Toward a Dynamic Tasks Offloading and Resource Allocation for the Metaverse in In-network Computing

Ibrahim Aliyu[1], Seungmin Oh[1], Sangwon Oh[1], Hyeju Shin[1], Kwangmoo Chung[1],Tai-Won Um[2], Young-Ae Jung[3], Minsoo Hahn[4], Jinsul Kim[1*]

[1]*Department of ICT Convergence System Engineering, Chonnam National University,* Gwangju 61186, Korea.
[2]*Graduate School of Data Science, Chonnam National University,* Gwangju 61186, Korea.
[3]*Division of IT Education, Sunmoon University,* Asan 31460, Korea
[4]*Astana IT University,* Astana, Kazakhastan.
*jsworld@jnu.ac.kr

***Abstract*—** The Metaverse is anticipated to provide an immersive experience to a large group of simultaneous users. Optimal computing resource allocation is critical to meet the massive users' demand. However, existing studies pay little attention to the Metaverse tasks decoupling in joint dynamic resource allocation problems. This paper considers the metaverse tasks decoupling, offloading and resource allocation problem for massive deployment. We model the joint tasks offloading problem as the mixed non-linear programming (MNLP) to minimize network energy consumption and task completion delay. We employ fuzzy c-means (FCM) clustering to group the users into groups. In each zone, a computing resource is available to serve several resource-constraint users. Each user can perform the tasks locally or offload to nearby computing nodes (FIN) or to a more distant, rich edge computing resource (EIN) in the operator's access network, such as MEC. The experimental results suggest that a joint dynamic solution is critical for enabling massive metaverse task offloading.

Keywords*— *In-network computing, Metaverse, Resource allocation, SDN, Tasks offloading

I. INTRODUCTION

The Metaverse is a persistent and immersive simulated environment breaching the physical and digital world through virtual, augmented and extended reality [1]. The massive deployment of Metaverse is expected to bring tremendous demand for computing resources as it transforms how we live, work, and interact with the physical world [2]. Since the Metaverse is expected to provide immerse experience to a large group of simultaneous users [3], optimal computing resource allocation is critical to the simultaneous users' demand.

In-network computing (COIN) paradigm is a promising solution that suggests using unused network resources for performing some tasks to reduce delay and meet QoE [4]. However, adding computing resources or enabling computing in the network would increase power consumption. This competing situation leads us to a joint optimization problem of time delay and energy consumption of metaverse tasks. Although huge progress has been made in solving networks' joint task offloading problem [5-7], most studies focused on atomic task offloading. Furthermore, more attention is needed to understand various metaverse tasks and offloading to COIN-enabled nodes with the corresponding resources needed for the task.

Computation task offloading consist of mainly two modes- full offloading, where all task are either executed remotely or offloaded remotely and partial offloading, where tasks are decomposed into parts with some task processed locally while other parts are computed remotely [8]. In this study, we consider XR computational tasks in Metaverse that can be split into subtasks and processed in a distributed manner by optimally offloading the tasks to COIN resources. We model the joint tasks offloading problem as the mixed non-linear programming (MNLP) and investigated various offloading ratios and clustering of the network. The clustering of massive users into zones is performed using fuzzy c-means (FCM). In each zone, a computing resource is available to serve several resource-constraint users. Each user can perform the tasks locally or offload to nearby computing nodes (FIN) or a more distant, rich edge computing resource in the operator's access network, such as MEC (EIN). We rely on Software-defined networking (SDN) to create and manage network topology. The experimental results suggest that a joint dynamic solution is critical for enabling massive metaverse task offloading.

