An Efficient Bandit Learning based Online Task Offloading in Fog Computing-enabled Systems

Hoa Tran-Dang
Dept. of IT convergence engineering
Kumoh National Institute of Technology
Gumi-si, South of Korea
{hoa.tran-dang}@kumoh.ac.kr

Dong-Seong Kim
Dept. of IT convergence engineering
Kumoh National Institute of Technology
Gumi-si, South of Korea
dskim@kumoh.ac.kr

IECON 2023- 49th Annual Conference of the IEEE Industrial Electronics Society | 979-8-3503-3182-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/IECON51785.2023.10311831

***Abstract*—Fog computing technology has developed to support delay-sensitive applications by offering shared and adaptable communication, computation, and storage resources along the cloud-to-things continuum in Internet of Things (IoT) and cyber-physical systems (CPS). In order to minimize the delay of every task, task nodes (TNs) with computation-intensive and delay-sensitive tasks should have efficient strategies to select the optimal helper nodes (HNs) having spare computation resources for task offloading operations. However, the dynamic nature of fog computing environment characterized by the time varying change of HN computing resources as well as various types of tasks with different quality of service (QoS requirements) impose as inherent challenges for designing the efficient task offloading algorithms. To deal with these challenges, we apply the principle of multi-armed bandit (MAB) learning method to efficiently learn the uncertainty of fog computing environment. In particularly, we use Thomson sampling (TS) technique to acquire better exploitation and exploration trade-off, allowing TNs to select their corresponding HNs efficiently. Extensive simulation results demonstrate the potential advantages of the TS-type algorithm over the ϵ-greedy and UCB based offloading algorithms.**

***Index Terms*—Fog computing network, multi-armed bandit, exploitation and exploration dilemma, Thompson sampling, Task offloading.**

I. Introduction

Fog computing has been introduced and integrated widely in the practical IoT and cyber-physical systems (CPS). As an extension of cloud computing, fog computing platforms placed between the cloud layer and user equipment (UE) layer in the systems also provide the cloud-like services (i.e., IaaS, PaaS, SaaS) to the UEs. In this context, devices with integrated fog computing platforms called fog nodes (FNs) can process and offload most tasks requested by the UEs on behalf of the cloud servers [1], thereby allowing the systems to achieve the improved performances in terms of service delay, energy saving, and service cost [2].

However, to realize these benefits of fog computing paradigm, there requires efficient task offloading operations to overcome inherent challenges in the fog computing environment such as the heterogeneity of fog computing devices, various types of computational tasks with different quality of service (QoS) and quality of experience (QoE) requirements [3].

There are a large number of centralized optimization techniques and algorithms proposed in the literature to provide optimal offloading solutions [4]. These approaches require a centralized control to gather the global system information, thus incurring a significant overhead and computation complexity of algorithms especially in case of density and complicated heterogeneity of fog computing environment [5].

The aforementioned limitations of optimization have lead to a second class of game theory based offloading solutions that can avoid the cost-intensive centralized resource management as well as substantially reduce the complexity of algorithms [6]. However, classical game theoretical algorithms such as best response require some information regarding actions of other players [7]. Correspondingly, many assumptions are introduced in the game theory based algorithms to simplify the system models that, in some case, are impractical.

Recently, matching theory has emerged as a promising technique applied to derive distributed task offloading algorithms [8] that significantly can reduce the service latency in the fog-based systems. More importantly, the matching-based approaches have potential advantages over the optimization and game theory based solution owing to the distributed and low computational complexity algorithm.

However, most of aforementioned solutions assume the information regarding the resource states of fog computing nodes are known $a\ priori$, which is not realistic in many practical applications. For example, the resource demand-side FNs called task nodes (TNs) are likely to be uncertain about the computing capability (i.e., CPU frequency, queuing delay) of resource supply-side FNs (i.e., helper nodes (HNs)) at time of offloading requesting since it is varying over time. Therefore, to efficiently offload the tasks in an online manner, the TNs must interact iteratively with the HNs to learn their unknown computing resource status.

