# Artificial Intelligence-based Service Aggregation for Mobile-Agent in Edge Computing

Md. Shirajum Munir, Sarder Fakhrul Abedin, and Choong Seon Hong*
*Department of Computer Science and Engineering, Kyung Hee University,*
*Yongin-si 17104, Republic of Korea*
E-mail: munir@khu.ac.kr, saab0015@khu.ac.kr, cshong@khu.ac.kr

***Abstract*—The ongoing development of edge computing in fifth-generation (5G) networks promises to provide an artificial intelligence-as-a-service (AIaaS) for meeting the stringent requirements of everything as a service (XaaS) in the edge of the networks. Therefore, the concept of edge-artificial intelligence (edge-AI) is not only evolving but also emergent enabler toward AI service fulfillment. In this paper, we investigate an AI-based service aggregation problem for a mobile agent in AIaaS-enabled edge computing. First, we propose an optimization problem for the mobile agent and the objective is to maximize the AI service fulfillment achieved rate while satisfying the computational, memory, and delay requirements. Thus, we show that this optimization problem is NP-hard. Second, we compel the formulated problem in a community discovery problem and derive a solution by executing a data-driven approach. To do this, we incorporate density-based spatial clustering of applications with noise (DBSCAN) and flow control algorithm, and propose a low computational complexity algorithm for AI service aggregation of the mobile agent. Finally, numerical analysis shows the proposed model can perform better over other baseline methods in terms of deprived AI services, server utilization, and complexity analysis.**

***Index Terms*—Edge-AI, mobile edge computing (MEC), artificial intelligence-as-a-service (AIaaS), data-driven, mobile agent.**

## I. INTRODUCTION

In recent years, edge computing superintended the artificial intelligence (AI)-based services for the 5G and beyond, which enables the AI facilities in the edge of the network. Therefore, the role of this edge-artificial intelligence (edge-AI) can be divided into two categories [1], first, AI for edge computing (i.e., AI-powered network management, edge AI over wireless systems, and so on) and second, edge computing for AI (i.,e., tactile sensing services, health-care services, smart energy, smart transportation services, virtual reality (VR) tracking, smart agriculture, etc.) [2], [3]. In order to perform these scalable AI services with low-latency and high-reliability in the edge, artificial intelligence-as-a-service (AIaaS)-enabled infrastructure is needed for the edge computing.

The AIaaS market compound annual growth rate (CAGR) is expected to reach $56.7\%$ in 2025, where the predicted market size is around $\$77,047.7$ million [4] and AIaaS is shifting from cloud to the fog [5] and edge. Further, the potentiality of edge-AI services are already taken into account by the infrastructure and service providers such as content caching in the edge [6], smart energy management for the wireless network [7], AI-based smart city service management in software-defined networks (SDN) [8], federated learning model [9] for wireless network, intelligent psychiatric emergency application in medical domain [10], and so on. Additionally, the deployment of mobile edge computing (MEC) infrastructure is not only limited to a static environment but also exposes aerial support using MEC-enabled unmanned aerial vehicles (UAVs) for scaling up the efficiency of the edge computing [11], [12]. Thus, to enable the eavesdroppers AI services in the edge, a mobile-agent-based AIaaS infrastructure is required. Additionally, a mobile agent is capable of controlling AI services, and also can impose edge computing-empowered aerial support by UAV for the network. A mobile-agent supported AIaaS-enabled architecture for edge computing is shown in Fig. 1. To the best of our knowledge, this paper provides one of the first models for AI service aggregation in AIaaS-enabled edge computing.

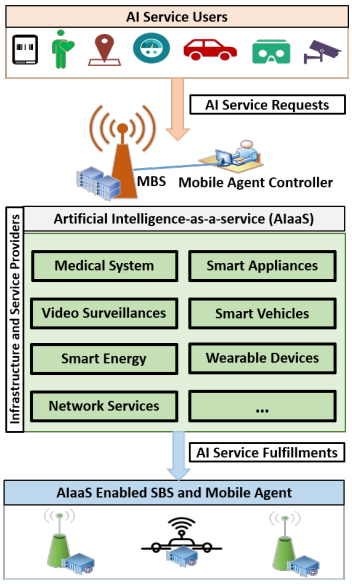


Fig. 1. Artificial Intelligence-as-a-Service (AIaaS)-enabled architecture for edge computing.

To design an *AIaaS-enabled AI service aggregation for mobile-agent* in edge computing, we face various challenges:

- First, how to accomplish the spatiotemporal requirements

This work was supported by Institute for Information & Communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2015-0-00557,Resilient/Fault-Tolerant Autonomic Networking Based on Physicality, Relationship and Service Semantic of IoT Devices).

***Dr. CS Hong** is the corresponding author.

for the heterogeneous AI service requests, where this requirement belongs to both space and time together. Thus, by analyzing the nature of AI service requests possibly meets these requirements.