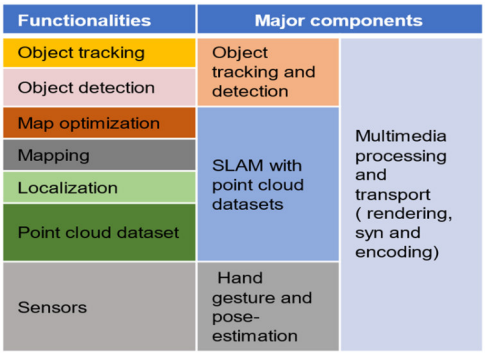
The rest of the paper is organized as follows. Section II discusses related studies, and Section III presents the system model. The problem formulation is discussed in Section IV. Section V discusses dynamic task offloading and resource allocation. Section VI discusses the results and concludes the paper

II. RELATED WORKS

Joint communication and computing resource allocation problems for task offloading have recently received considerable attention, particularly in MEC networks [5-7]. For instance, Jošilo and Dán [9] consider the problem of offloading latency-sensitive computational tasks in edge computing under network slicing. Inter-slice and intra-slice radio and computing resource management were investigated for low-complexity dynamic resource allocation. Meanwhile, other works considered task offloading to the three major resources-caching, communication and computing resources [7, 10, 11].

However, most previous works [12-18] treated the tasks as a single unit and did not consider situations where the tasks could be atomized and handled by different computing nodes. This is particularly important in the Metaverse, where a

metaverse task consists of multiple tasks that can be decomposed and offloaded to different computing nodes (e.g. COIN node). Although [19] proposed XR task decoupling in 5G, their solution offers three upload modes. This may be sufficient for a VR, AR or XR experience with few users. But, for a massive metaverse deployment scenario optimizing the task offloading mode is essential, considering the tremendous simultaneous demand for resource demand in the network. Similarly, Tütüncüoğlu, et al. [20] consider subtask offloading in a serverless edge computing and proposed online learning algorithm for maximizing application utility. Zhang, et al. [21] addressed the subtask dependency offloading problem in MEC and proposed a scheme that minimizes the subtask energy and latency execution. However, these studies considered binary offloading decision-local or serverless edge computing/MEC and a static topology with users always accessing the same resources. Therefore, our study considers dynamic network clustering and computational tasks that can be split into subtasks and performed in a distributed manner by optimally offloading the tasks to EIN or FIN resources. The network is



dynamically partitioned into zones to address mobility issue. In addition to optimal XR computational task splitting in Metaverse, we considered the joint optimization problem of computing resource allocation under delay and energy constraints.

Fig. 1. Typical XR task processing on a device

III. System Model

In this study, we considered XR applications being one of the constituents of Metaverse with growing interest across a spectrum of users. The XR processing entails eight functionalities which can be grouped into four components: object tracking and detection, simultaneous localization, mapping and map optimization (SLAM) with point cloud dataset, hand gesture and pose estimation, and multimedia processing and transport (MPT)-e.g. Rendering, encoding etc.) [19] (see Fig 1.).

In our scenario, a group of users with a Metaverse application simultaneously generate XR tasks on each device. Let a variable, $\Re_i(t) \in [\Re_{i,min}, \Re_{i,max}]$, denote the computational tasks at the user equipment i at time slot t. Each of these tasks can be split into four subtasks (four major components) that can be executed in parallel, series, or combined. The user equipment can locally perform the task (LIN) or offload it to FIN or EIN considering constraints of power, computing capacity, quality of Experience (QoE) as well as demand from other users in the network (see Fig. 2). More so, COIN concept will guarantee computing resources to be added or made available in the network. Thus, we considered the availability of computing resources alongside the joint optimization problem.

A. Communication and computing resources model

In in-network computing, resources or component such as the LIN, FIN and EIN can be connected via wireless or wireline communication resources. Although wireline offers better bandwidth, it is impractical and expensive for a dense network like edge computing [22]. In our scenario, wireless communication with the uplink rate where the access point $a \in \mathcal{A}$ is equally shared among the set of connected node $\mathcal{N}_a$ is considered. Thus uplink rate for the FIN and EIN for the node i in time t, respectively, is [13, 23]:

$$\omega_i^F = \frac{B}{M}\, log\left(1 + \frac{\rho_i^F \eta_i^F}{n_F}\right). \quad (1)$$

$$\omega_i^E = \frac{B}{M}\, log\left(1 + \frac{\rho_i^E \eta_i^E}{n_E}\right). \quad (2)$$

Where ρ_i^F, ρ_i^E are the transmission power for user i to FIN and EIN; η_i^F, η_i^E are the channel gains for the two offloading destinations, and n_F, n_E are the noise powers.