Multi-armed bandit (MAB) is a common approach to modeling this type of learning process, which aims to solve the exploitation and exploration dilemma. 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 [9]. 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 [10], [11], [12], 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 [13], [14], [15]. Unlike TS, performance of this type of policies heavily rely on the confidence sets used to compute the exploration bonus [12]. This together with the superior performance of TS evaluated in numerous applications [16], [17] motivate us to investigate a TS based learning for our problem.

In these regards, this paper aims to develop an online task offloading algorithm abbreviated as TS-BLOTO for the dynamic fog computing-enabled systems based on MAB learning, which particularly use the TS technique to efficiently learn the uncertainty of fog computing environment.

II. Related Works

Task offloading has recently gained popularity as a potential approach to efficiently use distributed computing resources, and several studies conducted in this area have proposed efficient offloading solutions using different methodologies.

The work presented in [18] developed a code offloading architecture with on-demand computing resource allocation and concurrent task execution. The task offloading was represented as a deterministic optimization problem in [19], [20]. To enhance the total energy efficiency in homogeneous fog networks, the authors in [21] specifically made the assumption that they had complete knowledge of the system status. The offloading decision of which node to offload optimally becomes an integer programming problem and is challenging to solve under the supposition that the tasks could not be divided arbitrarily [22]. Real-time information collection and response are essential for making intelligent offloading decision in a fog-enabled network [23]. Using the real-time statuses of the users and the servers, such as the sizes of the compute queues, one efficient task offloading technique should be able to quickly adapt to the demanding dynamics in the environments. As a result, task offloading is frequently a stochastic programming issue, and the above-mentioned traditional optimization techniques with deterministic parameters are no longer appropriate. Utilizing the Lyapunov optimization approach like in [24], [25] is one option to resolve this conundrum. Additionally, a game-theoretic decentralized strategy was offered in [26] to provide independent offloading decisions from each user. Accordingly, the task offloading is structured as a game and followed the Nash equilibrium rather than attempting to solve a challenging integer programming issue.

All of the computation offloading strategies listed above presupposed complete knowledge of the system characteristics. In practical, these factors are sometimes unknown or just partially known to the user. As an example, some values (also known as bandit feedbacks) are only disclosed for the nodes that are queried. The authors in [27] specifically considered the calculation and communication delays of each task as a posteriori. Each device's movement was thought to be unpredictable in [28]. Offloading feedbacks are used in the reinforcement learning-based offloading methods, such those in [29], [30], to learn characteristics like latency and energy cost. There will be a tradeoff between using the empirically best node as frequently as possible and researching other nodes to uncover more lucrative activities when the number of nodes that can be queried is constrained owing to the limiting amount of resources that are available. To tackle this trade-off, the ϵ-greedy approach, which is commonly used in reinforcement learning [29], [30], [31], should be used. This strategy, nevertheless, converges slowly and is non-optimal [15].

The authors in [32] introduced D2CIT - a two-tier distributed strategy for offloading computing in an IoT context with fog. They took into account SNs with time-sensitive tasks needed to be computed within the predefined deadlines. The high-level tasks will be divided into more manageable subtasks by D2CIT, which will then create a DATG. For the FN selection, they provided a greedy solution that separates the job from the appropriate SNs. In a dynamic setting, they develop an ϵ-greedy nonstationary MAB-based strategy for automated subtask redistribution.Additionally, they contrasted D2CIT with already existing solutions and demonstrated how the suggested algorithm performs better in terms of latency and speedup.

The study [33] examined a performance guarantee for an effective online task offloading approach in a fog-enabled network. A stochastic programming with delayed bandit feedbacks was defined and formulated since the expectations for processing speeds vary rapidly at unknown time instants and the system information is only accessible after completing the related tasks. BLOT, an effective online task offloading method based on the UCB policy, is developed to address this issue. The simulation results show that the proposed BLOT algorithm is capable of learning and selecting the appropriate node to offload tasks in an online manner under non-stationary conditions.

