Delay Optimization for Augmented Reality Service using Mobile Edge Computing Federation System

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

***Abstract*—In the last year, Augmented Reality (AR) applications have used computationally intensive vision algorithms on devices with low resources. Computation offloading has shown to be an effective solution for resource constraints. We propose an AR sub-tasks offloading framework based on the AR service's sub-tasks dependency model to reduce the end-to-end latency in the MEC federation system while taking resource situation into account. The problem is then solved by switching to the Markov Decision Process (MDP) and employing the Deep Deterministic Policy Gradient (DDPG) model. The simulation findings suggest that the DDPG can successfully reduce the AR task latency.**

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

I. Introduction

With strong real-time requirements, Augmented Reality (AR) may be the most computationally expensive multimedia application. A typical AR application analyzes massive volumes of data, such as video feeds, in order to render a virtual layer on top of the actual environment. These procedures are often carried out on mobile devices such as smartphones or smart glasses, which can only carry out rudimentary tasks. As network performance and ubiquity improve, distant devices and servers can run increasing chunks of code. However, the latency limits of AR applications (which can be as low as 20 ms [1]) mean that the available bandwidth and computing capacity on a single connection are insufficient for in time processing.

Several solutions employ numerous servers that collaborate to boost the capacity to tackle AR tasks. However, these models are still limited: the servers connect through the backbone network, which means they must still contact the core network, at a cost that is nearly equal to accessing the cloud center. Other study [2] makes vague assumptions, such as all servers being connected in pairs via a wired link, while [3] only considered the collaboration between adjacent MEC servers for AR tasks executing.

In this paper, we offer an effective task-offloading framework as well as a resource allocation model. To begin, we propose the MEC federation system concept for transferring AR sub-tasks between MEC servers. Second, we concentrate on the end-to-end latency of AR services and propose an optimization problem with resource limitations. Finally, we move this problem to MDP and solve it with the DDPG framework. Our solution provides an efficient framework for computing the dependencies of AR sub-tasks. Finally, we undertake tests to illustrate our system model's superior performance when compared to the dedicated MEC system and neighboring MEC-collaboration without multi-hop system.

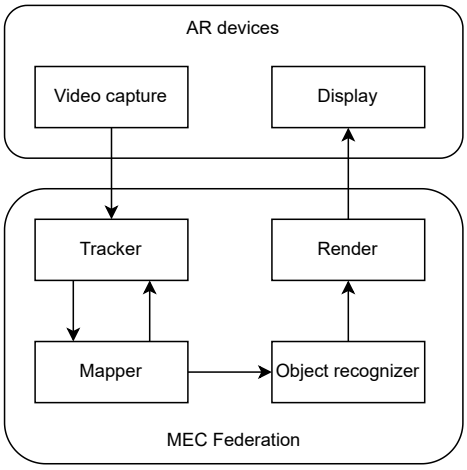

Fig. 1. AR application model

II. System model

The Fig. 1 illustrates six sub-tasks of the AR task model and defines the executed location of each sub-task [3]. The *Video capture* and *Display* always are executed by local AR devices, while *Tracker*, *Mapper*, *Object recognizer* and *Render* can be processed on MEC servers due to heavy computing resource requirement of these sub-tasks. Assume that there are M AR application and each application is denoted by $m \in \{1, 2, ..., M\}$. The $k \in \{1, 2, ..., 6\}$ denotes each sub-tasks of one application.

The MEC federation system model is shown in Fig. 2. There are multiple AR devices of clients denoted by $n \in \{1, 2, ..., N\}$. The MEC federation includes multiple MEC servers $s \in \{1, 2, ..., S\}$, each MEC server can communicate with AR devices by the base stations (BSs). All information of the resources and AR sub-tasks status will be informed to the controller to make the decision that the AR sub-tasks will

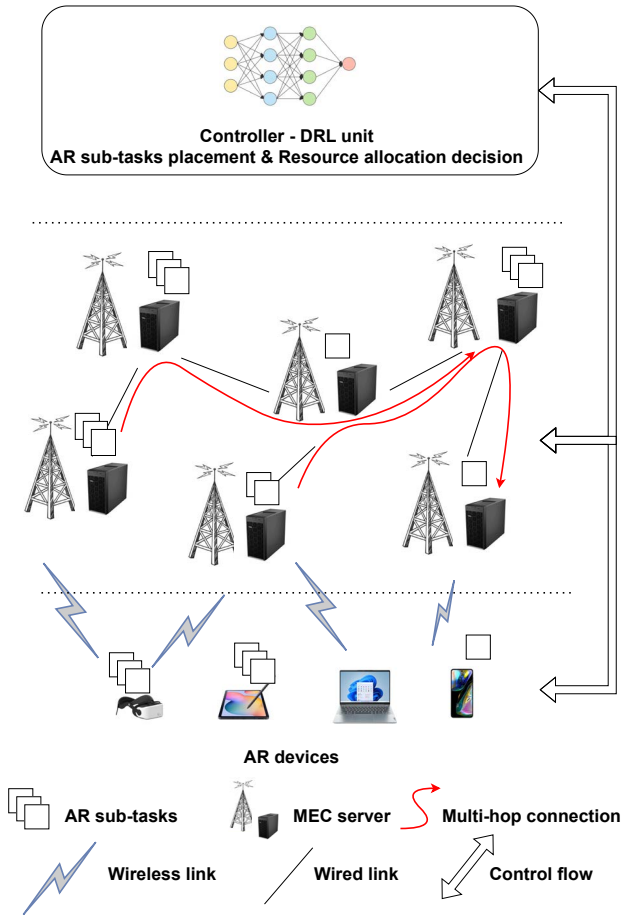

Fig. 2. Proposed system architecture

be executed by the local device, offloaded to local server or migrated to remote server $r \in \{1, 2, ...S | r \neq s\}$ by multi-hop routing.