Computing resources COIN may be diverse, ranging from LIN, FIN and EIN to other cloud computing resources. The computing may have different computing capabilities and architecture (as in the case of LIN, FIN and EIN). The notion of clock frequency F^c for every computing $c \in C$ component is introduced to capture the heterogeneity of the computing resources. $k_i^c \in [0,1]$ denotes the suitability of the computing architecture of component c for task $\Re_i(t)$. Meanwhile, $\mathcal{K}_c$ denotes a computationally intensive task assigned to the resource-poor (constraint) component c. Thus, the actual frequency F_i^c at which the resource(component) c executes task $\Re_i \in \mathcal{K}_c$ is:

$$F_i^c = f_{i,c}(F^c, k_i^c, \mathcal{K}_c), \forall c \in C, \forall \Re_i \in \mathcal{K}_c. \quad (3)$$

The task completion time, energy consumption and cost model are discussed as follows.

B. Task completion time

We model the task completion time (time delay) for each of the computing nodes, ie. LIN, FIN and EIN in order to comprehensively model the cost of operating the nodes as follows.

(1) LIN time delay model- The time taken to execute a given task (K_i, L_i) in LIN only includes the processing time on the local node at time t and is defined as:

$$T_i^L(t) = \frac{L_i}{F_i^L(t)} = \frac{K_i \mathcal{U}_i(t)}{F_i^L(t)}, \quad \text{s.t } F_i^L(t) \leq F_{i,max} \quad (4)$$

where F_i^L is the actual frequency at which a local node can execute task.

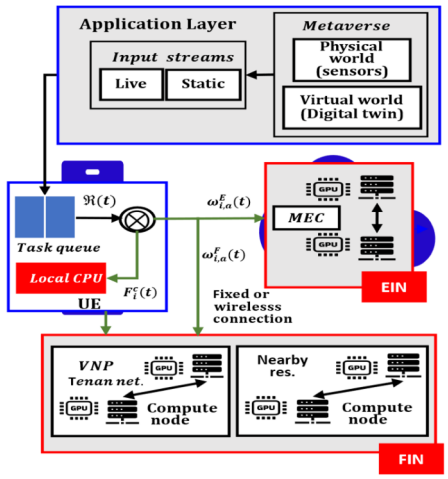


Fig. 2. Metaverse tasks offloading in In-network

(2) FIN time delay model- Offloading a task of input size (K_i, L_i) to external compute nodes, such as FIN, consist of three delay parts. The first part deals with the time needed to transmit the input data K_i through an access point a and is expressed as:

$$T_{i,F}^{t}(t) = \frac{K_i}{\omega_i^F}, \quad \text{s.t } T_i^F(t) \le \theta_{i,F}(t), \tag{5}$$

The second part deals with the time taken to execute the task in the external resource FIN and is expressed as:

$$T_{i,F}^{exe}(t) = \frac{L_i}{F_i^F(t)} \tag{6}$$

where F_i^F is the actual frequency at which the FIN node can execute a task.

The last part of the delay is the time taken to return the computation results from the FIN node to the mobile/local node. However, this delay is observed to be negligible compared to the input data K_i [22, 24-26]. Thus, the total delay for offloading task to the FIN over an access point a can be defind as:

$$T_{i,a}^{F}(t) = T_{i,F}^{t}(t) + T_{i,F}^{exe}(t) \tag{7}$$

(3) EIN time delay model- For simplicity, the EIN node is assumed to have sufficient resources to compute the offloaded task, K_i, transmitted via the cellular network. The tasks can be processed in parallel. The transmission delay, return transmission delay and processing delay determines the time delay. Similar to previous works[22, 24-27], we considered the return delay negligible. At time t, the transmission and execution delay are given as:

$$T_{i,E}^{t}(t) = \frac{K_i}{\omega_i^E}, \tag{8}$$

$$T_{i,E}^{exe}(t) = \frac{L_i}{F_i^E(t)} \tag{9}$$

where F_i^E is the actual frequency at which the EIN node can execute a task.

