Mobility-Aware MEC Planning With a GNN-Based Graph Partitioning Framework

Jiayi Liu, *Member, IEEE*, Zhongyi Xu, Chen Wang, Xuefang Liu, Xuemei Xie, *Senior Member, IEEE*, and Guangming Shi, *Fellow, IEEE*

***Abstract*—Mobile service continuity is essential important to ensure that user sessions and services will survive user mobility. The 5G enhances its mobility management by providing the flexibility and offering three types of Session and Service Continuity (SSC) modes to address various service continuity requirements. Multi-access edge computing (MEC) is a type of widely adopted network architecture that delivers network services from the boundary of the mobile network by provisioning a set of edge servers. Determining an optimum planning of MEC edge servers, which involves determining edge servers appropriate geographical positions and their serving areas, is a precondition for more efficient service provisioning and better usage of network resources. In this work, we investigate the MEC servers planning problem by considering the management cost for maintaining MEC service continuity. The problem is formulated as a graph partitioning problem to partition the RAN graph with minimum SSC management costs and balanced MEC servers workloads. Then, we adapt a generalizable approximate Graph Partitioning framework which leverages on Graph Neural Network (GNN) to embed the RAN network spacial feature and on Multilayer Perceptron (MLP) for graph partitioning. Based on the framework, we propose a MEC server planning algorithm named MECP-GAP. Finally, we evaluate MECP-GAP with extensive simulations and real network data. Comparing to several baselines, MECP-GAP achieves better performance with lower running time.**

***Index Terms*—Mobile edge computing, edge server planning, mobility management, graph neural network, learning-based framework.**

I. Introduction

A MOBILE network is a complex wireless communication system that provides seamless connectivity and efficient communication for mobile devices. The mobile networking technology has been developed for decades, and it is playing an increasingly important role in the field of daily life and industrial production. Since the first-generation (1G) mobile network, mobile networking has evolved significantly and each generation represents significant technological improvements. For each generation of mobile network, mobility management is one of the fundamental functionalities that enables effective services delivers and maintains connections towards mobile users on the move [1]. The 3GPP 5G network also supports the mobility management functions with enhancements in user state management, handover and cell reselection, and Session and Service Continuity (SSC) [2]. The mobile data service is the fundamental service type in the 5G, hence service continuity has attracted more concern to ensure that user sessions will survive movements, and the 5G provides the flexibility by offering three types of SSC modes to address various service continuity requirements.

Multi-access edge computing (MEC) is a type of network architecture that delivers network services from the boundary of the mobile network by provisioning a set of geographically distributed edge servers at the edge of the network. MEC has been widely adopted to enhance the performance of various network services, such as mobile content delivery [3], smart city [4], Vehicle-to-Everything (V2X) applications [5], etc. For deploying an MEC system, determining an optimum planning of MEC edge servers is a fundamental precondition for more efficient service provisioning and better usage of network resources [6]. The planning of MEC edge servers refers to the determination of (1) the optimal locations for MEC edge servers, and (2) the appropriate MEC edge servers serving areas, which has attracted wide attention in recent years. These two problems are tightly coupled, and the serving area determination is essential, based on which the location determination problem can be reduced into finding the centroid within the serving area.

In [7], the authors introduce a framework to jointly optimize the placement of MEC servers and User Plane Functions (UPF) in a dynamic network environment. They formulate the problem as Integer Linear Programming (ILP) and utilize a scheduling technique based on Optimal Stopping Theory (OST) to decide the optimal reconfiguration time. In [8], the authors jointly consider edge server deployment and service placement to maximize the overall profit of the MEC system. They propose a two-step method including a clustering algorithm and nonlinear programming to solve the formulated problem. In [6], the authors formulate the MEC edge servers placement problem with dynamic uncertain resource demands

Manuscript received 10 April 2024; accepted 5 June 2024. Date of publication 11 June 2024; date of current version 21 August 2024. The associate editor coordinating the review of this article and approving it for publication was M. Shojafar. *(Corresponding author: Jiayi Liu.)*

Jiayi Liu is with the Guangzhou Institute of Technology, Xidian University, Guangzhou 510530, China, and also with Pazhou Laboratory (Huangpu), Guangzhou 510530, China. (e-mail: jyliu@xidian.edu.cn).

Zhongyi Xu and Xuefang Liu are with the School of Telecommunication, Xidian University, Xi'an 710071, China.

Chen Wang is with the Noah's Ark Lab, Huawei, Shenzhen 518129, China.

Xuemei Xie is with the School of Artificial Intelligence, Xidian University, Xi'an 710071, China, and also with Pazhou Laboratory (Huangpu), Guangzhou 510530, China.

Guangming Shi is with the School of Artificial Intelligence, Xidian University, Xi'an 710071, China, also with Pazhou Laboratory (Huangpu), Guangzhou 510530, China, and also with Peng Cheng Laboratory, Shenzhen 518066, China.

Digital Object Identifier 10.1109/TNSM.2024.3412959

through an uncertain programming formulation, and a deep-learning based algorithm is proposed to practically solve the problem. In [9], the authors formulate the edge server placement problem as a capacitated location allocation problem and provide an algorithm for effective solutions. In [10], the authors study the MEC server placement in smart cities. They formulate the problem as a multi-objective constraint optimization problem and adopt mixed integer programming to find the optimal solution.

However, MEC edge servers are provisioned to provide network services, and none of the above works consider the optimization of MEC servers planning in terms of minimizing the management costs in providing service continuity for moving users. Moreover, existing works are based on traditional model-based methods which formulate and solve the MEC server planning problem as an optimization problem. With the development of Artificial Intelligence (AI) technology, machine learning-based methods are widely applied in various domains, including network optimization [11]. Comparing to the traditional model-based optimization methods, machine learning are model-free and data-driven, which could explore the intrinsic inner relationships between complex system parameters and possible to discover better solutions. Hence, in this current work, *we aim to investigate the MEC servers planning problem by considering the management cost for maintaining MEC service continuity with machine learning-based methods*. Specifically, we focus on the MEC edge server serving area determination problem. Because upon the determined serving area of each MEC server, the optimal location can be further decided trivially by finding the centroid within the serving area.

By considering the underlying Radio Access Network (RAN) in a certain region as a graph, the MEC servers planning problem is formulated as a graph partitioning problem such that node weights represent Base Stations (BSs) average workloads and edge weights represent users handover management costs in service data delivery. Hence, partitioning the RAN graph with minimum edge cuts and balanced cluster node weights refers to the optimum MEC servers planning with minimum SSC management costs and balanced MEC servers workloads. To solve this formulated problem, we adapt a generalizable approximate Graph Partitioning framework proposed in [12] which leverages on Graph Neural Network (GNN) to embed the RAN network spacial features and on Multilayer Perceptron (MLP) for graph partitioning. We define the loss function to generate partitions with balanced node weight and minimum edge cut. Based on this generalizable approximate partitioning (GAP) framework, we design an MEC server Planning algorithm, named as MECP-GAP, to determine the optimum MEC servers planning from an input RAN and users movements to minimize SSC management costs with balanced MEC servers workloads. Moreover, the framework is generalizable, once trained it can produce efficient partitions results on unseen graphs at inference time. Finally, we evaluate MECP-GAP with extensive simulations and real network data. Comparing to two traditional algorithms for graph partitioning problem, MECP-GAP achieves better performance with lower running time. Our contributions are summarized as follows:

- Firstly, we investigate the MEC servers planning problem to minimize the management cost for maintaining service continuity for moving users. We formulate this problem as a graph partitioning problem with minimum edge cuts and balanced cluster node weights.
- Then, we adapt a learning-based framework to solve the above problem. By integrating GNN and MLP, the framework is generalizable for producing solutions for various types of unseen graphs at inference time. This is an advantage over the traditional model-based methods which must reperform the optimization problem for every new use case.

The rest of the paper is organized as follows. Related works are summarized in Section II. Then, Section III formulates the problem of mobility-aware MEC planning through graph partitioning. Then, we detail the MECP-GAP algorithm in Section IV. In Section V, we evaluate the performance gain of MECP-GAP over several baselines. Finally, we conclude this work in Section VI.

