average offloading delay, cumulative learning regret, ratio of optimal arm selection, and cumulative matching unstability.

D. Evaluation Analysis

Fig. 3 depicts the average offloading delay achieved by different bandit learning approaches (i.e., ϵ-greedy, UCB, CMAB, and proposed TS) over time slots. Notably, our TS-based BLM algorithm can achieve the sub-optimal result compared to the optimal solutions obtained when the information of network is available. This presents the efficiency of the proposed bandit learning using TS to deal with the dynamic environment of fog computing networks.

In addition, the TS-based learning technique presents its out-performance over ϵ-greedy and UCB due to the principle of posterior distribution estimation. Meanwhile, ϵ-greedy and UCB tend to assign the best HNs to tasks in a greedy manner. Notably, the CMAB learning-based algorithm performs a little worse than the proposed TS-based learning. When the network experiences a large number of time slots, the performance gap between the comparative algorithms are smaller because the network achieves more stable states. As more observations collected over time slots allow TN to select the optimal HNs efficiently. Similar observations regarding the performance of the comparative algorithms can also found when the network size increases. However, they take more time slots (over 200) to achieve the robust performance because the learning methods need to explore more to get the accurate estimations of network statues.

We also analyze the delay performance of proposed algorithms through learning regret LR metric, which is the expected difference between the delay achieved with the information availability and the delay achieved under the information unavailability. Mathematically, LR is defined as follow:

$$LR = \mathbb{E}\{\sum_{t=1}^{K}\sum_{i=1}^{M}\sum_{j=1}^{M}(\alpha_{ij}(t)D_{ij}(t) - \alpha_{ij^*}(t)D_{ij^*}(t))\}, \quad (9)$$

where $T_i = \mathcal{M}(H_j)$ under the matching with learning support and $T_i = \mathcal{M}(H_{j^*})$ under the matching without learning (i.e., information availability).

The out-performance of the proposed algorithm is enabled by TS-based learning technique. Fig. 4 shows the comparison of cumulative regret obtained by four algorithms.

The ϵ-greedy strategy uses a constant to balance exploration and exploitation. This is not ideal when the hyperparameter may be hard to tune. Also, the exploration is done in 100% randomly, such that we explore bandits equally (in average), irrespective of how promising they may look. The UCB strategy, unlike the ϵ-greedy, uses the uncertainty of the posterior distribution to select the appropriate bandit at each round. It supposes that a bandit can be as good as it's reward distribution. The UCB-based CMAB learning method is better than the pure epsilon-greedy and pure UCB learning since it can learn the reward distribution of multiple arms (HNs) simultaneously, thus having greater chance to estimate the optimal HNs. However, the CMAB learning converges slower than the TS-based learning because the action space includes a large number of arm combinations. Finally, TS uses a very elegant principle: to choose an action according to the probability of it being optimal. In practice, this makes for a very simple algorithm by taking one sample from each reward distribution, and choose the one with the highest value. Despite its simplicity, TS achieves state-of-the-art results, greatly outperforming the other algorithms. That is because TS promotes efficient exploration: it explores more where it is promising, and quickly discards bad actions. In addition, when more players pull multiple arms in the CMAB learning, there is a large probability of collisions. When the network size is large (20), the collision rate is higher, resulting in more explorations needed and thus the regret gaps between CMAB and TS learning are larger as shown in Fig. 4.

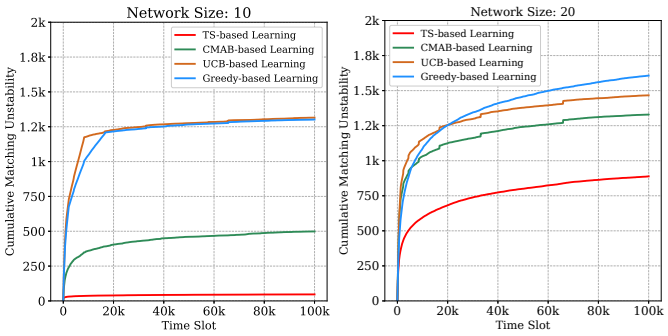

Fig. 6. Cumulative matching unstability with different network size.