Therefore, in case of offloading to EIN, the total delay is as follows:

$$T_{i,a}^{E}(t) = T_{i,E}^{t}(t) + T_{i,E}^{exe}(t) \tag{10}$$

C. Energy consumption

The total energy consumption of a mobile terminal is characterized by the node's execution and transmission energy consumption. The transmission energy entails the energy required to offload the task to FIN and EIN. Each component's energy consumption is considered as follows:

(1) LIN energy consumption model-The energy consumption of executing a given task (K_i, L_i) using LIN at frequency F_i^L is linearly proportional to the square of the $F_i^L(t)$ and is given as:

$$\mathcal{E}_i^L = \tau\,(F_i^L(t))^2\,L_i \tag{11}$$

where $\tau \sim 10^{-11}$ [22].

(2) FIN energy consumption model- For offloading task a given task (K_i, L_i) to FIN through access point a, the energy consumption is characterized by the energy used in offloading the task input size a given task K_i, considering the energy for connection scanning is negligible[15, 28]. The energy consumption can be expressed as:

$$\mathcal{E}_{i,a}^{F}(t) = \frac{K_i \wp_{i,a}^F}{\omega_i^F}, \tag{12}$$

where $\wp_{i,a}^F$ is the transmission power of the device through access point a.

(3) EIN energy consumption model- The energy consumption at EIN is similar to FIN. However, the execution energy is negligible [15, 28]. Therefore, the EIN energy consumption can be modelled as:

$$\mathcal{E}_{i,a}^{E}(t) = \frac{K_i \wp_{i,a}^E}{\omega_i^E} \tag{13}$$

where $\wp_{i,a}^E$ is the transmission power the EIN through the access point a.

D. Cost model

Considering the multi-level heterogeneity of COIN compute nodes characterized by compute node capacity, battery energy capacity, type of task and the rate at which tasks are generated, it is necessary to define the preferences of devices over performance metrics [22]. The heterogeneity can be defined in terms of completion time (delay) preference $0 \le \delta_i^T \le 1$ and energy consumption preference $0 \le \delta_i^E \le 1$. The cost model of node i in terms of the completion time and energy consumption time can modeled as follows:

$$\text{LIN cost: } \mathcal{C}_i^L = f(\delta_i^T T_i^L, \delta_i^E \mathcal{E}_i^L) = \delta_i^T T_i^L + \delta_i^E \mathcal{E}_i^L \tag{14}$$

$$\text{FIN cost: } \mathcal{C}_i^F = f(\delta_i^T T_i^F, \delta_i^E \mathcal{E}_i^F) = \delta_i^T T_i^F + \delta_i^E \mathcal{E}_i^F \tag{15}$$

$$\text{EIN cost: } \mathcal{C}_i^E = f(\delta_i^T T_i^E, \delta_i^E \mathcal{E}_i^E) = \delta_i^T T_i^E + \delta_i^E \mathcal{E}_i^E \tag{16}$$

This cost model allows the computing nodes to dynamically adjust objectives factor such as task requirements or battery state by changing of the δ_i^T and δ_i^E values.

IV. PROBLEM FORMULATION

In this study, we considered a mobile COIN system that consists of set of mobile computing node $|N| = N$ communication resource $|\mathcal{A}| = A$ following closely the

formulation of [22]. A given task can be allowed access to computing resources using task placement matrices $\mathcal{X} = \{x_{nf}, x_{ne} | n \epsilon N, f \epsilon F\}. \sum_{f \epsilon F} x_{nf} + x_{ne} = 1$, where $\sum_{f \epsilon F} x_{nf}$ and x_{ne} represent FIN and EIN, respectively. The management policies on the allocation of the computation resource can express as $\mathcal{P}_{\mathcal{X}}: \rightarrow \mathbb{R}^{N\,X\,A}$. Therefore, the system cost is given as $\mathcal{C}(\mathcal{X}, \mathcal{P}_{\mathcal{X}})$. By relying on the definitions, the joint optimization problem for the metaverse task offloading and resource allocation can be formulated as follows:

$$\mathcal{J}_p : \min_{\mathcal{X}, \mathcal{P}_{\mathcal{X}}} \mathcal{C}(\mathcal{X}, \mathcal{P}_{\mathcal{X}}) \tag{17}$$

$$s.t. \quad d1 : \mathcal{B}_i(\mathcal{X}, \mathcal{P}_{\mathcal{X}}) \leq \delta_i, \forall i \in \mathcal{N},$$