II. Related Works

A. Mobility Management

Mobile network planning optimization with respect to mobility management is an important issue and has been intensively studied in the literature. From the GSM system, a group of cells is called Location Area (LA) to facilitate paging. The planning of LAs aims to balance the tradeoff between two conflicting metrics: the amount of paging messages and registration signalling. In [13], the authors design the layout of LAs to balance this tradeoff for the GSM system. From the LTE system, Tracking Area (TA) is utilized to group cells and to locate mobile devices. In [14], the authors determine the TAs for the LTE network by applying multi-objective optimization for achieving a balanced tradeoff between paging overhead and tracking area update, and they design an evolutionary multi-objective algorithm based on a population decomposition strategy. In [15], the authors propose an TA list management framework to format the optimal TA distribution in the form of TA lists and determine the association from TA lists towards mobile devices for the 5G system. Besides, data delivery is now the fundamental service types in mobile network, and service continuity is essential important to ensure that user sessions and services will survive user mobility. In view of this, 5G enhances its mobility management by providing three types of SSC modes to address various service continuity requirements. In [16], the authors analyze mobility-aware cellular network performance with several mobility patterns and network topologies. They also point out that mobility would impact on service delivery and outline the challenges for mobility-aware analysis. Hence, the 5G network planning optimization should also consider to optimize the structure and layout of the network for minimizing the management cost in maintaining service continuity. In this work, we aim to study the MEC servers planning problem by optimizing the management cost for maintaining MEC service continuity.

In [17], the authors consider the issue of managing high-speed mobility in 5G networks. To solve this problem,

the authors propose an online learn-based mechanism that uses Kalman filters to compute the posterior value of the RSRP values of service units and adjacent units to determine the best target unit for switching, so as to maintain high performance under extremely high mobility after migration. Simulations show that this mechanism can significantly improve handover execution in 5G, enabling intelligent handling of high-speed mobile management in 5G and beyond. In [18], a novel two-layer cooperative satellite network architecture is proposed. Based on the proposed two-layer cooperative satellite network architecture, four mobility management scenarios in satellite-ground communication scenarios are studied. For each scenario, the corresponding switching mode is proposed to ensure the continuity of terminal services, and the corresponding signaling overhead is analyzed. Simulation results show that the proposed switching mode can greatly reduce the signaling overhead. In [19], the authors examine the challenges facing emerging and future cellular Networks such as Ultra-Dense Multi-Band Networks (UDMNs), including Resource-Inefficiency and Scarcity, User-Experience Degradation and Unprecedented Channel Behavior in mmWave Cells. The author proposes an Advanced Mobility Management and Utilization based on machine-learning-based mobility and traffic prediction models Framework. The framework is capable of predicting mobility patterns, next candidate cells for handover, and handover time, as well as future cell loads.

B. MEC Planning

Determining an optimum planning of MEC edge servers is a fundamental precondition for deploying an MEC system, which has attracted wide attention in recent years. Most existing works are traditional model-based methods which formulate the MEC planning problem as an optimization problem. Due to the complexity of the problem, heuristic algorithms are normally proposed. In [7], the authors jointly optimize the placement of MEC servers and UPF through ILP in a dynamic network environment. In [8], the authors jointly consider edge server deployment and service placement to maximize the overall profit of the MEC system. They propose a two-step method including a clustering algorithm and nonlinear programming to solve the formulated problem. In [6], the authors formulate the MEC edge servers placement problem with dynamic uncertain resource demands through an uncertain programming formulation, and a learning based algorithm is proposed to practically solve the problem. In their algorithm, meta-heuristic is utilized to explore the solution space, with MLP utilized to perform a fast prediction for the fitness function. In [9], the authors formulate the edge server placement problem as a capacitated location allocation problem and provide an heuristic algorithm for effective solutions. In [10], the authors study the MEC server placement in smart cities. They formulate the problem as a multi-objective constraint optimization problem and adopt mixed integer programming to find the optimal solution. For mobility-aware MEC network resource allocation, in [20], the authors study the edge server user association problem in dynamic environment where edge user high mobility is considered. The problem is formulated as an online decision-making and evolvable process for allocating users at real-time. In [21], the authors propose a mobility-aware task offloading scheme for vehicular MEC network, and they develop a Deep Reinforcement Learning (DRL)-based algorithm to train the offloading strategy.

C. Graph Partitioning Problem

We formulate the MEC planning problem as a graph partitioning problem, where BS nodes are partitioned into clusters and each cluster is served by one MEC server. The graph partitioning problem has been extensively studied in the literature. Hence, we also survey related works in this topic. Graph partitioning is the problem of dividing the nodes of a graph into balanced partitions while minimizing the edge cut across the partitions. Graph partitioning is a well studied optimization problem and has a wide application in various domains. Traditional well-known algorithms include the Breadth First Search (BFS) algorithm, the Kernighan-Lin (KL) algorithm, the spectral method based on eigen-vector decomposition of the graph Laplacian matrix. From the above methods, the METIS algorithm [22] works very well in practice and is largely adopted. Moreover, with the development of the AI technology, there are also works that investigate solving the graph partitioning problem through machine learning-based methods. In [23], by combining the simulated-annealing algorithm and the Hopfield neural network, the authors first propose an learning based algorithm for solving the graph partitioning problem. In [12], the authors propose a Generalizable Approximate Partitioning framework, named as the GAP framework. The framework contains two parts: a graph embedding part that leverages on Graph Neural Network (GNN) to embed the graph spacial feature; and a graph partitioning part that leverages on Multilayer Perceptron (MLP) for graph partitioning. In [24], a theoretical performance analysis of the GNN is presented. A mean-field theory of a minimal GNN architecture is developed for the graph partitioning problem. Numerical experiment results shows GNN's accuracy in performing classification tasks.

D. Graph Neural Network

Graph neural networks (GNNs) are a set of deep learning methods that treat graph structured data in non-Euclidean space. GNN has been intensively studied and applied in multiple domains including: combinatorial optimization [12], computer vision [25], and network management [26]. The idea of GNN was first introduced in [27], the authors extend existing neural network methods for processing data represented in graphical form. The general idea of GNN is to learn a state embedding by exploring topological structure of the graph and aggregating information from the neighbors of each node. This embedding is then further utilized for downstream tasks, such as graph-level, node-level or edge level tasks. There are several types of GNNs. The Graph Convolutional Networks (GCNs) [28] utilize the convolution operation as

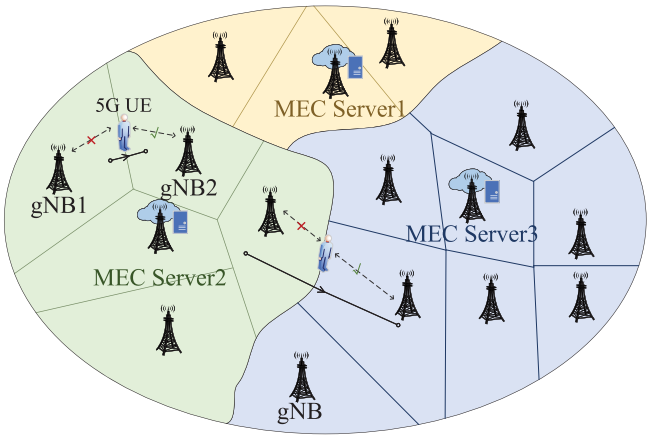

Fig. 1. 5G MEC data network and MEC planning.