Fig. 5 presents the probability of selecting the optimal HNs. The results show that the matching conflicts occurred during the resource competition have a severe impact on the ratio of optimal selection for the greedy and UCB-based learning algorithms. Meanwhile, our proposed algorithm using TS technique and especially the developed matching conflict avoidance mechanism can efficiently handle the issues. It naturally incorporates the uncertainty or stochasticity in the reward distribution by sampling from a probability distribution. This helps balance exploration and exploitation effectively. While the greedy method tends to be overly exploitative, always choosing the action with the highest estimated value, which may lead to suboptimal decisions if the estimates are inaccurate. Finally, UCB can be overly optimistic and might converge to suboptimal actions.

Fig. 6 reports the cumulative matching unstability, which is defined as the number of unstable matchings over time slots.

As shown in the simulation results, our TS-based learning algorithm shows the least matching unstability. Meanwhile, the greedy and UCB based learning solutions converge much slower and explore more to find a stable matching. The CMAB learning enables better performance than the greedy and UCB approach due to learning multiple arm distributions simultaneously. Particularly, the TS algorithm explicitly models uncertainty by sampling from a distribution of possible values for each arm. This allows it to adapt to changes in the environment and handle situations where the true reward distributions are not well-known. In the presence of uncertainty, the greedy and UCB based learning approaches rely on point estimates of the mean or upper confidence bound, respectively, which may lead to suboptimal decisions if the estimates are inaccurate.

VI. Conclusions

This paper introduces the TS-based bandit learning for distributed task offloading in the dynamic fog computing-based system. In principle, the algorithm applies the MAB learning empowered by Thompson sampling method to efficiently learn the uncertainty of fog computing environment, thus allowing TN to select the optimal HN for task offloading over time slots. Extensive simulation results demonstrate the the proposed algorithm outperform the benchmark algorithms using ϵ-greedy and UCB learning methods.

VII. Future Works

The superior performance of TS-based bandit learning demonstrated in the simulation results open many opportunities for future works. Firstly, it is worthy to investigate the proposed algorithms in a general setting of FCNs, in which the set of TNs and HNs has different number of nodes and the network is not fully connected. Secondly, several matching models would be notably investigated such as many-to-many (MTM) and many-to-one (MTO), in which multiple TNs can be matched with multiple HNs and single HN, respectively. Furthermore, other dimension of dynamic fog networks also need to be investigated such as the mobility of fog nodes as well as the volatile of HNs. The other research issue is to evaluate the performance of the proposed algorithm when the hard deadlines of tasks are concerned.

References

[1] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet Things J.*, vol. 1, pp. 22–32, 2014.

[2] D. A. Chekired, L. Khoukhi, and H. T. Mouftah, "Industrial IoT data scheduling based on hierarchical fog computing: A key for enabling smart factory," *IEEE Trans. Ind. Inf.*, vol. 14, pp. 4590–4602, Oct 2018.

[3] H. Tran-Dang, N. Krommenacker, P. Charpentier, and D.-S. Kim, "The internet of things for logistics: Perspectives, application review, and challenges," *IETE Technical Review*, pp. 1–29.

[4] H. Tran-Dang, N. Krommenacker, P. Charpentier, and D. Kim, "Toward the internet of things for physical internet: Perspectives and challenges," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 4711–4736, 2020.

[5] B. P. Rimal, E. Choi, and I. Lumb, "A taxonomy and survey of cloud computing systems," in *2009 Fifth International Joint Conference on INC, IMS and IDC*, 2009.

[6] A. Botta, W. de Donato, V. Persico, and A. Pescapé, "Integration of cloud computing and internet of things: A survey," *Future Generation Computer Systems*, vol. 56, pp. 684–700, 2016.

[7] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proc. ACM Proceedings of the first edition of the MCC workshop on Mobile cloud computing - MCC 2012*, 2012.