- Second, how to cope with a huge amount of AI services, where these service have more variety of requirements with respect to computational, memory, delay tolerance. Therefore, a data-driven approach can be a possible way that discretized the characteristics of requests and AI services.
- Third, even the data-driven approach can deal with the nature of AI services and its requests toward the AI service aggregation. Hence, it is hard to manage when the computational resource is limited.

To address the stringent challenges, we summarize our main contributions as follows:

- First, we formulate an *AI service aggregation* problem for mobile agent in artificial intelligence-as-a-service (AIaaS)-enabled edge computing, where the objective is to maximize the AI service fulfillment achieved rate under the computational, memory, and tolerable delay constraints. Thus, the formulated problem is NP-hard while it is hard to find a globally optimal solution even if there exists a solution.
- Second, to devise a solution using a data-driven approach, we analyze the formulated problem with a community discovery problem, where we impose a density-based spatial clustering of applications with noise (DBSCAN) and flow control algorithm. Hence, we propose an AI service aggregation algorithm for mobile agent.
- Finally, we perform an extensive numerical analysis to justify our proposed scheme, where we show the AI service aggregation for mobile agent model outperforms other baseline approaches. This proposed model gains up to $12\%$ more efficient MEC server utilization and also increases the AI service fulfillment achieved rate up to $9\%$ with low computational complexity solution.

The reminder of this paper is organized as follows: we present the proposed system model and problem formulation of the AI service aggregation for the mobile agent in Section II. Section III represents the AI service aggregation solution via a data-driven approach for mobile agent. We discuss the numerical analysis in Section IV. Finally, we conclude our discussion in Section V.

## II. System Model and Problem Formulation

### A. System Model of AI Service Aggregation for Mobile Agent in AIaaS-Enabled Edge Computing

We consider an artificial intelligence-as-a-service (AIaaS)-empowered wireless network as shown in Fig. 2, where a set of edge server-enabled small cell base stations (SBSs) $\mathcal{B} = \{0, 1, 2, \ldots, B\}$ are employed under a macro base station (MBS). A single AIaaS-enabled mobile SBS (UAV) $m$ (i.e., $m = b = 0$) is taken into account with the same capacity of a SBS $b$ for ensuring the maximized achieved

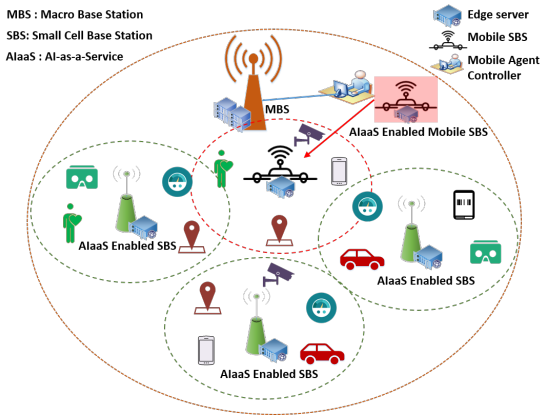


Fig. 2. System model of AI service aggregation for mobile agent in artificial intelligence-as-a-service (AIaaS)-enabled edge computing.

rate of AI services and can communicate with MBS via a UAV subscribed channel. This AIaaS-enabled network is capable of performing a set of heterogeneous delay sensitive AI services $\mathcal{K} = \{1, 2, \ldots, K\}$ (e.g., smart grid, smart health, smart traffic, emergency monitoring, and so on) based on user-specific requirements. The requirement of AI services not only depends on user demand (i.e., latency) but also build upon the characteristics of AI service model (i.e., computational and memory). Therefore, the properties of AI service $k \in \mathcal{K}$ is defined as a tuple $\langle \alpha_k, \beta_k, \gamma_k \rangle$, where $\alpha_k$, $\beta_k$, and $\gamma_k$ are expected computational units (CPU cycles/second), memory requirement, and maximum tolerable delay (i.e., computation and communication), respectively.

*1) Computation and Communication Model of AI Service:* In this model, we consider a given uplink transmission data rate $r_b$ between AI service $k \in \mathcal{K}$ and SBS $b \in \mathcal{B}$, and an user association indicator $\phi_{bk}$, where $\phi_{bk} = 1$ if AI service user $k$ is assigned to SBS $b$, and 0 otherwise. The data rate is defined as follows [7], [13]:

$$r_b(\phi_{bk}) = \begin{cases} \sum_{b\in\mathcal{B}} \sum_{k\in\mathcal{K}} \phi_{bk} w_{bk} \\ \log_2\left(1 + \frac{p_{bk} g_{bk}}{\sigma_{bk}^2 + \sum_{j\in\mathcal{B}, j\neq b} I_j(t)}\right), \text{if } \phi_{bk} = 1, \\ 0, \qquad \text{otherwise}, \end{cases} \tag{1}$$