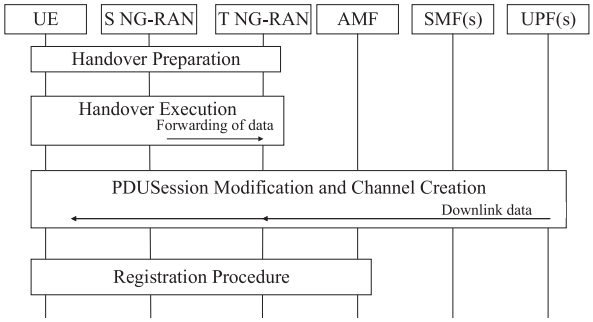

Fig. 2. SSC mode 1 handover procedure between gNBs.

defined in the Fourier domain by computing the eigen decomposition of the graph Laplacian. The Message Passing Neural Networks (MPNNs) consists of message, update, and readout functions which operate on nodes and edges in the graph [29]. The Graph Attention Networks (GAT) applies the attention mechanism to aggregate information from node neighbors through a weighted sum [30]. The GAP framework [12] is designed to solve graph partitioning problem by using graph embedding and optimizing the partitioning loss function. In this work, we leverage on GAP to solve the mobility-aware MEC planning problem, which is formulated as a graph partitioning problem. We design the input and the loss function to adjust the algorithm into our case. Simulation results shows that our improved framework and algorithm achieves better planning result with lower running time by comparing to several traditional baselines.

III. System Model

In this section, we formulate the mobility-aware MEC edge servers planning problem, which takes into account the management cost for SSC continuity. The handover cost of users is formulated based on 3GPP technical specifications. Finally, the problem is formulated as a graph-partitioning problem with minimum edge cut and balanced node weight.

A. Problem Description

An overview of the 5G MEC data network and the corresponding MEC planning is shown in Fig. 1. Three MEC servers are deployed to serve mobile users from multiple gNBs. Each MEC server is equipped with resources to serve users from several certain gNBs, and these gNBs are the serving area of the MEC server. By applying the Uplink Classifier (UL CL) technology and configuring UPFs, traffic from these gNBs can be directed to their serving MEC servers. In the case of PDU Sessions of type IPv4 or IPv6 or IPv4v6 or Ethernet, the SMF may decide to insert in the data path of a PDU Session an "UL CL" (Uplink classifier). The UL CL is a functionality supported by an UPF that aims at diverting (locally) some traffic matching traffic filters provided by the SMF [2]. The insertion and removal of an UL CL is decided by the SMF and controlled by the SMF using generic N4 and UPF capabilities. The UL CL can be configured with the traffic rules to forward the uplink traffic towards different targeted applications and network functions, and in the downlink direction it will merge the traffic destined to the UEs [31].

Mobile users are free to move, consequently, handovers can happen in different levels: gNBs level and MEC servers level. For a moving user, when she crosses the boundaries of gNBs, the user's PDU session, which starts from the user and ends in the MEC server, should take a handover from gNB1 to gNB2, incurring signaling costs. The handover delay also leads to deteriorated user Quality of Experience (QoE). Furthermore, if the user moves in a large geographical scale which extends beyond the boundaries of MEC server serving area, this would results in higher handover signaling cost and increased transmission delay. Therefore, the objective of mobility-aware MEC planning is to determine the serving areas of each MEC server to minimize the handover costs for mobile users.

B. Mobility Management Cost

Then, we quantify the handover management costs in multiple different levels. In this paper, two types of handover procedures are considered: the first one is between gNBs but within the same MEC serving area, and the second is between MEC servers. The first one is also referred as SSC mode 1. SSC mode 1 is defined in detail in the 3GPP technical specification TS 23.502 [32], which comprehensively defines various procedures within the 5G system. Following the specification, we provide a summary of the SSC mode 1 handover procedure between gNBs, as shown in Fig. 2, with detailed procedures outlined in Chapter 4.9.1.2 of TS 23.502.

This handover process comprises handover preparation, handover execution, PDU Session modification, channel creation and registration procedure. Following these procedures, the UE PDU Session service undergoes a handover from the source-gNB (noted as Source NG-RAN in the figure) to the target-gNB (Target NG-RAN), controlled by 5G core network functions including AMF, SMF, and UPF. Consequently, we calculate the cost c_g for handovers between gNBs within the same MEC serving area in SSC mode 1 as follows:

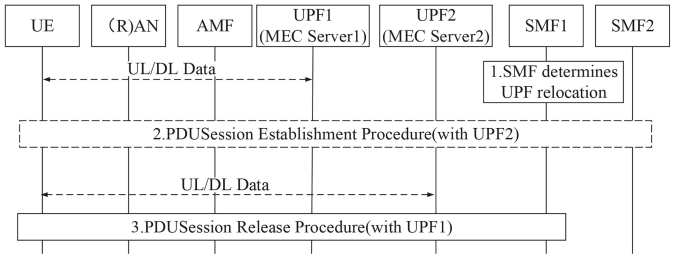

Fig. 3. SSC mode 3 handover procedure between MEC servers.

$$\begin{aligned} c_g &= T_{HP} + T_{HE} + T_{PMCC} + T_{RP} \\ &= 2T_{N2} + T_{N3} + 2T_{N4} + 2T_{N11} \\ &\quad + 5T_{Xn} + 3T_{UE-RAN} \\ &\quad + T_{handover\_decision} + T_{admission\_control}, \end{aligned}$$

where T_{N2}, T_{N3}, T_{N4}, and T_{N11} represent the time delays between core network functions in the corresponding interface, whilst T_{Xn} and T_{UE-RAN} denote the time delays between access network interfaces. Specially, $T_{handover\_decision}$ and $T_{admission\_control}$ stand for the time consumed by network functions in decision-making processes. Considering that the time consumption for these functions is significantly lower than the others, we neglect them in the following part.

Then, we also provide an overview of the handover procedure between MEC servers in Fig. 3. The handover between MEC servers corresponds to SSC mode 2 and mode 3, as defined in Chapter 4.3.5.2 of TS 23.502. For improved user QoE, in this work, we consider that the handover follows SSC mode 3 due to its lower service interruption delay. Then, as depicted in Fig. 3, this procedure comprises three primary steps. Initially, SMF1 determines UPF relocation based on the UE's mobility. Then, SMF2 establishes a new PDU session from MEC server2 to the UE. Finally, SMF1 releases the old PDU session from MEC server1 to the UE. Let c_M denotes the handover cost between MEC servers, which can be calculated as:

$$\begin{aligned} c_M &= T_{determine} + T_{PRP} + T_{PEP} \\ &= 3T_{N1} + 2T_{N2} + T_{N3} + 13T_{N4} + 4T_{N6} + 6T_{N7} \\ &\quad + 4T_{N10} + 10T_{N11} + 3T_{UE-RAN} + T_{SMF\_Selection} \\ &\quad + T_{PCF\_Selection} + 2T_{UPF\_Selection} + T_{Trigger}. \end{aligned}$$

where T_{N1}, T_{N2}, T_{N3}, T_{N4}, T_{N6}, T_{N7}, T_{N10}, and T_{N11} represent the time delay between the corresponding core network functions through these interfaces, and T_{UE-RAN} denotes the time delay from access network interfaces. Additionally, $T_{SMF\_Selection}$, $T_{PCF\_Selection}$, $T_{UPF\_Selection}$, and $T_{Trigger}$ denote the time consumed by network function decision-making processes. These decision-making processes are described in detail in clause 4.3.2.2.1 of TS 23.502. Each decision-making process in c_M is a complex process to make decisions according to the actual situation, so the decision delay of c_M is much larger than c_g's.

C. Graph-Partitioning Problem Formulation

Let $G = (V, E)$ represents a 5G RAN network graph, where the set $V = \{v_i\}$ corresponds to the 5G gNBs, and the set $E = \{e(v_i, v_j) | v_i \in V, v_j \in V\}$ represents the handover behaviors between gNBs. Then, the weight of $e(v_i, v_j)$ is denoted as w_{ij}, which quantifies the average number of users moving from gNB v_i to gNB v_j. Moreover, we consider the graph G as an undirected graph, because the handover behavior is symmetric between two gNBs. We assume that there are N gNBs in total, and all gNBs are served by several MEC Server. The total number of MEC Servers is denoted as P. Consequently, the entire RAN network should be divided into P partitions, defined as $S_1, S_2, \ldots, S_P$. Hence, the union of all partitions is equal to V: $\cup_{k=1}^{P} S_k = V$, and each gNB exclusively belongs to one partition: $S_k \cap S'_k = \emptyset \ \forall k,\ k'$. The number of gNBs in partition S_k is N_k, satisfying $\sum_{K=1}^{P} N_k = N$.