A. Task model

We saw that one user is only running one AR application in each time, hence m_n represents one AR request made by user n at some point in time. We build a tuple to express the parametric of each sub-task k user n:

$$\Theta_m^k = \{I_n^k, O_n^k, L_n^k, \tau_n^k\},$$

where I_n^k, O_n^k, L_n^k, and τ_n^k signify the size of the input AR sub-task data, the size of the result output data, the number of CPU cycles required to process a unit of data, and the deadline to complete this sub-task. When α_n^k denotes the ratio of data size output to input, the connection is as follows:

$$O_n^k = I_n^k * \alpha_n^k.$$

B. Communication delay model

There are two types of connection in our system: wired and wireless. While the wireless is used to transfer the data between MEC server platform and AR devices platform, the wired is used to transfer data between MEC servers with the neighbors.

- Wireless connection:

The delay from wireless communication for transferring sub-task k between client n and server s:

$$d_{n,k} = \sum_{s=1}^{S} \frac{I_n^k}{R_{n,s}} y_{n,k,s}, \tag{1}$$

where the binary value $y_{n,k,s} \in \{0, 1\}$ denotes the connection status between client n and server s, $R_{n,s}$ denotes the data transmission rate of this link. The $R_{n,s}$ can be calculated as:

$$R_{n,s} = B_{n,s} \log_2 \left(1 + \frac{PG_{n,s}}{N_0 B_{n,s}}\right), \tag{2}$$

where $B_{n,s}$ denotes the channel bandwidth, P denotes the transmission power of the client (uplink) or MEC server (downlink), $G_n(t)$ denotes the wireless channel gain, and $N_0 = -174$ dBm is Gaussian noise power spectrum density.

The process is similar to transferring the results of sub-tasks using a wireless link.

- Wired connection:

The delay from wired communication for transferring sub-task k of client n between two servers s and r by multi-hop is:

$$t_{n,k} = \frac{I_n^k}{\min\{B_{s,r} x_{n,s,r,k} | x_{n,s,r,k} \neq 0\}} + \sigma\left(\sum_{n=1}^{N}\sum_{k=1}^{6} x_{n,s,r,k} - 1\right), \tag{3}$$

where the binary value $x_{n,s,r,k} \in \{0, 1\}$ denotes one of direct connection (no hop) in multi-hop between local MEC server s and server r using for transferring sub-task k of client n, $B_{s,r}$ is channel bandwidth between server s and r, σ is the constant delay value when crossing over each server [4].

C. Computation delay model

The delay of computing process to execute the sub-task k of client n is as below;

$$c_{n,k} = \sum_{s=1}^{S} z_{n,k,s} \frac{I_n^k L_n^k}{f_{n,s}^k} + (1 - \sum_{s=1}^{S} z_{n,k,s}) \frac{I_n^k L_n^k}{f_{n,n}^k}, \tag{4}$$

where $z_{n,k,s}$ is binary value which denotes which place will execute the sub-task, $\sum_{s=1}^{S} z_{n,k,s} = 0$ means the sub-task is executed by AR device, $f_{n,s}^k$ and $f_{n,n}^k$ corresponding are computing resource allocated of server s or device n.

D. Queuing delay model

We are following the First In First Out (FIFO) model, the queuing delay of each sub-task when offload or migrate to each MEC server is total executing time of previous sub-tasks. Therefore, the queuing delay of sub-task k of client n is:

$$q_{n,k} = \sum_{s=1}^{S} \sum_{n'=1, n' \neq n}^{N} \sum_{k'=1, k' \neq k}^{6} z_{n,k,s} c_{n',k'}. \quad (5)$$

E. Problem formulation

The objective of this study is to minimize average end-to-end delay of all AR services using the MEC federation in consideration of the resources. The objective can be expressed below:

$$\min_{x,y,z} \frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{6} (d_{n,k} + t_{n,k} + c_{n,k} + q_{n,k}). \quad (6)$$

We also present five constraints as below:

$$C1: \quad x, y, z \in \{0, 1\}, \quad (7)$$

$$C2: \quad p_{min} \leq P \leq p_{max}, \quad (8)$$

$$C3: \quad F_{available} \geq \sum_{n=1}^{N} \sum_{k=1}^{6} f_{n,k}, \quad (9)$$

$$C4: \quad f_{n,k} \geq 0, \quad f_{n,n} \geq 0, \quad (10)$$

$$C5: \quad y_{n,k,s} = 0, \quad x_{n,s,r,k} = 0, \quad z_{n,k,s} = 1 \quad with \quad k = \{1, 6\}. \quad (11)$$

Constraint 1 guarantees that the sub-tasks placement and routing for multi-hop only can select one options. Constraint 2 presents that the transmission power should not be out of range. Constraint 3 and 4 means the total computing resource allocated for all tasks should not be over the resource available at this time and the computing resource allocated is a positive value. Constraint 5 denotes that the Video capture and Display sub-tasks are executed by AR devices only.