where a fixed channel bandwidth is defined by $w_{bk}$, $p_{bk}$ determines an uplink power transmission between AI service $k \in \mathcal{K}$ and SBS $b \in \mathcal{B}$, the channel gain is represented by $g_{bk}(t)$, a variance of an Additive white Gaussian noise (AWGN) is given by $\sigma_{bk}^2$, and the transmission channel interference with other SBSs is denoted by $I_j$. Therefore, we consider a task arrival rate $\lambda_b$ that follows the Poisson process at SBS $b \in \mathcal{B}$ with an average AI service requests (traffic) size, $S_b = \frac{1}{|\mathcal{K}|} \sum_{k\in\mathcal{K}} \beta_k$. Hence, an average traffic load is defined by $\lambda_b S_b(t)$ with a given average data rate $r_b(\phi_{bk})$ that follows the general distribution. Let us consider a $M/G/1$ queuing model for this communication model since a task arrival rate follows the Poisson process. Therefore, a service time $T_s$ of an AI service task is determined by a general distribution [14]. Hence, the distribution of service time belongs to $\mathbb{E}[T_s]$ and $\mathbb{E}[T_s^2]$. We consider the service time $T_s$ are independent

and identically distributed ($i.i.d.$) and also, this is independent from an AI service request arrival time. Thus, for a service rate $\mu_b = \frac{\lambda_b S_b}{r_b}$ of SBS $b$, an expected service time is determined by $\mathbb{E}[T_s] = \frac{1}{\mu_b}$.

The edge server utilization depends on the expectation of service time $\mathbb{E}[T_s]$ and arrival rate $\lambda_b$ of the AI service requests. Therefore, overall server utilization is determined by $\rho_b = \sum_{\forall k \in \mathcal{K}} \frac{\lambda_b}{\mu_b} = \sum_{k \in \mathcal{K}} \lambda_b \mathbb{E}[T_s]$. Here, a waiting time $T_w$ represents a discrete time and the waiting time determines an amount of time which is needed to spend in the queue before an AI service execution. Using the Pollaczek-Khinchin mean formula [15], the expected waiting time $\mathbb{E}[T_w]$ is defined as, $\mathbb{E}[T_w] = \frac{\lambda_b \mathbb{E}[T_s^2]}{2(1-\rho_b)}$, where $\rho_b$ represents the server utilization of the network. Therefore, the expected system time of an AI service that includes both execution (AI service computation) time and a waiting time (in the queue). Thus, the average system time is as follows:

$$\tau_b = \mathbb{E}[T] = \mathbb{E}[T_s] + \frac{\lambda_b \mathbb{E}[T_s^2]}{2(1-\rho_b)}. \tag{2}$$

Let us consider the following binary decision variable:

$$x_{bk} = \begin{cases} 1, \text{ if service } k \text{ is assigned to SBS } b \in \mathcal{B}, \\ 0, \qquad \text{otherwise,} \end{cases} \tag{3}$$

where $x_{bk} = 1$ if AI service $k$ is served by SBS $b$, and 0 otherwise. We consider a maximum computational capacity $\Psi_b^{max}$ for SBS $b$, where value of $\Psi_b^{max}$ is fixed and relies on the vendor of devices [7]. Thus, the residual of computational capacity for SBS $b$ is determined as follows:

$$\Psi_b = \Psi_b^{max} - \sum_{k \in \mathcal{K}} \phi_{bk} x_{bk} \alpha_k, \tag{4}$$

where $\alpha_k$ is a given computational requirement of AI service $k$. Further, for a given maximum memory capacity $\Phi_b^{max}$ of SBS $b$, the surplus memory of a SBS $b$ is determined as follows:

$$\Phi_b = \Phi_b^{max} - \sum_{k \in \mathcal{K}} \phi_{bk} x_{bk} \beta_k, \tag{5}$$

where $\beta_k$ denotes the memory requirements of AI service $k$.

*2) AIaaS-Enabled Mobile SBS Model:* In this paper, we consider one AIaaS-enabled mobile SBS $m$ (UAV) that is the mobile agent to serve the AI services under the MBS while other SBSs are unable to perform these AI services due to the resource limitation. Let us consider mobile agent $m$ can hover with a fixed height $h_m$ and is capable of moving from the mobile agent ground station to a high-density service requests locations with a fixed velocity $v_m$ [11]. In this model, a mobile agent controller is responsible for activating the mobile agent $m$ in on-demand basis, and also this mobile agent can move from current location to a new destination during it's maximum flying capacity. Let us consider a trajectory between the source and destination $d_m$, where we measure an Euclidean distance (straight-line distance) [12] from source (longitude and latitude) to destination (longitude and latitude). Thus, the traverse distance is determined as follows:

$$d_m = \sqrt{(Lon_m^{cur} - Lon_m^{des})^2 + (Lat_m^{cur} - Lat_m^{des})^2}, \tag{6}$$

where $Lon_m^{cur}$, $Lon_m^{des}$, $Lat_m^{cur}$, and $Lat_m^{des}$ are source longitude, destination longitude, source latitude, and destination latitude, respectively. Hence, a traverse time $T_m^{tvs}$ is determined as follows:

$$T_m^{tvs} = \frac{d_m}{v_m} + \delta_m T_m^{int}, \tag{7}$$

where $\delta_m$ is the coefficient of mobile agent initialization time $T_m^{int}$ and the value of $\delta_m$ depends on type of mobile agents [11]. Since capacity of mobile agent $m$ is same as other SBSs $\forall b \in \mathcal{B} \backslash \{0\}$, thus the average service time $\tau_m$ of mobile agent $m$ is as follows:

$$\tau_m = \tau_b + T_m^{tvs}, \tag{8}$$

where $\tau_b$ is determined by (2). The maximum computational capacity of the mobile agent $m$ is also same as other SBSs $\Psi_b^{max}$, thus the mobile agent current computational capacity can be calculated is as follows:

$$\Psi_m = \Psi_b^{max} - \sum_{b \in \mathcal{B} \backslash \{0\}} \sum_{k \in \mathcal{K}} (1 - x_{bk}) \alpha_k. \tag{9}$$

The surplus memory $\Phi_m$ of the mobile agent is determined as follows:

$$\Phi_m = \Phi_b^{max} - \sum_{b \in \mathcal{B} \backslash \{0\}} \sum_{k \in \mathcal{K}} (1 - x_{bk}) \beta_k, \tag{10}$$

where $\Phi_b^{max}$ is the maximum memory capacity, which is same for both mobile agent and other SBSs.

In this system model, we observe the AI service fulfillment achieved rate $\Lambda(\mathbf{x})$ for all SBSs while the deprived AI services are executed by utilizing the mobile agent $m$. Thus, the observable achieved rate of SBSs $\forall b \in \mathcal{B} \backslash \{0\}$ is as follows:

$$\Lambda(\mathbf{x}) = \frac{\sum_{b \in \mathcal{B}} \sum_{k \in \mathcal{K}} \phi_{bk} x_{bk}}{|\mathcal{K}|}, \tag{11}$$

where $\mathbf{x} = \{x_{bk}, \forall b \in \mathcal{B} \backslash \{0\}, \forall k \in \mathcal{K}\}$.

### *B. Proposed AI Service Aggregation Problem Formulation for Mobile Agent*

The goal of this AI service aggregation problem is to maximize the AI service fulfillment achieved rate for the all artificial intelligence-as-a-service (AIaaS)-enabled MEC supported SBSs, where computational, memory, and maximum tolerable delay requirements are imposed by the AI services. Therefore, the problem is formulated as follows:

$$\max_{\mathbf{x}} \sum_{b \in \mathcal{B}} \sum_{k \in \mathcal{K}} \Lambda(\mathbf{x}) + \omega(1 - \Lambda(\mathbf{x})), \tag{12}$$

$$\text{s.t.} \quad x_{bk} \Psi_b + (1 - x_{bk}) \Psi_m \geq \alpha_k, \tag{12a}$$

$$x_{bk} \Phi_b + (1 - x_{bk}) \Phi_m \geq \beta_k, \tag{12b}$$

$$x_{bk} \tau_b + (1 - x_{bk}) \tau_m \leq \gamma_k, \tag{12c}$$

$$\omega \sum_{k \in \mathcal{K}} (1 - x_{bk}) \leq \hat{\omega}, \tag{12d}$$

$$\sum_{b \in \mathcal{B}} \sum_{k \in \mathcal{K}} x_{bk} \leq K, \forall b \in \mathcal{B}, \tag{12e}$$

$$x_{bk} \in \{0, 1\}, \forall k \in \mathcal{K}. \tag{12f}$$

In problem (12), constraints (12a), (12b), and (12c) satisfy the computational, memory, and tolerable delay of the AI services, respectively. In constraint (12a), the computational capacity of SBSs are determined by (4) and (9) decides computational capacity of the mobile agent $m$. Constraint (12b) determines the memory capacity of SBSs and mobile agent using (5) and (10), respectively. The AI services execution delay are calculated in (2) and (8) for SBSs and mobile agent, respectively for fulfilling the constraint (12c). Therefore, the decision variable $x_{bk} = 1$ indicates the AI service is executed in MEC-enabled SBS, and $x_{bk} = 0$ otherwise (deprived AI service). Hence, constraint (12d) admits a coupling between SBSs and mobile agent $m$, where a penalty coefficient $\omega = [0, 1]$ is used as an input of the objective function and $\hat{\omega}$ denotes capacity of AI service fulfillment by the mobile agent. Thus, constraint (12d) ensures the activation of mobile agent $m$ and constraint (12e) guarantees the AI service fulfillment decision for all requests $\forall k \in \mathcal{K}$ under the MBS. Finally, constraint (12f) ensures that the decision variable $x_{bk}$ is a binary variable.