As defined in Section III-B, the handover cost between gNBs within the same MEC serving area is denoted as c_g, whilst the handover cost between gNBs from different MEC serving area is denoted as c_M. Based on the descriptions provided for each interface in 3GPP TS23.501, it is obvious that $c_M > c_g$.

We introduce the decision variable x_{ik} to show the dependency relationship between gNB v_i and partition S_k, where $x_{ik} = 1$ means that gNB v_i belongs to partition S_k, and $x_{ik} = 0$ indicates the opposite. Our objective is to minimize user handover costs by optimizing the distribution of MEC servers under load balancing conditions. Therefore, the objective function is expressed as follows

$$\min \quad \alpha \cdot Edge\_cut + \beta \cdot Load\_balancing, \tag{1}$$

where

$$\begin{aligned} Load\_balancing &= \sum_{k=1}^{P} \left(N_k - \frac{N}{P} \right)^2 \\ &= \sum_{k=1}^{P} \left(\sum_{i=1}^{N} x_{ik} - \frac{N}{P} \right)^2, \end{aligned} \tag{2}$$

$$\begin{aligned} Edge\_cut &= c_M \cdot \sum_{k=1}^{P} \sum_{v_i \in S_k, v_j \in \overline{S_k}} e(v_i, v_j) \cdot w_{ij} \\ &\quad + c_g \cdot \sum_{k=1}^{P} \sum_{v_i, v_j \in S_k} e(v_i, v_j) \cdot w_{ij} \\ &= c_M \cdot l_M + c_g \cdot l_g \\ &= c_M \cdot l_M + c_g \cdot (l_{sum} - l_M) \\ &= (c_M - c_g) \cdot l_M + c_g \cdot l_{sum}, \end{aligned} \tag{3}$$

α, β are positive normalized parameters. In Eq. (3), we set

$$\begin{aligned} l_M &= \sum_{v_i \in S_k, v_j \in \overline{S_k}} e(v_i, v_j) \cdot w_{ij}, \\ l_g &= \sum_{v_i, v_j \in S_k} e(v_i, v_j) \cdot w_{ij}, \\ l_{sum} &= l_M + l_g, \end{aligned}$$

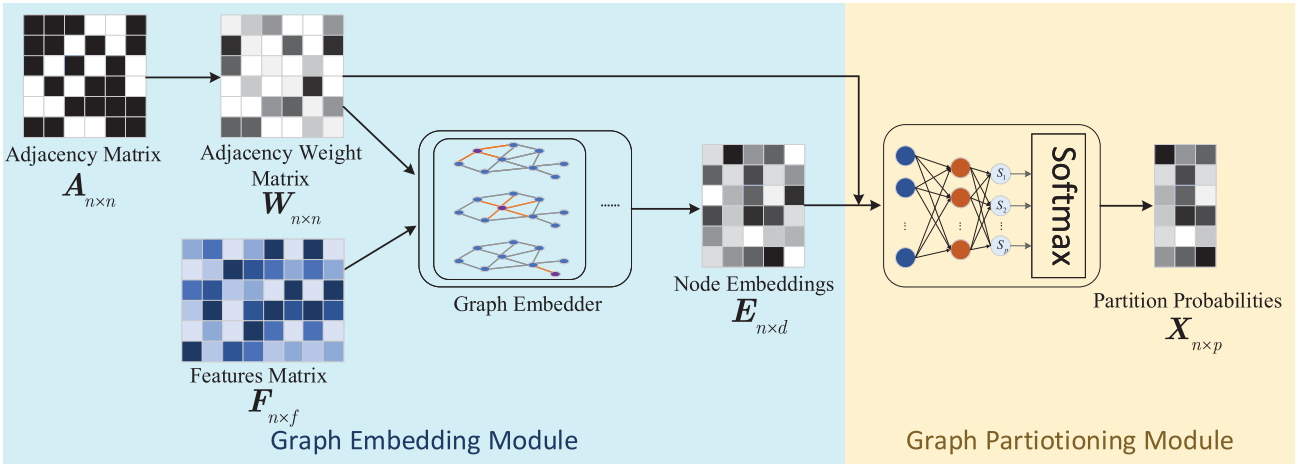

Fig. 4. The I-GAP framework.

where $\overline{S_k}$ is the complement of S_k in V. The constraints of the problem are as follows:

$$\sum_{k=1}^{P} x_{ik} = 1 \ \forall v_i \in V, \tag{4}$$

$$\sum_{k=1}^{P}\sum_{i=1}^{N} x_{ik} = N. \tag{5}$$

From Eq. (1) to (3), our optimization objective is to minimize the sum of user mobility handover cost and network load balancing, with α and β are used for balancing the optimization direction. Eq. (2) represents the variance of the number of gNBs within different partitions. For the layout and planning of gNBs, they are deployed to serve users by considering the geographical population density. Hence, the number of gNBs represents the workloads for MEC servers. In Eq. (3), we split the user handover cost into two parts: one is the handover cost resulting from movement between different partitions, and the other is the handover cost resulting from movement between gNBs within the same partition. Furthermore, Constraint (4) ensures that a gNB can only belong to one partition, and Constraint (5) implies that each gNB should be assigned to one partition.

In Eq. (3), when $c_M > c_g$ is ensured, the partition results $S_1, S_2, \ldots, S_P$ remain unaffected by the specific values of c_M and c_g. Therefore, we set $c_M = 1$, $c_g = 0$, and simplify Eq. (3) into the following form. In this case, $Edge\_cut$ is defined to minimize the weights of edges between different partitions.

$$Edge\_cut = l_M = \sum_{k=1}^{P} \sum_{v_i \in S_k, v_j \in \overline{S_k}} e(v_i, v_j) \cdot w_{ij}, \tag{6}$$

Based on Eq. (2) and (6), the problem of mobility-aware MEC server planning is formulated into a graph partitioning problem. The objective is to determine an optimum balanced partitions of the 5G RAN network to minimize the sum weights of edge cuts between partitions.

IV. Algorithm

In this section, we adapt the GAP framework [12] and design the MECP-GAP algorithm for mobility-aware MEC planning. Firstly, we introduce the graph learning-based GAP framework to tackle the Graph Partitioning problem. Then, the structure of our improved GAP framework is outlined, and the design of the loss function is described in detail. Subsequently, we develop the MECP-GAP algorithm based on deep learning to solve the MEC server planning problem. Finally, the algorithm's complexity is analyzed.

A. Problem Definition

In Section III, we formulate the mobility-aware MEC server planning problem as a Graph Partitioning problem. Many traditional algorithms have been proposed for solving the Graph Partitioning problem [33]. For instance, the Metis algorithm is a powerful graph partitioning software package known for its efficiency [22]. Recently, deep learning is also applied in solving optimization problems. The Graph Partitioning problem is also investigated to be solved through learning methods. The GAP framework, proposed in [12], utilizes the GNN for embedding graph nodes features and MLP for partitioning nodes. Comparing to traditional algorithms, the Learning-based GAP algorithm has made significant improvements in optimization objectives and algorithm runtime. We improved this framework (noted as I-GAP framework) to adapt it to solve the above MEC servers planning problem. The I-GAP partitioning framework is illustrated in Fig. 4.