[8] S. Sarkar, S. Chatterjee, and S. Misra, "Assessment of the suitability of fog computing in the context of internet of things," *IEEE Transactions on Cloud Computing*, vol. 6, pp. 46–59, 2018.

[9] A. Yousefpour, G. Ishigaki, R. Gour, and J. P. Jue, "On reducing IoT service delay via fog offloading," *IEEE Internet Things J.*, vol. 5, pp. 998–1010, 2018.

[10] S. Sarkar and S. Misra, "Theoretical modelling of fog computing: a green computing paradigm to support iot applications," *IET Networks*, vol. 5, no. 2, pp. 23–29, 2016.

[11] M. Aazam, S. Zeadally, and K. A. Harras, "Offloading in fog computing for IoT: Review, enabling technologies, and research opportunities," *Future Generation Computer Systems*, vol. 87, pp. 278–289, 2018.

[12] Z. Liu, X. Yang, K. Wang, Y. Yang, and Z. Shao, "Computation offloading game for multi-task multi-helper fog networks," in *Proc. IEEE GLOBECOM*, 2019.

[13] K. H. Abdulkareem *et al.*, "A review of fog computing and machine learning: Concepts, applications, challenges, and open issues," *IEEE Access*, vol. 7, pp. 153123–153140, 2019.

[14] L. Liu *et al.*, "Multiobjective optimization for computation offloading in fog computing," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 283–294, 2018.

[15] G. Lee, W. Saad, and M. Bennis, "An online optimization framework for distributed fog network formation with minimal latency," *IEEE Trans. Wireless Commun.*, vol. 18, no. 4, pp. 2244–2258, 2019.

[16] K. Guo, M. Sheng, T. Q. S. Quek, and Z. Qiu, "Task offloading and scheduling in fog ran: A parallel communication and computation perspective," *IEEE Wireless Commun. Lett.*, vol. 9, no. 2, pp. 215–218, 2020.

[17] Y. Yang *et al.*, "Pomt: Paired offloading of multiple tasks in heterogeneous fog networks," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8658–8669, 2019.