III. System Model

A. Fog Computing Network

For typical applications as introduced in [34], [33], we study a FCN as illustrated in Fig. 1 including a TN and M HNs. Denoting the set of HNs as $\mathcal{H} = \{H_1, \ldots, H_j, \ldots, H_M\}$. FNs are heterogeneous in terms of computing capability, storage, and connectivity technology. We assume that TN can connect to every HNs 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 $\mathcal{S} = \{1, \ldots, t, \ldots, K\}$. At the t^{th} slot, TN generates a corresponding task $T(t)$, which is represented by a tuple

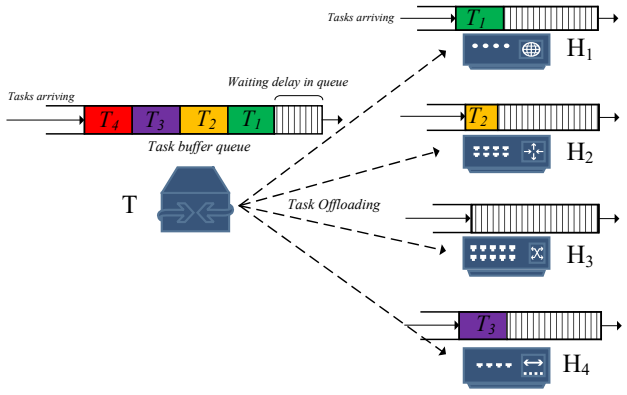

Fig. 1: An illustrative model of fog computing enabled systems with a TN and multiple HNs.

$T(t) = \{L(t), \Gamma(t), D^{max}(t)\}$, where $L(t)$ is the task size (bits), $\Gamma(t)$ is the computation complexity of task (CPU cycles/bit), and $D^{max}(t)$ is the maximum permitted latency for the task T 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 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 the quality of service (QoS) and the quality of experience (QoE). To enable low-latency processing, one FN, i.e., task node (TN), may offload some of its computation burdens to the nearby helper nodes (HNs). These HNs typically posses more computation and storage resources and are deployed to help other TNs on demand. 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 across different time slots.

B. Problem Formulation

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

$$D_j(t) = \delta_j^{tx}(t) + \delta_j^{w}(t) + \delta_j^{p}(t), \tag{1}$$

where $\delta_j^{tx}(t)$ is the transmission delay, $\delta_j^{w}(t)$ is the waiting delay in queue of $H_j(t)$, and $\delta_j^{p}(t)$ is the processing delay by $H_j(t)$. Given the data rate $R_j(t)$ (bits/s) between TN and H_j at time slot t, $\delta_j^{tx}(t) = L(t)/R_j(t)$. The processing delay is derived as:

$$\delta_j^{p}(t) = \frac{L(t)\Gamma(t)}{f_j(t)}. \tag{2}$$

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

$$\overline{D} = \lim_{K\to\infty} \frac{1}{K} \sum_{t=1}^{K} \sum_{j=1}^{M} \alpha_j(t) D_j(t). \tag{4}$$

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

$$\begin{aligned} \mathbf{P:} \quad & \min_{\alpha_j(t)} \overline{D} \\ \text{s. t.} \quad & C_1 : \alpha_j(t) \in \{0, 1\}, \forall \{j, t\} \in \{\mathcal{H}, \mathcal{S}\}, \\ & C_2 : \sum_{j=1}^{\mathcal{H}} \alpha_j(t) = 1, \forall t \in \mathcal{S}. \end{aligned} \tag{5}$$

Here, the constraints C_1 and C_2 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 $D_j(t)$ is not available before the task $T(t)$ is completed. In addition, event if $D_j(t)$ is estimated a priori, this problem is still a combinatorial optimization problem and the complexity is in the order of $\mathcal{O}(K^T)$. 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. TS-BLOTO Algorithm Design