1) Loss Function: Before introducing the I-GAP framework model, we first present the model's loss function. Our objective is to obtain the partition result $\mathbf{X}_{n\times p}$ based on gNB features $\mathbf{F}_{n\times f}$ and adjacency weights $\mathbf{W}_{n\times n}$. The partition result is represented as a $n \times p$ matrix $\mathbf{X}_{n\times p}$. In this matrix, the element $x_{ik} = 1$ indicates that gNB v_i belongs to partition S_k, while $x_{ik} = 0$ represents the opposite. Thus, we can reformulate Eq. (6) as follows

$$E[edge\_cut] = \sum_{reduce-sum} \mathbf{X} \cdot (\mathbf{1} - \mathbf{X})^T \odot \mathbf{W}, \tag{7}$$

where $\odot$ represents the element-wise product. $\sum_{reduce-sum}$ refers to the operation of summing the elements in an matrix, adding all elements together to obtain a single cumulative total. This is a common data processing and computational operation. The matrix $\mathbf{W}_{n\times n}$ is the adjacency weight matrix, where each element w_{ij} corresponds to the number of users moving from gNB v_i to gNB v_j. Similarly, we can express Eq. (2) in matrix form as follows

$$E[load\_balancing] = \sum_{reduce-sum} \left(\mathbf{1}^T\mathbf{X} - \frac{N}{P}\right)^2. \quad (8)$$

Combining Eq. (7) and Eq. (8), the loss function of I-GAP model can be expressed as follows

$$L = \alpha \cdot \sum_{reduce-sum} \mathbf{X} \cdot (\mathbf{1} - \mathbf{X})^T \odot \mathbf{W} + \beta \cdot \sum_{reduce-sum} \left(\mathbf{1}^T\mathbf{X} - \frac{N}{P}\right)^2, \quad (9)$$

where α and β are used for normalization. They are configured to ensure that the values of edge cut and load balancing are within the same range.

2) I-GAP Framework: The I-GAP model comprises two main modules: the graph embedding module and the graph partitioning module. The former focuses on learning node embeddings using both graph structure and node features. In I-GAP, the graphSAGE [34] model is employed to embed node representations, which emphasizes sampling and aggregation. As an inductive learning method, graphSAGE not only learns the embedding representations of each node in the graph, but also learns the aggregation function used to sample and aggregate neighbor nodes to generate new embedding representations of node [34]. Unlike other transductive methods which only perform representations learning on a fixed graph, graphSAGE has good inductive ability and does not need to know the information of all nodes when implementing graph embedding, which is suitable for large-scale graphs and unseen graphs [35].

In our I-GAP framework, we replace the adjacency matrix with the adjacency weight matrix to improve the sampling process of graphSAGE. Each node sample among its neighbors to aggregate their representations. This process is shown in Eq. (10):

$$\mathbf{h}^k_{N(v)} \leftarrow AGGREGATE_k\left(\{\mathbf{h}^{k-1}_u,\ \forall u \in N(v)\}\right), \quad (10)$$

where $\mathbf{h}^k_{N(v)}$ represents the embedded representation of node v at the current step k. $\mathbf{h}^{k-1}_u$ represents the embedded representation of node u at step $k-1$, and $N(v)$ represents all neighbors of node v. In the original graphSAGE model, neighbor sampling follows the uniform distribution. This means nodes take fixed samples from their neighbors in a completely random manner. The process is shown in Eq. (11):

$$u \leftarrow UNIFORM(N(v)). \quad (11)$$

The improvement of graphSAGE lies in the improvement of the random sampling process. Specifically, suppose node v and node u are adjacent, and the edge weight between them is w_{vu}. Then the probability that node u is sampled as the neighbor of node v is:

$$P(u|v) = \frac{w_{vu}}{\sum_{i\in N(v)} w_{vi}}, \quad (12)$$

where $\sum_{i\in N(v)} w_{vi}$ represents the sum of adjacent edge weights from node v to all its neighbors. Then in our graphSAGE model, the sampling process can be described as:

$$u \leftarrow u\ with\ probability\ P(u|v). \quad (13)$$

As can be seen from Eq. (13), the larger the edge weight between two nodes, the easier it is sampled and aggregated together. After graph aggregation, their feature vectors are more similar, so they are more easily to be divided into the same partition after graph partitioning module, which is also consistent with the problem to be solved in this paper.

The graph partitioning module consists of a fully connected network followed by softmax and is trained to minimize the loss function Eq. (9). Because the graph partitioning module actually completes a classification task. That is, according to the node embedding representation generated by the graph embedding module, the gNBs is classified into different partitions. As a flexible model architecture, fully connected network can be applied to various classification problems and has wide applicability in various scenarios [36].

B. MECP-GAP Algorithm

In this section, the MECP-GAP algorithm is proposed. The primary steps of this algorithm are outlined in **Algorithm 1**.

It is emphasized that **Algorithm 1** operates in parallel with network training, concurrently producing the network model and partition results. Consequently, the primary factor influencing the time complexity is the number of training epoch, denoted as e_1. Let N represent the number of gNBs. In the sampling process, as we set a fixed number of sampling, the time complexity of the sampling process is $O(1)$. In the aggregation process, the mean representation of the neighbor nodes needs to be calculated. Assuming that there are n fixed samples in the neighbors and the data feature dimension is m, the upper limit of the time complexity of the aggregation process is $O(n * m)$. As matrix operations are used in the actual calculation, the actual complexity will be less than this value.

In the classification process, the classification network needs to be trained. Specifically, the training epoch is set as e_2, b represents the number of batches, and the time complexity of calculating the loss function is related to batch size, denoted as $O(s)$. We ignore the time which takes to update the network parameters. So, the time complexity of Algorithm 1 is $O(e_1(N(1 + n * m) + e_2 * b * s))$, simplifies to $O(e_1N + e_1Nnm + e_1e_2bs)$.

C. Practical Relevance

The proposed MECP-GAP algorithm solves the problem of MEC server deployment for mobility management. Since the MEC server needs sufficient hardware support, the MEC server in our scenario will not move once deployed, which is

Algorithm 1 MECP-GAP Algorithm

Input: learning rate η; normalization parameters α, β; epoch number N_{epoch}; feature matrix $\mathbf{F}_{n\times f}$; adjacency weight matrix $\mathbf{W}_{n\times n}$.

Output: partition probabilities matrix $\mathbf{X}_{n\times p}$.

Initialize the fully network parameters and $\mathbf{X}_{n\times p}$;
Divide the nodes into train nodes v_{train} and test nodes v_{test};
for $epoch1 = 1, ..., e_1$ **do**
  Divide v_{train} into batches.
  for v_i in v_{train} **do**
    $f_i \leftarrow$ sample neighbors according to Eq. (13) and aggregate $f_{Neigh(i)}$;
    Normalize f_i;
  end for
  for $epoch2 = 1, ..., e_2$ **do**
    for batch in range(batches) **do**
      $\mathbf{X}$ = classification($\mathbf{F}_{batch}$);
      L = calculate($\mathbf{X}$) with $\mathbf{W}$ using v_{batch};
      Update classification network parameters using gradient descent based on L;
    end for
  end for
  Evaluate the network using v_{test} and update graphSAGE network parameters;
end for
return $\mathbf{X}_{n\times p}$, I-GAP network model.

also consistent with the reality. First, regarding the adaptability of the algorithm, based on the generalization ability of the I-GAP graph neural network framework, the network parameters obtained by training on a large graph can be directly used to partition the new graph (network) to deploy the MEC server. Therefore, MECP-GAP algorithm has good adaptability. MECP-GAP can even deal with unseen graphs, so it also has good robustness. Regarding the scalability of the algorithm, the MEC algorithm is suitable for small-scale networks (hundreds of gNBs) and large-scale networks (thousands of gNBs). When facing the challenge of hyper-scale network (millions of gNBs), network operators can manually split the network into many general scale networks. Then the MECP-GAP algorithm is applied to the general scale networks.

V. Simulation

We conduct intensive simulations to evaluate our proposed MECP-GAP algorithm. Two sets of simulations are conducted, the first is a small scale network simulation to illustrate the partitioning results more clearly. The second is a large scale network simulation with real gNBs dataset. Comparing to two widely used traditional graph partitioning algorithms, MECP-GAP achieves better partitioning results with lower running time.

A. Simulation Setting

In the small scale simulation, there are 200 gNBs, and their geographical locations are generated using a Poisson Point Process. For the large scale simulation, we utilize the location coordinates of 2170 gNBs located in Shanghai, China, from the official website of Opencellid [37]. Opencellid is an open-source platform that provides location coordinates for gNBs and eNBs across the globe.

