Delay Optimization in Mobile Edge Computing Federation using Task Offloading and Resource Allocation

Huong Mai Do
Department of Information Communication Convergence
Soongsil University
Seoul, Korea Republic of.
huongdm@soongsil.ac.kr

Myungsik Yoo
School of Electronic Engineering
Soongsil University
Seoul, Korea Republic of.
myoo@ssu.ac.kr

2022 13th International Conference on Information and Communication Technology Convergence (ICTC) | 978-1-6654-9939-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICTC55196.2022.9952817

***Abstract*—In recent years, computation offloading has become an effective solution for limited resource issues. This article investigates decision offloading and computing resource allocation. We propose a task offloading framework to minimize the total delay considering the resource condition in the MEC federation system. Then, we also solve this problem by transferring to Markov Decision Process (MDP) and using the Deep Deterministic Policy Gradient (DDPG) framework. Simulation results show that the DDPG can successfully lower the delay of the tasks.**

***Index Terms*—MEC federation, task offloading, resource allocation, Markov decision process, deep reinforcement learning (DRL).**

I. Introduction

Mobile Edge Computing (MEC) is a solution for the issue of limited computational, battery, and storage capability of user devices. A rising tendency is to offload or shift computation-intensive operations to potent distant computing platforms. So many studies focus on the optimization problem in MEC as minimizing the latency and energy cost [1] [2] [4] or maximizing the offload tasks [3], but their problems are usually solved in MEC single while the technology is gradually moving towards the MEC federation, the migration between MEC servers probably can help improve efficiently of MEC system.

In this study, we develop a resource allocation model by proposing an efficient task offloading framework. First, we propose the MEC federation system model that can migrate tasks between MEC servers. Second, we focus on delay and formulate an optimization problem with the constraints of resources. Third, we transfer this problem to MDP and solve it by using DDPG framework. Finally, we conduct experiments to demonstrate the better performance of the DDPG framework by comparing it with uniform resource management (URM) and random resource management (RRM) algorithms.

II. System model

The MEC federation system model is shown in Fig. 1. There are multiple user devices (UDs) N denoted by $\{1, 2, ..., N\}$, assuming that the devices are IoT equipment in the industry. The MEC federation includes multiple MEC servers, each

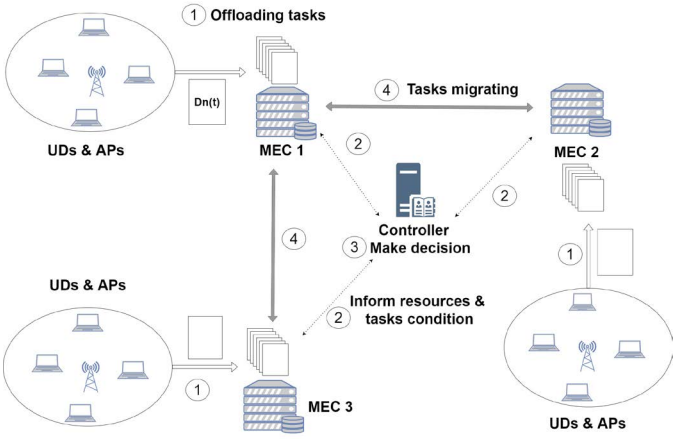

Fig. 1. Proposed system architecture

MEC server can communicate with user devices by the base stations (BSs) [1] [4]. All information of the resources and tasks status will be informed to the controller to make the decision that the tasks will be executed by the local device or offloaded to the server [4]. If the MEC server can not handle the offloaded tasks and has the remaining tasks, the remaining need to be migrated and executed by another MEC server or take the time for waiting in this MEC server.

A. Task model

We adopt a time-slotted model denoted $\{1, 2, ..., T\}$. At time slot t, each UD generates a task, where the task data size is denoted by $D_n(t)$ with $n \in \{1, 2, ..., N\}$. We also suppose that a UD can only request one task per time slot and tasks can not be partitioned into sub-tasks to be executed in multiple places at the same time.

B. Communication model

We consider 4 kinds of transfer links, up/down-link between the user device and edge server and between two edge servers

[5]. The data transfer rate of all links is obtained by the Shannon formula as follows:

$$R_n(t) = B\log_2\big(1 + \frac{P_n^T(t)G_n(t)}{BN_0}\big), \quad (1)$$

where B denotes the channel bandwidth, $P_n^T(t)$ denotes the transmission power of the user device or the MEC server to send out the task or the result, $G_n(t)$ denotes the wireless channel gain between two MEC servers or the user device and the MEC server, and $N_0 = -174\ dBm$ is Gaussian noise power spectrum density.