Since the formulate problem (12) is a mixed-integer programing problem, with the corresponding constraints from (12a) to (12f). The problem (12) can be reduced to a $0/1$ multiple-knapsack problem as a base problem [16]. In general, the $0/1$ multiple-knapsack is NP-Complete [17], which infer that the problem (12) is NP-Complete. Therefore, the complexity of problem (12) leads exponentially $\mathcal{O}(2^{|\mathcal{K}|\times|\mathcal{B}|})$. As a result, we can infer that problem (12) is NP-hard, which is similar to the multiple-knapsack problems [18]. In fact, it is extremely hard to obtain a globally optimal solution of the problem (12). Therefore, to obtain a solution of (12), we propose a data-driven approach, where we adopt the thought of density-based spatial clustering of applications with noise (DBSCAN) with flow control algorithm. This approach efficiently manages AI services for AIaaS operation while considering the computational, memory, and tolerable delay requirements. The solution of AI service aggregation via data-driven approach is discussed in later section.

## III. AI Service Aggregation Solution via Data-Driven Approach

We devise a solution of *AI service aggregation for mobile agent* problem (12) through a data-driven approach. To do this, we employ a community discovery problem [19], in which this problem is similar to a label propagation method. Thus, we consider an additional decision variable $\xi$ to ensure a well partition SBSs selection for AI service execution. Additionally, $\xi$ is a control variable, which assures the distance between inter-class while considering a three-dimensional space (i.e., computation, memory, and tolerable delay requirements). Thus, the problem (12) can be represented as follows:

$$\max_{\xi,\mathbf{x}} \sum_{b\in\mathcal{B}} \sum_{k\in\mathcal{K}} \Lambda(\mathbf{x}) + \omega(1 - \Lambda(\mathbf{x})), \tag{13}$$

$$\text{s.t.} \quad \text{(12a) to (12f)}. \tag{13a}$$

In problem (13), all of the constraints from (12a) to (12f) remain same as in problem (12). The proposed AI service aggregation for mobile agent model in Algorithm 1, which is run by the mobile agent controller in MBS. The inputs of this algorithm are AI service requests with corresponding requirements (i.e., computation, memory, and tolerable delay) and the outputs are AI service allocation with mobile agent activation $y_m$ decision. Therefore, lines 5 to 20 (in Algorithm 1) are involved for AI service fulfillment, where in line 8, the constraints (12a), (12b), and (12c) are evaluated for computation, memory, and tolerable delay, respectively. The AI service assignments in SBSs are dealing in lines 10 to 16, where lines 10 to 12 take decision for mobile agent and SBS assignment are made in lines 13 to 16. Further, the mobile agent activation decision is taken place from lines 24 to 29, in which the distance of mobile agent trajectory is made in line 25 (in Algorithm 1). The value of $Lon_m^{des}$ and $Lat_m^{des}$ for (6) are determined by the concept of core point from the proposed DBSCAN-based Algorithm 1. Finally, Algorithm 1 returns the AI service fulfillment and mobile agent activation

**Algorithm 1** AI Service Aggregation for Mobile Agent-based on DBSCAN

**Input:** $\mathcal{K}, \forall\langle\alpha_k, \beta_k, \gamma_k\rangle \in \mathcal{K}$, $\omega$
**Output:** $\mathbf{x}$, MA activation $y_m$
**Initialization**: $\xi$, *minAIservice*, $\mathcal{B}$, $m$, *tempAssignment*, $Lon_m^{cur}$, $Lat_m^{cur}$, $\hat{\omega}$

1: **while** $\forall k \in \mathcal{K}$ **do**
2:  **if** ($k$ is checked) **then**
3:  Continue to $k + 1 \in \mathcal{K}$
4:  **else**
5:  **for** $\forall b \in \mathcal{B}$ **do**
6:  **Evaluate constraint:** (12e)
7:  **if** ($\sum_{b\in\mathcal{B}} \sum_{k\in\mathcal{K}} x_{bk} \leq K$) **then**
8:  **Evaluate constraints:** (12a), (12b), and (12c)
9:  *tempAssignment* $\leftarrow$ evaluate $\xi$
10:  **if** (*tempAssignment* $<$ *minAIservice*) **then**
11:  $x_{bk} \leftarrow 0$
12:  $k$ is assign to $m$
13:  **else**
14:  $x_{bk} \leftarrow 1$
15:  $k$ is assign to $b \in \mathcal{B}\backslash\{0\}$
16:  **end if**
17:  **else**
18:  *break*
19:  **end if**
20:  **end for**
21:  **end if**
22: **end while**
23: **Evaluate constraint:** (12d)
24: **if** $\omega \sum_{k\in\mathcal{K}}(1 - x_{bk}) \leq \hat{\omega}$ **then**
25:  **Calculate:** $d_m$ using (6)
26:  Activate mobile agent: $y_m \leftarrow 1$
27: **else**
28:  $y_m \leftarrow 0$
29: **end if**
30: **return** $\mathbf{x}$, $y_m$