This section introduces the TS-BLOTO algorithm, which allows each FN to offload the tasks in an online and distributed manner using TS-based bandit learning method. Basically, the task offloading in the studied system can be viewed as a MAB problem, in which TN and HNs are served as player, and arms, respectively. In addition, the reward when player pull an arm H_j is defined as follow: For the objective of delay minimization, the preference value takes into account the capability of HNs to processing the tasks and is calculated as follow:

$$x_{ij} = \frac{1}{1 + e^{-(D_i^{max}(t) - D_{ij}(t))}}. \tag{6}$$

At each time slot $s = 1, 2, ..., K$, each player T_i attempts to pull an arm H_j, it will receive a random reward $x_{ij}(t)$ with expectation $\overline{X}_{ij}$. Over time slots, $\overline{X}_{ij}$ is estimated as follow:

$$\overline{X}_{ij}(t) = \frac{\sum_{s=0}^{t} \gamma^t x_{ij}(s)}{\sum_{s=0}^{t} \gamma^s}, \tag{7}$$

where $\gamma \in (0,1)$ serves as the discount factor. The aim is to maximum the cumulative rewards, thus minimizing the average service delay.

Algorithm 1 presents the procedures to implement TS-BLOTO.

Algorithm 1: TS-BLOTO Algorithm

Input: Player TN, arm set $\mathcal{H}$, parameter $\lambda \in (0,1)$.
Output: Optimal arm pulled in each time slot $\mathcal{H}^* = \{H_i(1), ..., H_j(t), ..., H_k(K)\}$, $\{H_i, H_j, H_k\} \in \mathcal{H}$.

Initialization: $a_j = b_j = 1$, $\forall H_j \in \mathcal{H}$
for $t = 1, 2, ..., K$ **do**
  $\forall H_j \in \mathcal{H}$, sample $\theta_j(t) \sim Beta(a_j, b_j)$
  Pull $H_j(t) \in \arg\max_{H_j \in \mathcal{H}} \theta_j(t)$
  Insert $H_j(t)$ into $\mathcal{H}^*$
  TN receives a reward $x_j(t)$ dervied as (6)
  Update $\overline{X}_j(t)$ as Equation (7)
  Draw $\omega_j(t) \sim Bernoulli(\overline{X}_j(t))$
  Update $a_j(t) \leftarrow a_j(t) + \omega_j(t)$
  Update $b_j(t) \leftarrow b_j(t) + (1 - \omega_j(t))$

V. Simulation Results and Evaluation Analysis

A. Simulation Environment Configuration

The event-driven framework supported SimPy library in Python is used to conduct the simulation scenarios, which investigate the performance of algorithms. The fog computing network consists of one TN and nine HNs as evaluated in the related work [33]. Table I summarizes the important parameters and values for the simulation scenario, where U[x,y] indicates the uniform distribution on interval [x, y].

TABLE I: Parameters for simulation

Parameters	Values
Task size, $L(t)$	U[1,15](KB)
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)

B. Comparative Analysis

Fig. 2 depicts the average delay achieved by D2CIT, BLOT, and TS-BLOTO over time slots.

Due to the different bandit learning methods used in the algorithms, the average delay offered by these solutions are different over time slots. 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. In addition, more observations collected over time slots allow TN to select the optimal HNs efficiently. Notably, TS-BLOTO presents its out-performance over D2CIT, BLOT due to the principle of Thompson sampling. Meanwhile, D2CIT and BLOT assign the best HNs to tasks in a greedy manner.