$$d2 : \rho(\mathcal{X}, \mathcal{P}_{\mathcal{X}}) \leq \delta_a, \forall a \in \mathcal{A},$$

$$d3 : \sum_{f \in F} x_{nf} + x_{ne} = 1, \forall n \in N$$

$$d4 : x_{nf}, x_{ne} \in [0, 1], \forall n \in N, f \in F$$

$$d5 : \mathcal{P}_{\mathcal{X}}: \rightarrow \mathbb{R}^{N\,X\,A}$$

For the optimization to enforce that a completion time or energy consumption of the devices $i \in \mathcal{N}$ is within the defined threshold, constraint $d1$ is used. Constraint $d2$ limits the amount of computing resources to the devices. The decision to perform computation locally or offload the task to a FIN or EIN is enforced by $d3$ and $d4$. The constraints $d5$ describe allocation policies for computing resources.

V. Dynamic Task Offloading and Resource Allocation (DTR)

Considering the large group of simultaneous users in the Metaverse, we propose clustering the users in the network into zones/clusters. In each zone, there is at least a FIN and EIN node to provide computing resources to users in the zone. The clustering is simply a logical portioning of the nodes in the network to enable scalability and optimal resource allocation and utilization. We employed Fuzzy C-means (FCM) clustering algorithm to cluster the users based on their distances and relied on the SDN concept to create and manage network topology. The FCM is deployed to dynamically create network petitions periodically to meet up with dynamic user demands.

Unlike previous studies [12-18] that focus on atomic task, metaverse divisible task is considered. We assumed each metaverse task $f \epsilon \mathcal{F}$ is denoted by the tuple parameters: $\langle I_f, V_f, P_f \rangle$ where I_f, V_f, P_f are task f input size; software volume and computation load, respectively. The task $f = \{f_{k,0}, f_{k,1}, f_{k,2} \ldots f_{k,j}\}$ where k is the user and j is the number of subtasks. We then investigated several offloading configurations to establish the problem of joint dynamic tasks offloading problem (see Fig. 3). The system simulation was conducted using python 3 on a Windows 10, Corei5 system.

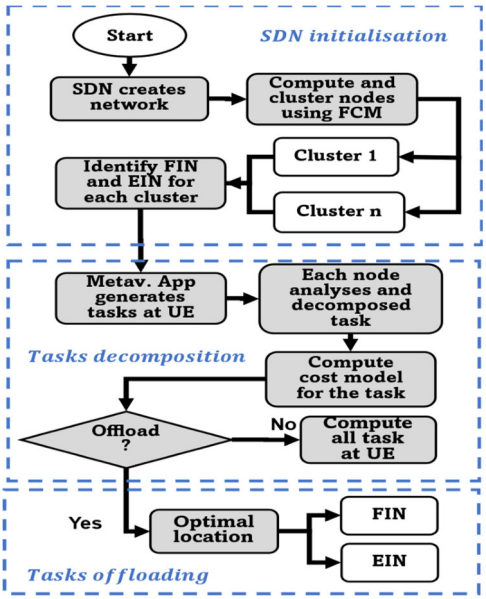


Fig. 3. Dynamic metaverse tasks offloading and resource allocation flowchart

VI. Results and Conclusion

We considered a separate experiment for groups of 16 and 25 users. For each experiment, the FCM cluster the users into two groups and assign computing nodes to each. 50 rounds are set for the task execution. Table 1 presents the experimental setup for the simulation. As indicated in the Table, each is assigned expected task completion time and energy consumption by relying on literature [29]. We split SLAM with the point cloud dataset into local and global, as the operation can be split into two stages. But we assumed that object tracking and detection are conducted at the user node, so we focused on the other task to examine the problem. Several offloading ratios were investigated to observe the variation in energy consumption and task completion time.

TABLE I. EXPERIMENTAL SETTINGS AND PARAMETERS

Users	Task/node	Completion time (30ms)	Energy (J)	Offload ratio
Tasks	Object tracking and detection	0.2	0.030	50:50, 40:60, 30:70, 60:40, 70:30
	SLAM local	0.1	0.07	
	SLAM global	0.2	0.082	
	Hand gesture and pose estimation	0.2	0.200	
	MPT	0.3	0.300	

For the 16 users' scenario, the 70:30 offload ratio performed the worst, with about 701ms completion time in about 27 rounds. 50:50 ratio recorded the least completion time of less than 101ms in about 37 rounds (see figure Fig. 4). Similarly, for the 25 users' scenario, the 50:50 offload ratio performance is the worst, with about 750ms completion time in about 36 rounds. 70:30 ratio recorded the least completion time of less than 100ms in about 38 rounds.