The adjacency weight matrix, denoted as $\mathbf{W}$, incorporates information about both the adjacency relationship and the number of handovers between gNBs. In our simulation, we utilize the human movement model proposed by Liang et al. [38] to generate the values in $\mathbf{W}$. This model is capable of estimating the volume of population movements between two regions based on the distance separating those regions and the sizes of the regions. It is a well-established human movement model commonly used in sociology, which enhances the realism of our simulation results. To generate $\mathbf{W}$ in our simulation, we first create a Voronoi graph based on the geographical locations of the gNBs. The Voronoi graph delineates the service areas for each gNB. Subsequently, we input the gNB location information and the associated area information into the human movement model to calculate the number of handover users moving between gNBs and obtain the adjacency weight matrix $\mathbf{W}$.

The I-GAP model comprises two primary modules: the graph embedding module and the graph partitioning module. In the graph embedding module, we employ the graphSAGE model, which is a 2-layer graph convolutional network. It takes input features with a dimensionality of 200 and outputs embeddings of dimensionality 128. The gNB's geographical location is utilized as its feature for both small scale and large scale simulations. To aggregate neighboring node representations in graphSAGE, we employ the mean aggregator. Compared with LSTM aggregator and pooling aggregator, the mean aggregator are more suitable for scenarios where the overall information of neighbor nodes is retained. This method is often used in tasks such as node classification and graph partitioning. In the graph partitioning module, we utilize a fully connected neural network to transform the 128-dimensional input matrix into the desired number of partitions. The partition result $\mathbf{X}$ is obtained using the softmax activation.

B. Baseline Algorithms

The greedy algorithm: in the work of Taleb and Ksentini [33], they introduced a pseudo-greedy algorithm to address the graph partitioning problem. This algorithm initiates by randomly selecting a node v_i from the node set to serve as the center of partition S_k. It then iterates through all neighbors of v_i, evaluating whether each neighbor satisfies certain constraint conditions. If a neighbor meets these conditions, it is added to partition S_k. This process continues until no more nodes can be added to the partition. This algorithm is a classical approach for graph partitioning, and its performance typically improves with an increased number of iterations.

The Metis algorithm [22]: is a widely used graph partitioning algorithm known for its high efficiency. It serves as a common baseline for graph partitioning algorithms. This algorithm requires input data regarding node and link information, as well as the desired number of partitions. It is

capable of swiftly producing partition results, which makes it a suitable choice for our baseline comparison.

PBPA: in [39], the author focuses on the operation and dynamic deployment of MEC servers. Based on PPO reinforcement learning framework, PBPA algorithm is designed to determine the placement and migration of MEC servers in dynamic scenarios to minimize operating costs, which is a very advanced method in MEC server deployment related works.

C. Small Scale Network Simulation

In this section, a small-scale simulation involving 200 gNBs is conducted, and the specific partition results are shown in Fig. 5.

Fig. 5 (a-e) shows the outcomes of the small scale simulation, ranging from two partitions to six partitions. The partitions are signified by different colored nodes, with blue lines denoting the boundaries of gNB's serving area created by the Voronoi graph. The black lines represent the number of handovers, with thicker lines indicating higher handover numbers.

In Fig. 5(a), the RAN network is partitioned into two parts with two colors. It is obvious that the two partitions are fairly balanced. In the case of more partitions, MECP-GAP ensures a balanced trade-off between edge cut optimization and load balancing. In Fig. 5(b), green nodes are added as a new partition. This outcome stems from MECP-GAP's inclination towards improving the edge cut in this particular access network topology. Fig. 5(c-e) show the partition results for scenarios involving four to six partitions. The network operators can judiciously choose the number of partitions for MEC server deployment, taking into consideration the local user load and server capacity.

Fig. 6 illustrates the convergence behavior of MECP-GAP, using the six partitions scenario as an example. It becomes notably stable when the $epoch \approx 50$. Fig. 7 presents a comparison of the average running times for the three algorithms. Since PBPA deploying MEC servers only be used on a fixed graph. So the algorithm execution and network training process of PBPA are carried out simultaneously. Its simulation time is about 242.32s~265.45s. As the time is much longer than the above three algorithms', we did not put it on Fig. 7. It should be noted that we rely on the runtime performance of MECP-GAP derived from the trained I-GAP network model. This approach leverages the inherent generalization capabilities of the I-GAP network model. Notably, the MECP-GAP algorithm consistently maintains a low running time. In contrast, both the Greedy algorithm and Metis take longer to execute than MECP-GAP. The running time of the Greedy algorithm appears to change with the number of partitions, suggesting that the algorithm's performance may be influenced by the partition number. Metis, on the other hand, runs the longest, approximately three times as long as MECP-GAP.

Fig. 8(a-c) illustrate the loss function, edge cut, and load balancing plots with different number of partitions. In Fig. 8(a), all the four curves increase as the number of partitions increases. It's worth noting that, for the purpose of demonstrating algorithm performance, we set $\alpha = 1/1000$ and $\beta = 1$. So the result values of edge cut and load balancing are in the same order of magnitude. Consequently, the trend of the curve in Fig. 8(a) is similar to that in Fig. 8(b).

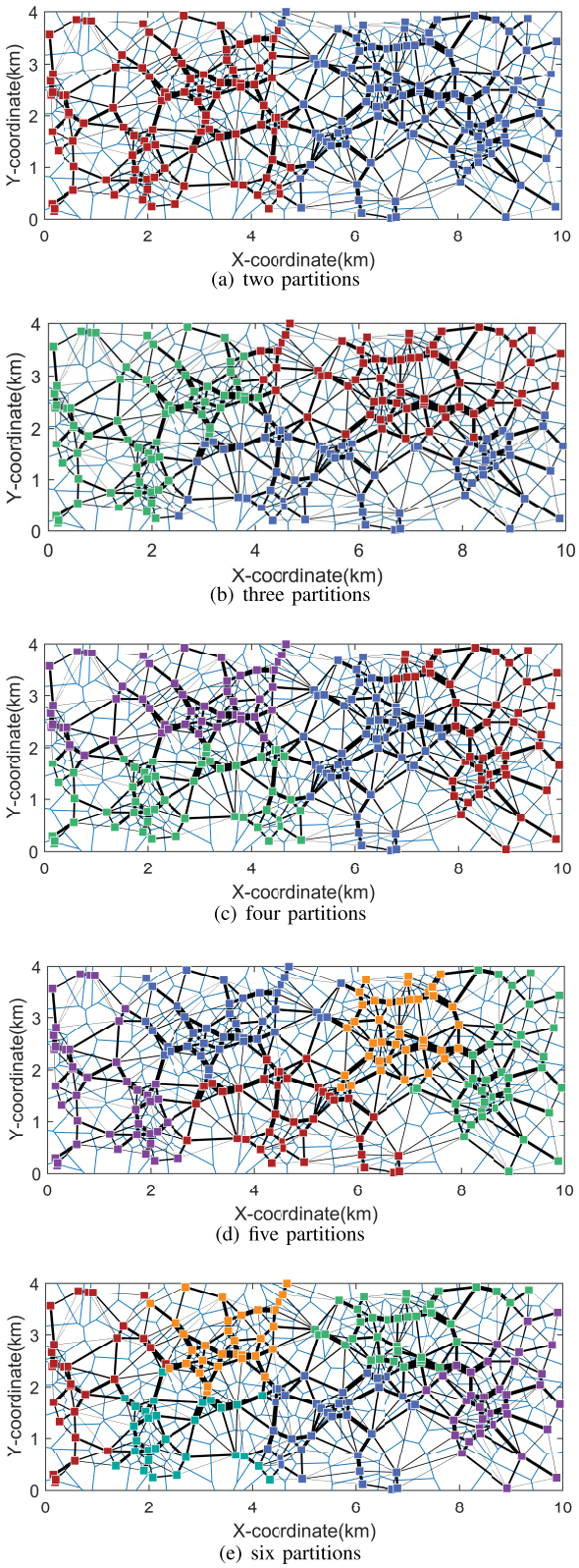

Fig. 5. Small scale network partition result from two partitions to six partitions.