When data is transferred, a delay will be incurred called transmission delay:

$$T_n^d(t) = \frac{D_n(t)}{R_n(t)}. \quad (2)$$

Each UD should have enough energy to support the offloading operation of the tasks. We call it the transmission energy, calculated as follows:

$$E_n^T(t) = T_n^d(t)P_n(t). \quad (3)$$

C. Computation model

The system has three computing modes:

- Execution by the local user device ($X_{ij} = 0$)
  Assume $1bit$ of task $D_n(t)$ needs L^U CPU cycles of the user device, so the computing resource allocate for task $D_n(t)$ is $f_n^U(t)$ [2].
  The delay is the computing time of the user device:

$$T_n^U(t) = \frac{L^U D_n(t)}{f_n^U(t)}. \quad (4)$$

- Offloading to only one edge server ($X_{ij} = 1$)
  When the tasks are offloaded to the edge server, they need to be stored in the buffer and wait for previous tasks completed [6]. The queue delay is as follows:

$$T_n^{E-queue}(t) = \frac{L^E}{f_n^E(t)}(S_{capacity}^E - S_{available}^E), \quad (5)$$

  with S is the storage space of the buffer.
  The delay includes transmission delay $T_n^{U-T}(t)$, $T_n^{E-T}(t)$, computing delay $T_n^E(t)$, and queue delay $T_n^{E-queue}(t)$) as follows:

$$T_n^E(t) = T_n^{U-T}(t) + T_n^E(t) + T_n^{E-T}(t) + T_n^{E-queue}(t). \quad (6)$$

- Migration of remaining tasks to another server ($X_{ij} = 2$)
  The delays include computing delay $T_{n'}^E(t)$, queue delay, and communication delay (both between the user device and edge server and between two edge servers $T_{n^r}^{M-T}(t)$, as follows:

$$\begin{aligned} T_n^M(t) = {} & T_n^{U-T}(t) + T_n^{E-queue}(t) + T_{n^r}^{M-T}(t) \\ & + T_{n'}^E(t) + T_{n^r}^{E'-queue}(t) + T_{n^r}^{E'}(t) + T_n^{E-T}(t). \end{aligned} \quad (7)$$

D. Problem formulation

The objective of this study is to minimize the delay of the MEC federation in consideration of the resources. The objective can be expressed below:

$$\begin{aligned} \min_{X_{ij}} T = \min_{X_{ij}} \sum_{t=1}^{T}\sum_{n=1}^{N} \frac{1}{2NT}[& T_n^U(t)(1-X_{ij})(2-X_{ij}) \\ & + T_n^E(t)X_{ij}(2-X_{ij})(3-X_{ij}) \\ & + T_n^M(t)X_{ij}(1-X_{ij})(3-X_{ij})]. \end{aligned} \quad (8)$$

We also present five constraints as below:

$$C1: \quad X_{ij} \in \{0,1,2\}, \quad (9)$$

$$C2: \quad 0 \le \sum_j X_{ij} \le 2\ , \quad \prod_j X_{ij} \ne 1, \quad (10)$$

$$C3: \quad p_{min}^i(t) \le p_n^i(t) \le p_{max}^i(t), \quad (11)$$

$$C4: \quad F_{available}^j(t) \ge \sum_{n=1}^{N} f_n^j(t)\ , \quad f_n^j(t) \ge 0, \quad (12)$$

$$C5: \quad E_{available}^j(t) \ge \sum_{n=1}^{N} E_n^j(t). \quad (13)$$

Constraint 1 guarantees that the offloading decision only can select one in 3 options. Constraint 2 means a server can choose to migrate to only one place in time t. Constraint 3 presents that the transmission power should not be out of range. Constraint 4 means the computing resource allocated is a positive value and the total computing resource allocated for all tasks should not be over the resource available at this time. Constraint 5 shows the total energy consumption of the device should not be over the energy available at time t [3].

III. Methodology

In this section, we transform our problem (8) into an MDP problem and then use the DDPG framework for solving it.

A. MDP-based resource allocation problem

An MDP consists of a 5-tuple $M = \{S, A, P, R, \gamma\}$, with $\gamma \in [0, 1]$.

- State space:

$$S(t) = \{G(t), D(t), E(t), F(t)\}. \quad (14)$$

- Action space:

$$A(t) = \{p(t), f(t), X(t)\}. \quad (15)$$

- State transition probability: The state transition probability $P(s(t+1)|(s(t), a(t))$ indicates the probability of $s(t+1)$ given $s(t)$ and selected $a(t)$.
- Reward function: The reward function $R(t)$ presents the immediate reward when selecting $A(t)$ in $S(t)$. The MDP solves the optimization problem by maximizing the reward, but our problem is to minimize the delay. So, we can maximize the negative of the objective function.