[18] Z. Liu, Y. Yang, K. Wang, Z. Shao, and J. Zhang, "Post: Parallel offloading of splittable tasks in heterogeneous fog networks," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3170–3183, 2020.
[19] S. Durand and B. Gaujal, "Complexity and Optimality of the Best Response Algorithm in Random Potential Games," in *Proc. SAGT*, 2016.
[20] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 4, pp. 974–983, 2015.
[21] Z. Liu, X. Yang, Y. Yang, K. Wang, and G. Mao, "Dats: Dispersive stable task scheduling in heterogeneous fog networks," *IEEE Internet Things J.*, vol. 6, pp. 3423–3436, 2019.
[22] Tran-Dang, Hoa and Kim, Dong-Seong, "Disco: Distributed computation offloading framework for fog computing networks," *J. Commun. Netw.*, pp. 1–11, 2023.
[23] D. Gale and L. S. Shapley, "College admissions and the stability of marriage," vol. 69, no. 1, pp. 9–15, 1962.
[24] T. Lattimore and C. Szepesvári, *Bandit Algorithms*. Cambridge University Press.
[25] J. Vermorel and M. Mohri, "Multi-armed bandit algorithms and empirical evaluation," in *Proc. ECML*, 2005.
[26] W. R. Thompson, "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples," *Biometrika*, vol. 25, no. 3-4, pp. 285–294, 1933.
[27] T. L. Lai, H. Robbins, *et al.*, "Asymptotically efficient adaptive allocation rules," *Advances in applied mathematics*, vol. 6, no. 1, pp. 4–22, 1985.
[28] D. Russo and B. Van Roy, "Learning to optimize via posterior sampling," *Mathematics of Operations Research*, vol. 39, no. 4, pp. 1221–1243, 1976.
[29] O. Chapelle and L. Li, "An empirical evaluation of thompson sampling," *Proc. NeurIPS*, vol. 24, 2011.
[30] A. D. Flaxman, A. T. Kalai, and H. B. McMahan, "Online convex optimization in the bandit setting: Gradient descent without a gradient," in *Proc. ACM SODA*, 2005.
[31] W. Kumagai, "Introduction to bandit convex optimization algorithms," in *Proc. ISITA*, 2018.
[32] T. Chen and G. B. Giannakis, "Bandit convex optimization for scalable and dynamic IoT management," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 1276–1286, 2019.
[33] T. Chen and G. B. Giannakis, "Harnessing bandit online learning to low-latency fog computing," in *Proc. IEEE ICASSP*, 2018.
[34] S. Misra, S. P. Rachuri, P. K. Deb, and A. Mukherjee, "Multiarmed-bandit-based decentralized computation offloading in fog-enabled IoT," *IEEE Internet Things J.*, vol. 8, no. 12, pp. 10010–10017, 2021.
[35] Z. Zhu, T. Liu, Y. Yang, and X. Luo, "Blot: Bandit learning-based offloading of tasks in fog-enabled networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 12, pp. 2636–2649, 2019.
[36] K. Wang, Y. Tan, Z. Shao, S. Ci, and Y. Yang, "Learning-based task offloading for delay-sensitive applications in dynamic fog networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 11, pp. 11399–11403, 2019.
[37] W. Chen, Y. Wang, and Y. Yuan, "Combinatorial multi-armed bandit: General framework and applications," in *Proceedings of the 30th International Conference on Machine Learning* (S. Dasgupta and D. McAllester, eds.), vol. 28, pp. 151–159, 2013.
[38] B. Cho and Y. Xiao, "Learning-based decentralized offloading decision making in an adversarial environment," *IEEE Trans. Veh. Technol.*, vol. 70, no. 11, pp. 11308–11323, 2021.
[39] H. Zhu, K. Kang, X. Luo, and H. Qian, "Online learning for computation peer offloading with semi-bandit feedback," in *Proc. IEEE ICASSP*, 2019.
[40] Y. Sun, S. Zhou, and Z. Niu, "Distributed task replication for vehicular edge computing: Performance analysis and learning-based algorithm," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 1138–1151, 2020.
[41] B. Shabir *et al.*, "A reliable learning based task offloading framework for vehicular edge computing," in *Proc. ICoDT2*, 2022. j
[42] Y. Shan, H. Wang, Z. Cao, and K. Yury, "Data offloading in heterogeneous dynamic fog computing network: A contextual bandit approach," in *Proc. ICCCI*, 2021.
[43] H. Liao, Z. Zhou, X. Zhao, B. Ai, and S. Mumtaz, "Task offloading for vehicular fog computing under information uncertainty: A matching-learning approach," in *Proc. IWCMC*, 2019.

Hoa Tran-Dang (M'17) received the B.E. degree in Electrical and Electronics Engineering from Hanoi University of Science and Technology (HUST), Vietnam and the M.S. degree in Electronics Engineering from Kumoh National Institute of Technology (KIT), South of Korea in 2010 and 2012, respectively. He pursued the Ph.D. degree with University of Lorraine, France during 2013-2017. He currently works in th department of IT Convergence Engineering at Kumoh National Institute of Technology, South of Korea as a Research Professor. His research interests include Internet of things (IoT), edge/fog computing, distributed intelligent computing, physical Internet, and AI.

Dong-Seong Kim (S'98-M'03-SM'14) received his Ph.D. degree in Electrical and Computer Engineering from the Seoul National University, Seoul, Korea, in 2003. From 1994 to 2003, he worked as a Full-Time Researcher in ERC-ACI at Seoul National University, Seoul, Korea. From March 2003 to February 2005, he worked as a Postdoctoral Researcher at the Wireless Network Laboratory in the School of Electrical and Computer Engineering at Cornell University, NY. From 2007 to 2009, he was a Visiting Professor with Department of Computer Science, University of California, Davis, CA. He is currently a Director of KIT Convergence Research Institute and ICT Convergence Research Center (ITRC program) supported by Korean government at Kumoh National Institute of Technology. He is IEEE and ACM Senior Member. His current main research interests are real-time IoT, industrial wireless control network, networked embedded system, and Fieldbus.