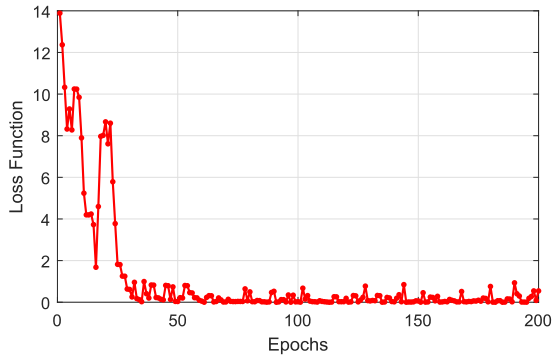

Fig. 6. Convergence of MECP-GAP for small scale network.

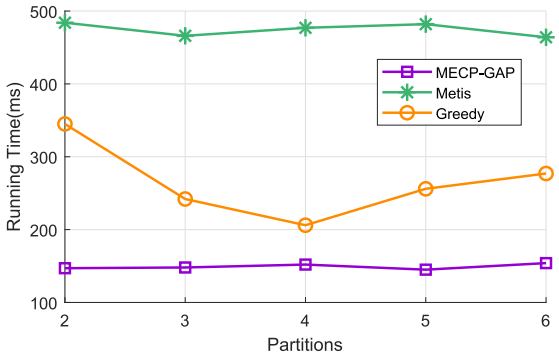

Fig. 7. Running time of MECP-GAP for small scale network.

The loss function of MECP-GAP is optimized under each partition because both of its components are optimal, as confirmed by Fig. 8(b) and Fig. 8(c). Fig. 8(b) shows the edge cut values, which increase with the growing number of partitions. Meanwhile, Fig. 8(c) displays the load balancing results of the three algorithms as they change with the number of partitions. Notably, the load balancing for the Greedy algorithm, Metis and PBPA exhibits no fixed trend due to the inherent randomness of these two algorithms. However, for MECP-GAP, the load balancing generally increases from 2 to 53 with the growing number of partitions.

Fig. 9 shows how the average handover delay of users changes over time in the small-scale network. The six partitions results generated by four algorithms are deployed to small-scale networks. By setting the mobile user to move linearly between different gNB areas, the simulation time is set to 80 minutes. The first ten minutes are the warm-up time. As the mobile users are deployed to the network, it begins to move in the network and across the boundaries of the MEC coverage area. Then the handover delay of the user to handover between the MEC server coverage areas under different algorithm is observed. It can be seen that MECP-GAP algorithm is the smallest in most of the time. It means that in the MEC service network generated by MECP-GAP, the average handover delay of users is the lowest. Besides, its fluctuation is also the most stable, which shows that the load balancing part of the loss function works, so that the handover tasks of users are more balanced. In contrast, Metis and Greedy perform less well. The average handover delay of PBPA is the highest and fluctuates in a wide range.

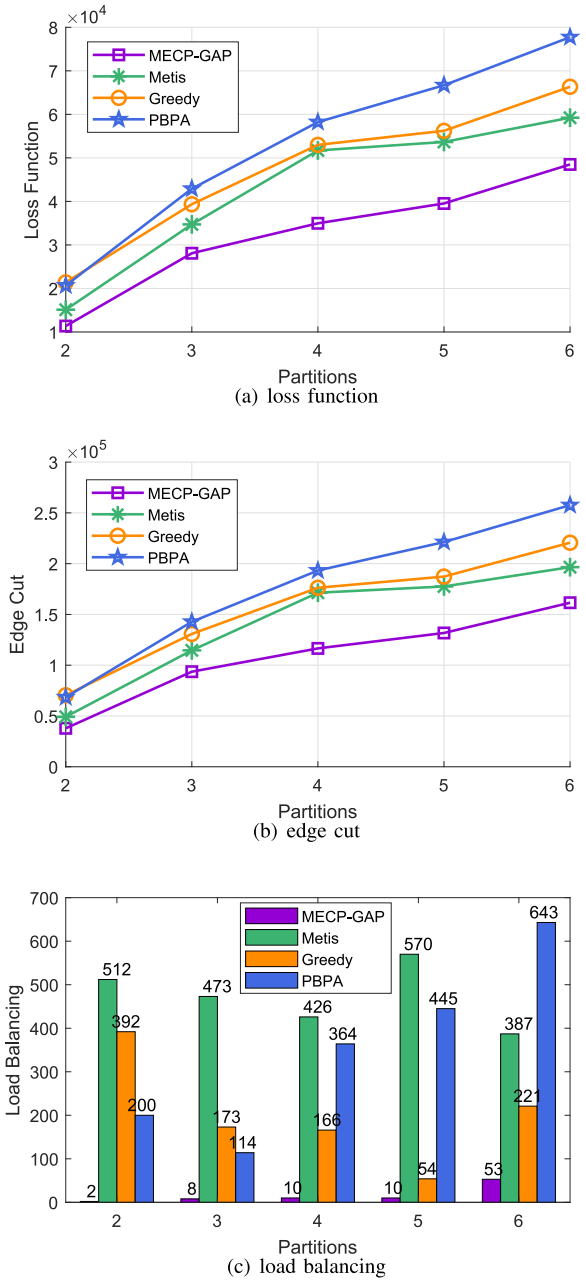

Fig. 8. Small scale optimization objective: (a) loss function, (b) edge cut, (c) load balancing.

D. Large Scale Network Simulation

In this section, we present the results of the large scale simulation. Due to the high density of gNBs in the simulation, the specific partition result graph would not be displayed.

Fig. 10 illustrates the convergence of the MECP-GAP algorithm in the large scale simulation. It's evident that the algorithm approaches convergence when $epoch < 50$. Fig. 11 displays the average running time of the three algorithms in the large scale simulation. Metis exhibits a slightly shorter average running time than MECP-GAP, indicating that Metis has a time advantage when dealing with large scale networks. The running time of Metis fluctuates between 1032 ms and 1104 ms, while MECP-GAP fluctuates between 1427 ms and

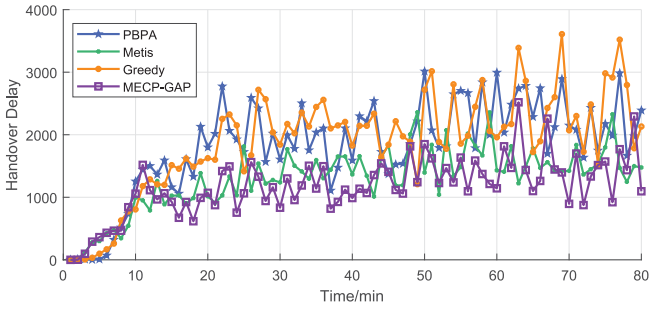

Fig. 9. Average handover delay for small scale network.

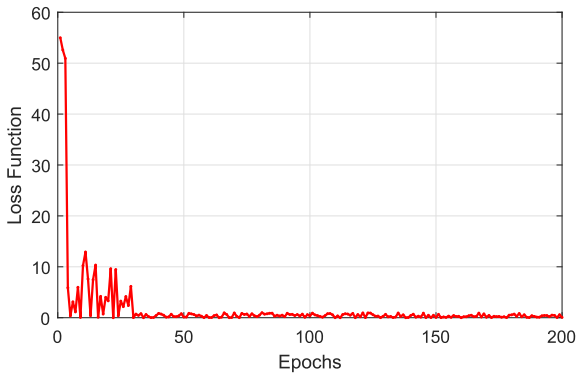

Fig. 10. Convergence of MECP-GAP for large scale network.

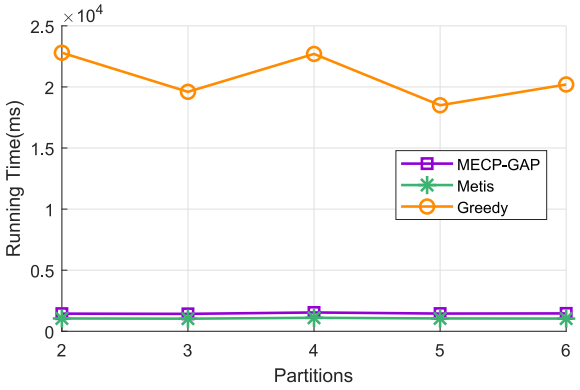

Fig. 11. Running time of MECP-GAP for large scale network.