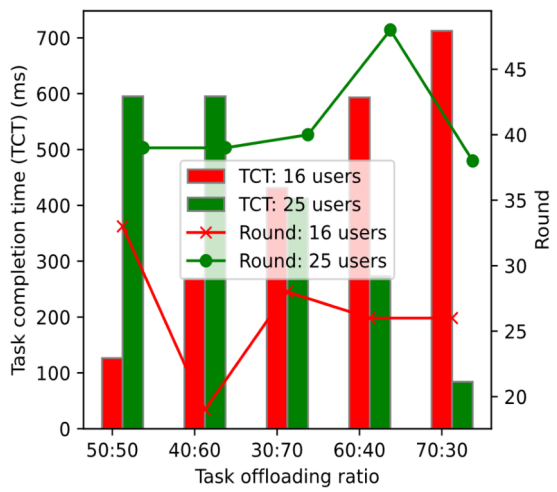


Fig. 4. Tasks completion time and round for the offload ratios

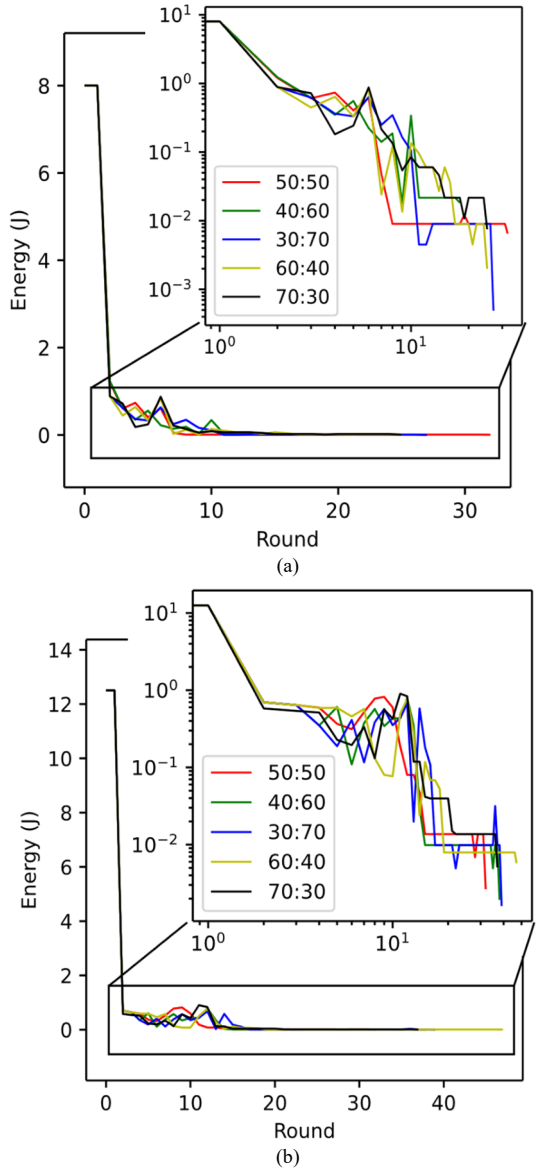


Fig. 5. Energy consumption (a) 16 users (b) 25 users

Furthermore, we consider the performance of the various offloading ratio in terms of energy consumption. For the 16 users' experiment, about l8.0024 J energy was consumed for all cases at the start of the task execution and gradually went to zero as the execution converged. The 50:50 shows no further sign of energy consumption at around 8 rounds, while other ratios show no further energy consumption after 10 rounds. To enhance the visibility of the energy consumption plot in Fig. 5, a logarithmic scale is applied on the x- and y-axis. As shown in Fig. 5(a), the 50:50 energy consumption converges faster, but 30:70 attended the energy lower energy after few more rounds. In addition, 12.5038 J was dissipated at the start of task execution for all cases in the 25 users experiment. The energy consumption gradually went to zero as the execution converged. The 50:50 shows no further energy consumption at around 10 rounds (see Fig. 5(b)). The rest of the offload ratio convergence around 12 rounds, with 30:70 experiencing a spark. More energy consumption details are projected by the logarithms scaling of the y- and x-axis. Thus, it is challenging to define the optimal ratio considering network parameters and demands frequently change.