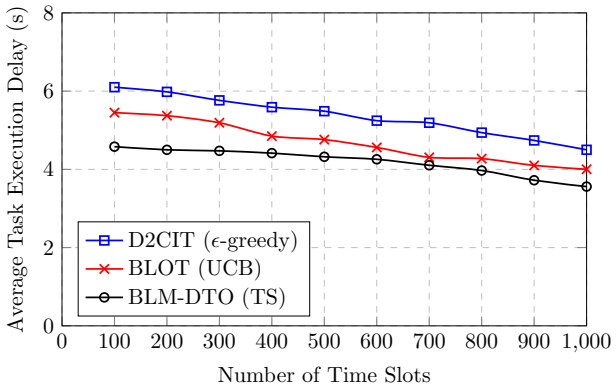

Fig. 2: The average delay offered by D2CIT, BLOT, and TS-BLOTO algorithms.

The outperformance of the proposed algorithm is enabled by TS-based learning technique. Fig. 3 shows the comparison of commutative regret obtained by three algorithms.

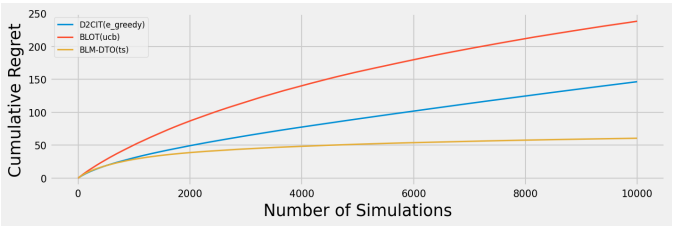

Fig. 3: Commutative regrets obtained by different bandit learning policies used in the three task offloading solutions

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 UCB. 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.

VI. Conclusions

This paper introduces TS-BLOTO, an algorithms for online task offloading in the dynamic fog computing-based system. In principle, TS-BLOTO 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.

With the proved advantage of TS technique for MAB learning, there are many directions for extending this current work in the future works. For example, a larger network size can be investigated to examine the convergence rate of algorithms. In addition, multi-player MAB problem is potential to be considered when multiple players (i.e., multiple TNs) pull a same arm and this incurs the conflict. Resolving this kind of conflict can involve the matching theory and its stability principle.

ACKNOWLEDGMENTS

This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the Grand Information Technology Research Center support program(IITP-2023-2020-0-01612) supervised by the IITP(Institute for Information & communications Technology Planning & Evaluation)", and Korea Research Fellowship Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (RS-2023-00249687).

REFERENCES

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

[2] H. Tran-Dang and D.-S. Kim, "FRATO: Fog resource based adaptive task offloading for delay-minimizing IoT service provisioning," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 10, pp. 2491–2508.

[3] 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, Oct. 2018.

[4] L. Liu, Z. Chang, X. Guo, S. Mao, and T. Ristaniemi, "Multiobjective optimization for computation offloading in fog computing," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 283–294, 2018.

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

[6] Y. Yang, Z. Liu, X. Yang, K. Wang, X. Hong, and X. Ge, "Pomt: Paired offloading of multiple tasks in heterogeneous fog networks," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8658–8669, 2019.

[7] S. Durand and B. Gaujal, "Complexity and Optimality of the Best Response Algorithm in Random Potential Games," in *Symposium on Algorithmic Game Theory (SAGT) 2016*, pp. 40–51.

[8] H. Tran-Dang and D.-S. Kim, "A survey on matching theory for distributed computation offloading in iot-fog-cloud systems: Perspectives and open issues," *IEEE Access*, vol. 10, pp. 118 353–118 369.

[9] J. Vermorel and M. Mohri, "Multi-armed bandit algorithms and empirical evaluation," in *Machine Learning: ECML 2005: 16th European Conference on Machine Learning, Porto, Portugal, October 3-7, 2005. Proceedings 16*. Springer, pp. 437–448.

[10] 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.

[11] S. Agrawal and N. Goyal, "Analysis of thompson sampling for the multi-armed bandit problem," in *Conference on learning theory*. JMLR Workshop and Conference Proceedings, pp. 39–1.