TABLE I
SUMMARY OF SIMULATION SETUP

| Simulation Parameters | Values |
|---|---|
| No. of SBSs $\lvert\mathcal{B}\rvert$ | [4, 10] |
| No. of Mobile agent $m$ | 1 |
| CPU capacity in edge server | 4 core with 1.2 GHz [20] |
| AI service sizes | [31,1546060] bytes [7] |
| No. of AI service requests $\lvert\mathcal{K}\rvert$ in MBS | [1,10000] [7] |
| Inter-class distance $\xi$ | $[0.3, 0.7]$ |
| Mobile agent height $h_m$ | 10 m [11] |
| Mobile agent velocity $v_m$ | 20 m/s [11] |
| Algorithm runs | 100 times |

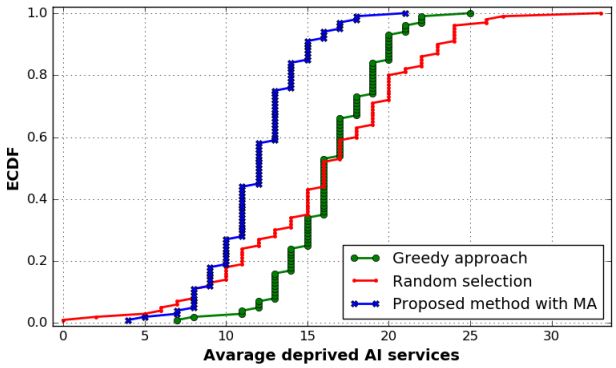


Fig. 3. ECDF of deprived AI services based on 100 time evaluation.

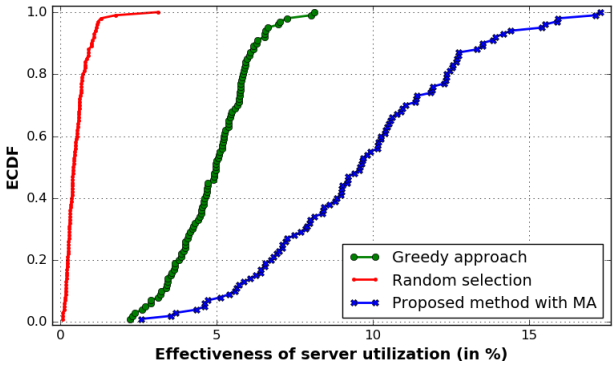


Fig. 4. ECDF for effectiveness of MEC server utilization based on 100 time evaluation.

decision in line 30.

The average case computational complexity of Algorithm 1 belongs to $\mathcal{O}(|\mathcal{K}|\log|\mathcal{K}|)$, whereas a worst case complexity is $\mathcal{O}(|\mathcal{K}|^2)$. Furthermore, space (memory) complexity of the proposed Algorithm 1 is $\mathcal{O}(|\mathcal{K}|)$ that is the size of AI service requests. The drawback of Algorithm 1 is controlling the distance parameter $\xi$ when data (input) dimension is very high. However, in the case of proposed AI service aggregation for mobile agent, this issue is considered to be neglected due to the low dimensional input space. Thus, the complexity of Algorithm 1 belongs to $\mathcal{O}(|\mathcal{K}|\log|\mathcal{K}|+|\mathcal{K}|)$. Additionally, the deprived AI services are admitted to a mobile agent using the concept of the outlier with this low complexity solution, which infers that the proposed AI service aggregation for mobile agent Algorithm 1 able to solve the proposed problem efficiently. The detail numerical analysis and insightful discussion of the proposed AI service aggregation model are given in the later section.

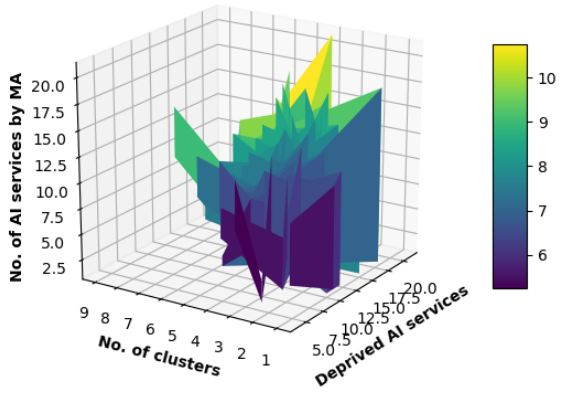


Fig. 5. Relationship between service fulfillment by mobile agent and no. of AI service clusters.