$$R(t) = \frac{-1}{N}\sum_{n=1}^{N} T_n(t). \quad (16)$$

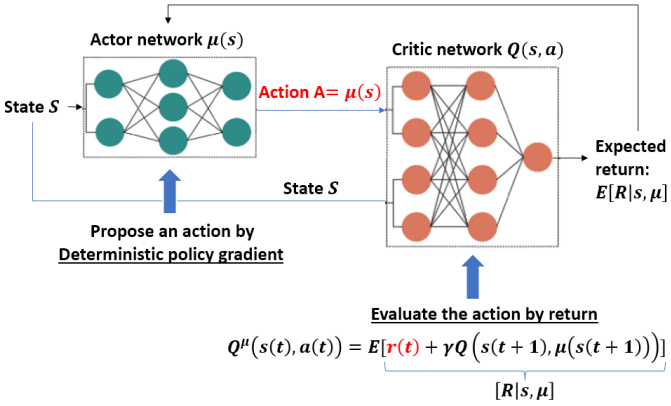

Fig. 2. DDPG framework

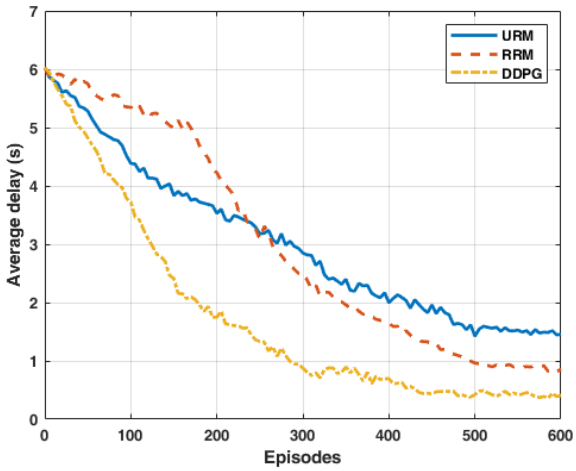

Fig. 3. Impact of different methods to delay

B. DRL-based resource allocation framework

In this article, we adopt the DDPG framework, which is one of the categories of the Actor-Critic-based DRL, the processing includes 3 phases:

- Update critic network then get Q-value: minimizing loss function between Q-value target and Q-value predicted.
- Update actor network then optimize the policy $\mu(\theta^{\mu})$: maximizing the performance objective function by using Determined strategy policy gradient.
- Update target network.

IV. Experiments and Results

We conduct a series of simulations to evaluate the performance of the DDPG scheme for resource allocation.

A. Experimental Setup

We give tasks size, the number of users, and the number of MEC servers as input, the simulation output will be efficient in offloading decisions and delay optimization with resource allocation. Besides, all of the details of the setting parameter are shown in Table 1.

TABLE I
Experiments Parameters

Parameters	Value
Number of UDs	5
Number of edge servers	2
CPU cycles per bit of edge server	735
CPU cycles per bit of device	600
Tasks size	[1, 10] GB
Bandwidth between device-MEC (up/down)	100M
Bandwidth between 2 MECs	150M
Power range	[5, 38] dBm
Computing resource capacity of device/edge	5/25 GHz
Battery capacity of device	1000J

B. Results

Fig. 3 shows the average delay of tasks after training with the DDPG, URM, and RRM schemes. When using DDPG, the delay curve converges is 0.5s at 430 episodes. When using URM and RRM, the convergence rates are slow, and the delay is higher than DDPG.

V. Conclusion

In the MEC federation system, we formulate a framework for offloading tasks and resource allocation to optimize the delay and use the DDPG framework to get the results. For the future direction, we want to apply more tight constraints that are close to reality and try to optimize both delay and energy with a trade-off between them.

Acknowledgment

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2022-0-01015, Development of Candidate Element Technology for Intelligent 6G Mobile Core Network). This research was also supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2021-0-02046) supervised by the IITP (Institute of Information & Communications Technology Planning & Evaluation)

References

[1] Alfakih, Taha, et al. "Task offloading and resource allocation for mobile edge computing by deep reinforcement learning based on SARSA." IEEE Access 8 (2020): 54074-54084.
[2] Fu, Fang, et al. "An actor-critic reinforcement learning-based resource management in mobile edge computing systems." International Journal of Machine Learning and Cybernetics 11.8 (2020): 1875-1889.
[3] Liao, Yangzhe, et al. "Joint offloading decision and resource allocation for mobile edge computing enabled networks." Computer Communications 154 (2020): 361-369.
[4] Cheng, Kang, et al. "Energy-efficient joint offloading and wireless resource allocation strategy in multi-MEC server systems." 2018 IEEE international conference on communications (ICC). IEEE, 2018.
[5] Wang, Sihua, et al. "A machine learning approach for task and resource allocation in mobile-edge computing-based networks." IEEE Internet of Things Journal 8.3 (2020): 1358-1372.

[6] Yang, Yiran, et al. “A resource allocation method based on the core server in the collaborative space for mobile edge computing.” 2018 IEEE/CIC International Conference on Communications in China (ICCC). IEEE, 2018.