The experimental results suggest that a joint dynamic solution is critical for enabling metaverse task offloading. This is still an ongoing study that will ultimately proper an algorithm that dynamically adapt to the changing network conditions and demand. Future works would focus on improving the joint optimization model to include cache and communication policies and developing of new algorithm to solve the problem.

ACKNOWLEDGEMENT

This work was supported by the Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (2021-0-02068, Artificial Intelligence Innovation Hub) and supported by the MSIT(Ministry of Science and ICT), Korea, under the Innovative Human Resource Development for Local Intellectualization support program(IITP-2023-RS-2022-00156287) supervised by the IITP(Institute for Information & communications Technology Planning & Evaluation).

REFERENCES

[1] L.-H. Lee *et al.*, "All one needs to know about metaverse: A complete survey on technological singularity, virtual ecosystem, and research agenda," *arXiv preprint arXiv:2110.05352,* 2021.

[2] Y. Cai, J. Llorca, A. M. Tulino, and A. F. Molisch, "Compute-and Data-Intensive Networks: The Key to the Metaverse," *arXiv preprint arXiv:2204.02001,* 2022.

[3] L. Rosenberg. "VR vs. AR vs. MR vs. XR: What's the difference?" https://bigthink.com/the-future/vr-ar-mr-xr-metaverse/ (accessed 02/11, 2022).

[4] S. Huang *et al.*, "Intelligent Eco Networking (IEN) III: A Shared In-network Computing Infrastructure towards Future Internet," in *2020 3rd International Conference on Hot Information-Centric Networking (HotICN)*, 2020: IEEE, pp. 47-52.

[5] J. Zhang *et al.*, "Joint resource allocation for latency-sensitive services over mobile edge computing networks with caching," *IEEE Internet of Things Journal,* vol. 6, no. 3, pp. 4283-4294, 2018.

[6] K. Guo, M. Yang, Y. Zhang, and J. Cao, "Joint computation offloading and bandwidth assignment in cloud-assisted edge computing," *IEEE Transactions on Cloud Computing,* vol. 10, no. 1, pp. 451-460, 2019.

[7] K. Poularakis, J. Llorca, A. M. Tulino, I. Taylor, and L. Tassiulas, "Service placement and request routing in MEC networks with storage, computation, and communication constraints," *IEEE/ACM Transactions on Networking,* vol. 28, no. 3, pp. 1047-1060, 2020.

[8] X.-Q. Pham, T. Huynh-The, E.-N. Huh, and D.-S. Kim, "Partial computation offloading in parked vehicle-assisted multi-access edge computing: A game-theoretic approach," *IEEE Transactions on Vehicular Technology,* vol. 71, no. 9, pp. 10220-10225, 2022.

[9] S. Jošilo and G. Dán, "Joint wireless and edge computing resource management with dynamic network slice selection," *IEEE/ACM Transactions on Networking,* 2022.

[10] A. Ndikumana *et al.*, "Joint communication, computation, caching, and control in big data multi-access edge computing," *IEEE Transactions on Mobile Computing,* vol. 19, no. 6, pp. 1359-1374, 2019.

[11] Q. Chen, F. R. Yu, T. Huang, R. Xie, J. Liu, and Y. Liu, "Joint resource allocation for software-defined networking, caching, and computing," *IEEE/ACM Transactions on Networking,* vol. 26, no. 1, pp. 274-287, 2018.

[12] G. Lia, M. Amadeo, G. Ruggeri, C. Campolo, A. Molinaro, and V. Loscrì, "In-network placement of delay-constrained computing tasks in a softwarized intelligent edge," *Computer Networks,* vol. 219, p. 109432, 2022.