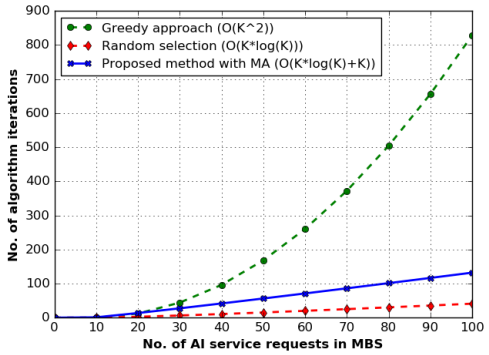


Fig. 6. Numerical analysis of computational time complexity.

## IV. NUMERICAL ANALYSIS AND DISCUSSION

In this section, we verify the performance of the proposed AI service aggregation model using numerical analysis and the important parameters of this setup is shown in Table I. Additionally, other network parameters are considered from the reference [20]. We implement this environment via Python, along with APIs and run the implemented model using a core i3 processor (3.9 GHz) with 16 GB of RAM as a mobile agent controller. To evaluate the proposed Algorithm 1 with a high degree of reliability, we consider greedy approach and random selection methods as baselines.

First, we present the (empirical cumulative distribution function) ECDF of the average deprived AI services in Fig. 3, where the proposed method (cross mark with a blue line) outperforms the greedy (circle mark with a green line) and random selection (dot mark with a red line) mechanisms. Thus, the proposed method significantly reduces the amount of deprived AI services, where this method fulfills $6\%$ and $9\%$ more AI services than the greedy and random selection methods, respectively. This illustrates that the proposed approach performs significantly better than the baseline methods.

Second, in Fig. 4, we discretized the ECDF for the effectiveness of MEC server utilization among three methods, in which the proposed Algorithm 1 efficiently utilize MEC servers with a higher performance gain (more than $9\%$) instead

of the greedy method since a data-driven approach discretized the characteristics of the AI service request. Further, the server utilization performance gaps of the random selection approach is far from the proposed model, in which the proposed method utilizes MEC server with $12\%$ more effectively than the random selection (in Fig. 4).

Third, we analyze a relationship among the number of cluster partitions, deprived AI services, and service fulfillment by a mobile agent, in Fig. 5. The number of deprived AI services are increased when the number of clusters is increasing due to the various type of AI service requests, in which the computational, memory and tolerable delay requirements are varying among them. Therefore, the mobile agent is capable of fulfilling these AI services with a higher degree of reliability, and efficiently increases the AI service fulfillment achieved rate of the network.

Finally, Fig. 6 illustrates the numerical complexity analysis of the proposed Algorithm 1 along with other two baseline models (greedy and random selection). Here, for an increasing number of AI service requests in the MEC network, the proposed AI service aggregation for mobile agent method takes a fewer iteration than the greedy approach, whereas the random selection method takes less iterations than the proposed model. However, a trade-off (no. of algorithm iterations) between proposed and random selection is compromised since the proposed method significantly reduces the deprived AI services and increases the effectiveness of MEC server utilization. Thus, the proposed method provides more efficient AI service aggregation with a low-cost solution.

## V. Conclusion

In this paper, we have introduced the artificial intelligence-as-a-service (AIaaS)-enabled edge computing model for accomplishing a vision of edge-AI, where we propose an AI service aggregation problem for mobile agent. We have increased the AI service fulfillment achieved rate under a large variety of AI service requirements, in which we consider a mobile agent as the decision maker. We have designed an optimization problem and the analogy shows this is an NP-hard problem. We have obtained the solution of this problem in a data-driven approach, where we propose an AI service aggregation algorithm for a mobile agent. To do this, we utilize the mechanism of DBSCAN along with the control flow algorithm. Our numerical analysis has established a significant performance gain with a higher degree of reliability for the proposed model than that the other baseline approaches, where the proposed scheme scaled up the AI service fulfillment achieved rate up to $9\%$. In the future, we will improve the scenarios into multiple mobile-agents using a more robust design mechanism.

## References


[1] L. Loven, E. Peltonen, T. Leppanen, J. Partala, E. Harjula, P. Porambage, M. Ylianttila, and J. Riekki, "EdgeAI: A Vision for Distributed, Edge-native Artificial Intelligence in Future 6G Networks," *6G Wireless Summit*, Levi, Finland, March 2019.

[2] W. Saad, M. Bennis, and M. Chen, "A Vision of 6G Wireless Systems: Applications, Trends, Technologies, and Open Research Problems," *arXiv:1902.10265 [cs.IT] (https://arxiv.org/abs/1902.10265)*, February 2019.