[12] D. Russo and B. Van Roy, "Learning to optimize via posterior sampling," *Mathematics of Operations Research*, vol. 39, no. 4, pp. 1221–1243.

[13] T. L. Lai, H. Robbins *et al.*, "Asymptotically efficient adaptive allocation rules," *Advances in applied mathematics*, vol. 6, no. 1, pp. 4–22.

[14] R. Agrawal, "Sample mean based index policies by o (log n) regret for the multi-armed bandit problem," *Advances in Applied Probability*, vol. 27, no. 4, pp. 1054–1078.

[15] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine learning*, vol. 47, no. 2, pp. 235–256.

[16] O. Chapelle and L. Li, "An empirical evaluation of thompson sampling," *Advances in neural information processing systems*, vol. 24.

[17] S. L. Scott, "A modern bayesian look at the multi-armed bandit," *Applied Stochastic Models in Business and Industry*, vol. 26, no. 6, pp. 639–658.

[18] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, "Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading," in *2012 Proceedings IEEE INFOCOM*, pp. 945–953.

[19] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Transactions on Wireless Communications*, vol. 16, no. 3, pp. 1397–1411.

[20] F. Wang, J. Xu, X. Wang, and S. Cui, "Joint offloading and computing optimization in wireless powered mobile-edge computing systems," *IEEE Transactions on Wireless Communications*, vol. 17, no. 3, pp. 1784–1797.

[21] Y. Yang, K. Wang, G. Zhang, X. Chen, X. Luo, and M.-T. Zhou, "Meets: Maximal energy efficient task scheduling in homogeneous fog networks," *IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 4076–4087.

[22] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. S. Quek, "Offloading in mobile edge computing: Task allocation and computational frequency scaling," *IEEE Transactions on Communications*, vol. 65, no. 8, pp. 3571–3584.

[23] P. Yang, N. Zhang, Y. Bi, L. Yu, and X. S. Shen, "Catalyzing cloud-fog interoperation in 5g wireless networks: An sdn approach," *IEEE Network*, vol. 31, no. 5, pp. 14–20.

[24] Y. Yang, S. Zhao, W. Zhang, Y. Chen, X. Luo, and J. Wang, "Debts: Delay energy balanced task scheduling in homogeneous fog networks," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 2094–2106.

[25] L. Pu, X. Chen, J. Xu, and X. Fu, "D2d fogging: An energy-efficient and incentive-aware task offloading framework via network-assisted d2d collaboration," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3887–3901.

[26] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 4, pp. 974–983.

[27] T. Chen and G. B. Giannakis, "Bandit convex optimization for scalable and dynamic iot management," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 1276–1286.

[28] C. Tekin and M. van der Schaar, "An experts learning approach to mobile service offloading," in *2014 52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 643–650.

[29] M. Min, X. Wan, L. Xiao, Y. Chen, M. Xia, D. Wu, and H. Dai, "Learning-based privacy-aware offloading for healthcare iot with energy harvesting," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4307–4316.

[30] M. Min, L. Xiao, Y. Chen, P. Cheng, D. Wu, and W. Zhuang, "Learning-based computation offloading for iot devices with energy harvesting," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1930–1941.

[31] R. Sutton and A. Barto, "Reinforcement learning: An introduction," *IEEE Transactions on Neural Networks*, vol. 9, no. 5, pp. 1054–1054.

[32] S. Misra, S. P. Rachuri, P. K. Deb, and A. Mukherjee, "Multiarmed-bandit-based decentralized computation offloading in fog-enabled iot," *IEEE Internet of Things Journal*, vol. 8, no. 12, pp. 10 010–10 017.

[33] Z. Zhu, T. Liu, Y. Yang, and X. Luo, "Blot: Bandit learning-based offloading of tasks in fog-enabled networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 12, pp. 2636–2649.

[34] Y. Yang, "Multi-tier computing networks for intelligent IoT," *Nature Electronics*, vol. 2, no. 1, pp. 4–5.