[13] Z. Chen, W. Yi, A. S. Alam, and A. Nallanathan, "Dynamic Task Software Caching-Assisted Computation Offloading for Multi-Access Edge Computing," *IEEE Transactions on Communications,* vol. 70, no. 10, pp. 6950-6965, 2022.

[14] S. Liang, H. Wan, T. Qin, J. Li, and W. Chen, "Multi-user computation offloading for mobile edge computing: A deep reinforcement learning and game theory approach," in *2020 IEEE 20th International Conference on Communication Technology (ICCT)*, 2020: IEEE, pp. 1534-1539.

[15] S. Yang, "A joint optimization scheme for task offloading and resource allocation based on edge computing in 5G communication networks," *Computer Communications,* vol. 160, pp. 759-768, 2020.

[16] L. Wu *et al.*, "DOT: Decentralized Offloading of Tasks in OFDMA-Based Heterogeneous Computing Networks," *IEEE Internet of Things Journal,* vol. 9, no. 20, pp. 20071-20082, 2022.

[17] J. Chen, Q. Deng, and X. Yang, "Non-cooperative game algorithms for computation offloading in mobile edge computing environments," *Journal of Parallel and Distributed Computing,* vol. 172, pp. 18-31, 2023.

[18] W. Yu, T. J. Chua, and J. Zhao, "Asynchronous Hybrid Reinforcement Learning for Latency and Reliability Optimization in the Metaverse over Wireless Communications," *arXiv preprint arXiv:2212.14749,* 2022.

[19] F. Alriksson, D. H. Kang, C. Phillips, J. L. Pradas, and A. Zaidi. "XR and 5G: Extended reality at scale with time-critical communication." https://www.ericsson.com/en/reports-and-papers/ericsson-technology-review/articles/xr-and-5g-extended-reality-at-scale-with-time-critical-communication (accessed 02/11, 2022).

[20] F. Tütüncüoğlu, S. Jošilo, and G. Dán, "Online Learning for Rate-Adaptive Task Offloading Under Latency Constraints in Serverless Edge Computing," *IEEE/ACM Transactions on Networking,* 2022.

[21] Y. Zhang, J. Chen, Y. Zhou, L. Yang, B. He, and Y. Yang, "Dependent task offloading with energy‐latency tradeoff in mobile edge computing," *IET Communications,* vol. 16, no. 17, pp. 1993-2001, 2022.

[22] S. Josilo, "Task placement and resource allocation in edge computing systems," KTH Royal Institute of Technology, 2020.

[23] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM transactions on networking,* vol. 24, no. 5, pp. 2795-2808, 2015.

[24] D. Van Huynh, S. R. Khosravirad, A. Masaracchia, O. A. Dobre, and T. Q. Duong, "Edge intelligence-based ultra-reliable and low-latency communications for digital twin-enabled metaverse," *IEEE Wireless Communications Letters,* vol. 11, no. 8, pp. 1733-1737, 2022.

[25] T. Do-Duy, D. Van Huynh, O. A. Dobre, B. Canberk, and T. Q. Duong, "Digital twin-aided intelligent offloading with edge selection in mobile edge computing," *IEEE Wireless Communications Letters,* vol. 11, no. 4, pp. 806-810, 2022.

[26] T. Liu, L. Tang, W. Wang, Q. Chen, and X. Zeng, "Digital-twin-assisted task offloading based on edge collaboration in the digital twin edge network," *IEEE Internet of Things Journal,* vol. 9, no. 2, pp. 1427-1444, 2021.

[27] S. Jošilo and G. Dán, "Joint wireless and edge computing resource management with dynamic network slice selection," *IEEE/ACM Transactions on Networking,* vol. 30, no. 4, pp. 1865-1878, 2022.

[28] J. Zhao, Q. Li, Y. Gong, and K. Zhang, "Computation offloading and resource allocation for cloud assisted mobile edge computing in vehicular networks," *IEEE Transactions on Vehicular Technology,* vol. 68, no. 8, pp. 7944-7956, 2019.

[29] *5G;Extended Reality (XR) in 5G (3GPP TR 26.928 version 16.1.0 Release 16)*, 3GPP, 2021. [Online]. Available:https://www.etsi.org/deliver/etsi_tr/126900_126999/126928/16.01.00_60/tr_126928v160100p.pdf