III. Methodology

In this section, we transform Problem (6) 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:

$$S(t) = \{G(t), D(t), E(t), F(t)\}. \quad (12)$$

- Action space:

$$A(t) = \{p(t), f(t), X(t)\}. \quad (13)$$

- 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:

$$R(t) = \frac{-1}{N} \sum_{n=1}^{N} \sum_{k=1}^{6} (d_{n,k} + t_{n,k} + c_{n,k} + q_{n,k}). \quad (14)$$

B. DRL-based resource allocation framework

In this article, we use the DDPG framework, which is one of the Actor-Critic-based DRL categories; the processing is divided into three stages:

- Update the critic network and then calculate the Q-value by minimizing the loss function between the Q-value target and the 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 our system model with dedicated MEC system and neighboring MEC-collaboration without multi-hop.

A. Experimental Setup

We give input random AR first sub-tasks size, the number of clients, and the number of MEC servers, the simulation output will be efficient in AR sub-tasks placement 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 clients	{5, 10, 15, 20}
Number of MEC servers	4
CPU cycles per bit require for each sub-tasks	735 cycles/bit
INput 1st sub-task size	[1, 10] MB
Bandwidth between device-MEC (up/down)	20 MHz
Transmission rate between 2 MECs	150 Mbps
Power range	[5, 38] dBm
Computing resource capacity of device/edge	5/25 GHz
Battery capacity of device	1000J

B. Results

TABLE II
Avg. Delay (ms) of AR services

Number of clients	Dedicated MEC	MEC-collaboration (no multi-hop)	MEC Federation (multi-hop)
5	11.36	5.64	2.37
10	23.68	11.73	5.50
15	39.91	19.84	7.22
20	48.95	25.44	10.02

Table. II shows the average delay of AR services. The results of the average delay analysis for Augmented Reality (AR) services across different scenarios and client counts provide valuable insights into the performance of various Mobile Edge Computing (MEC) deployment strategies. The evaluated scenarios include Dedicated MEC, MEC Collaboration (no multi-hop), and MEC Federation (multi-hop). As the number of clients increases, the average delay is a critical factor that

directly impacts user experience. The findings reveal that the Dedicated MEC approach exhibits the highest average delay values among the three strategies across all client counts, reaching up to 48.95 ms for 20 clients. This outcome suggests that relying solely on Dedicated MEC might not be sufficient to meet the stringent delay requirements for AR applications (20 ms [1]), especially as the number of clients scales.

Remarkably, both MEC Collaboration and MEC Federation consistently outperform the Dedicated MEC approach, showcasing significantly lower average delay values. The MEC Collaboration approach demonstrates notable improvement, offering up to 50% reduction in average delay compared to Dedicated MEC for certain client counts. Furthermore, the MEC Federation strategy achieves the lowest average delay, ranging from 2.37 ms for 5 clients to 10.02 ms for 20 clients. These results highlight the substantial benefits of leveraging multi-hop communication and resource sharing between MEC servers, with MEC Federation demonstrating an impressive average delay reduction of around 80% compared to Dedicated MEC for some scenarios.

Ultimately, the analysis underscores the importance of considering MEC collaboration and federation to ensure that AR services meet the desired delay thresholds, particularly in scenarios with a higher number of clients. The percentage improvements in average delay emphasize the significant strides that can be made in optimizing user experience by adopting advanced MEC deployment strategies, making strides toward meeting the stringent delay requirements of modern AR applications.

V. Conclusion

In the AR-based MEC federation system, we formulate a framework for AR sub-tasks placement 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 minimize the delay while maximize the quality of video to enhance the experience of clients.

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-2023-2021-0-02046) supervised by the IITP (Institute of Information & Communications Technology Planning & Evaluation).

References

[1] M. Abrash, "Latency–the sine qua non of ar and vr," *Blog post, Dec*, 2012.

[2] L. Zhang, X. Wu, F. Wang, A. Sun, L. Cui, and J. Liu, "Edge-based video stream generation for multi-party mobile augmented reality," *IEEE Transactions on Mobile Computing*, 2022.

[3] X. Chen and G. Liu, "Energy-efficient task offloading and resource allocation via deep reinforcement learning for augmented reality in mobile edge networks," *IEEE Internet of Things Journal*, vol. 8, no. 13, pp. 10 843–10 856, 2021.

[4] J. Wang, J. Hu, G. Min, Q. Ni, and T. El-Ghazawi, "Online service migration in mobile edge with incomplete system information: A deep recurrent actor-critic learning approach," *IEEE Transactions on Mobile Computing*, 2022.