[3] E. Dahlman, S. Parkvall, J. Peisa, H. Tullberg, H. Murai and M. Fujioka, "Artificial Intelligence in Future Evolution of Mobile Communication," *International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, Okinawa, Japan, February 2019, pp. 102-106.

[4] Online: Artificial Intelligence as a Service (AIaaS) Market by Technology, Organization Size and Industry Vertical - Global Opportunity Analysis and Industry Forecast, 2017-2025, *Research and Markets (http://researchandmarkets.com/research/p7xbzs/the_worldwide?w=4)*, (Visited on 10 May, 2019).

[5] S. F. Abedin, M. G. R. Alam, N. H. Tran and C. S. Hong, "A Fog based system model for cooperative IoT node pairing using matching theory," *"2015 17th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, Busan, 2015, pp. 309-314.

[6] K. Thar, N. H. Tran, T. Z. Oo, and C. S. Hong, "DeepMEC: Mobile Edge Caching Using Deep Learning," *IEEE Access*, Vol.6, Issue 1, pp.78260-78275, December 2018.

[7] M. S. Munir, S. F. Abedin, N. H. Tran, and C. S. Hong, "When Edge Computing Meets Microgrid: A Deep Reinforcement Learning Approach," *in IEEE Internet of Things Journal*, Early Access, February 2019.

[8] M. S. Munir, S. F. Abedin, M. G. R. Alam, N. H. Tran and C. S. Hong, "Intelligent service fulfillment for software defined networks in smart city," *International Conference on Information Networking (ICOIN)*, Chiang Mai, Thailand, January 2018, pp. 516-521.

[9] N. H. Tran, W. Bao, A. Zomaya , M. N.H. Nguyen, and C. S. Hong "Federated Learning over Wireless Networks: Optimization Model Design and Analysis," *IEEE International Conference on Computer Communications (INFOCOM 2019)*, Paris, France, April 29 - May 2, 2019.

[10] M. G. R. Alam, R. Haw, S. S. Kim, M. A. K. Azad, S. F. Abedin, and C. S. Hong, "EM-Psychiatry: An Ambient Intelligent System for Psychiatric Emergency," *in IEEE Transactions on Industrial Informatics*,vol. 12, no. 6, pp. 2321-2330, Dec. 2016.

[11] J. Xiong, H. Guo and J. Liu, "Task Offloading in UAV-Aided Edge Computing: Bit Allocation and Trajectory Optimization," *Tin IEEE Communications Letters*, vol. 23, no. 3, pp. 538-541, March 2019.

[12] S. F. Abedin, M. G. R. Alam, A. K. Bairagi, A. Talukder, and C. S. Hong, "UAV-assisted Intelligent Crowdsourcing in Natural Calamity," *The International Symposium on Perception, Action, and Cognitive Systems (PACS 2016)*, Oct. 27-28, 2016, Seoul, Korea.

[13] S. F. Abedin, A. K. Bairagi, M. S. Munir, N. H. Tran, and C. S. Hong, "Fog Load Balancing for Massive Machine Type Communications: A Game and Transport Theoretic Approach," *in IEEE Access*, vol. 7, pp. 4204-4218, December 2018.

[14] M. Darabi, B. Maham, W. Saad, A. Mehbodniya and F. Adachi, "Context Aware Medium Access Control for Buffer-Aided Multichannel Cognitive Networks," *IEEE Globecom Workshops (GC Wkshps)*, San Diego, CA, December 2015, pp. 1-6.

[15] L. Kleinrock, "Queueing Systems," *New York:Wiley*, vol. I, 1975.

[16] S. F. Abedin, M. G. R. Alam, S. M. A. Kazmi, N. H. Tran, D. Niyato and C. S. Hong, "Resource Allocation for Ultra-reliable and Enhanced Mobile Broadband IoT Applications in Fog Network," *in IEEE Transactions on Communications*, vol. 67, no. 1, pp. 489-502, January 2019.

[17] S. Aaronson, "Guest column: NP-complete problems and physical reality," *ACM SIGACT News*, vol. 36, no. 1, pp. 30-52, March 2005.

[18] Z. Li'ang and G. Suyun, "The complexity of the 0/1 multi-knapsack problem," *Journal of Computer Science and Technology*, vol. 1, no. 1, pp. 46-50, March 1986.

[19] M. Sozio, and A. Gionis, "The community-search problem and how to plan a successful cocktail party," *In Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '10)*, pp. 939-948, 2010, ACM, New York, NY, USA.

[20] Y. Mao, J. Zhang and K. B. Letaief, "Joint Task Offloading Scheduling and Transmit Power Allocation for Mobile-Edge Computing Systems," *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, San Francisco, CA, 2017, pp. 1-6.