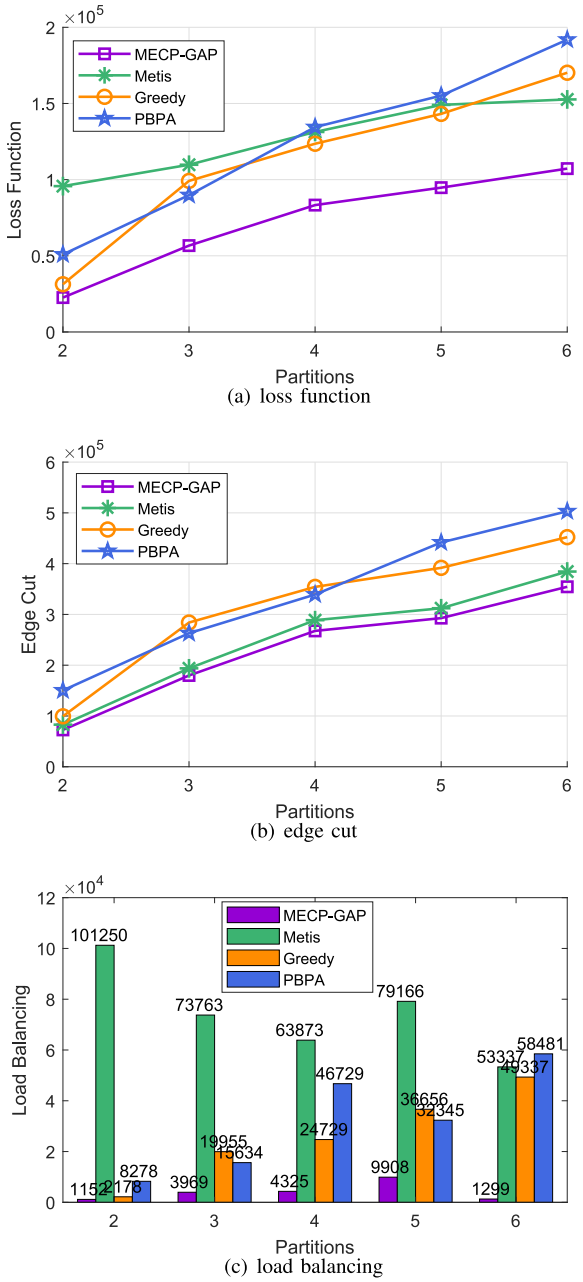

Fig. 12. Large scale optimization objective: (a) loss function, (b) edge cut, (c) load balancing.

1540 ms. Specially, the Greedy algorithm runs significantly slower than the first two when handling large scale networks, taking approximately ten times longer than the others. This is due to the high time complexity of greedy algorithms in the worst-case scenario. Similarly, PBPA only be used on a fixed graph when deploying MEC servers. The execution time of PBPA is in range of 1598.68s~1683.41s, which is much longer than the previous three algorithms. So it is not marked in Fig. 11.

Fig. 12(a-c) illustrates the changes in the loss function and its components for the four algorithms in the large-scale simulation. In Fig. 12(a), it's evident that MECP-GAP's loss function remains optimal, roughly $\frac{1}{3} \sim \frac{1}{2}$ of Metis. Fig. 12(b) specifically examines the edge cut component. The value for MECP-GAP is slightly lower than Metis's, both of which are lower than the Greedy algorithm and PBPA. In Fig. 12(c), it becomes apparent that the load balancing achieved by MECP-GAP is better in each partition compared to the other algorithms. This indicates that the partitions created by MECP-GAP are more balanced, making the server less prone to overload issues.

Fig. 13 shows how the average handover delay of the four algorithms changes over time in a large-scale network. It is generated in the same way as the small-scale simulation. It is obvious that MECP-GAP has the best performance in terms of delay value and stability. In contrast, Metis and Greedy delay values are longer and fluctuate in a large range. PBPA fluctuates in a large range, and its delay value is the longest in four algorithms. Mobile users have the worst handover experience in this scenario.

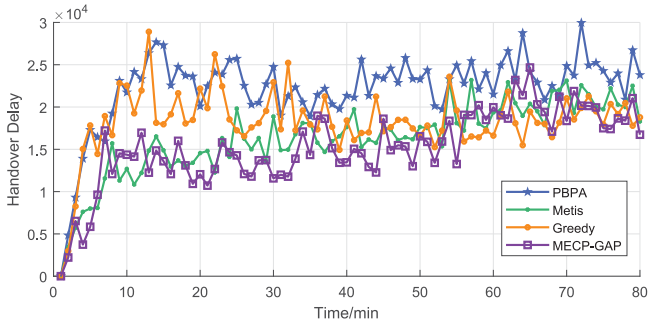

Fig. 13. Average handover delay for large scale network.

VI. Conclusion

In this work, we investigate the optimum mobility-aware MEC server planning problem by determining the serving area of each MEC servers to minimize the handover management cost for maintaining service continuity and balancing MEC servers workloads at the same time. The problem is formulated as a graph partitioning problem to partition the 5G RAN graph with minimum SSC management costs and balanced MEC servers workloads. Then, we adapt the GAP framework and propose a MEC server planning algorithm MECP-GAP. Leveraging on deep learning, MECP-GAP achieves better performance with lower running time by comparing to several traditional baselines. For future work, network planning optimization should be emphasized on more network scenarios, such as the 6G network. And learning based methods can be applied in dealing with more typical problems.

References

[1] S. Ji, M. Sheng, D. Zhou, W. Bai, Q. Cao, and J. Li, "Flexible and distributed mobility management for integrated terrestrial-satellite networks: Challenges, architectures, and approaches," *IEEE Netw.*, vol. 35, no. 4, pp. 73–81, Jul./Aug. 2021.

[2] "System architecture for the 5G system; stage 2; Release 15," 3GPP, Sophia Antipolis, France, Rep. TS 23.501, 2017.

[3] J. Liu, Q. Yang, and G. Simon, "Congestion avoidance and load balancing in content placement and request redirection for mobile CDN," *IEEE/ACM Trans. Netw.*, vol. 26, no. 2, pp. 851–863, Apr. 2018.

[4] T. Taleb, S. Dutta, A. Ksentini, M. Iqbal, and H. Flinck, "Mobile edge computing potential in making cities smarter," *IEEE Commun. Mag.*, vol. 55, no. 3, pp. 38–43, Mar. 2017.

[5] J. Qin and J. Liu, "Multi-access edge offloading based on physical layer security in C-V2X system," *IEEE Trans. Veh. Technol.*, vol. 71, no. 7, pp. 6912–6923, Jul. 2022.

[6] M. Shao, J. Liu, Q. Yang, and G. Simon, "A learning based framework for MEC server planning with uncertain BSs demands," *IEEE Access*, vol. 8, pp. 198832–198844, 2020.

[7] I. Leyva-Pupo, C. Cervelló-Pastor, C. Anagnostopoulos, and D. P. Pezaros, "Dynamic scheduling and optimal reconfiguration of UPF placement in 5G networks," in *Proc. 23rd Int. ACM Conf. Model., Anal. Simul. Wireless Mobile Syst.*, 2020, pp. 103–111. [Online]. Available: https://doi.org/10.1145/3416010.3423221

[8] X. Zhang, Z. Li, C. Lai, and J. Zhang, "Joint edge server placement and service placement in mobile-edge computing," *IEEE Internet Things J.*, vol. 9, no. 13, pp. 11261–11274, Jul. 2022.

[9] T. Lähderanta et al., "Edge computing server placement with capacitated location allocation," *J. Parallel Distrib. Comput.*, vol. 153, pp. 130–149, Jul. 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0743731521000605

[10] S. Wang, Y. Zhao, J. Xu, J. Yuan, and C.-H. Hsu, "Edge server placement in mobile edge computing," *J. Parallel Distrib. Comput.*, vol. 127, pp. 160–168, May 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0743731518304398

[11] C.-X. Wang, M. D. Renzo, S. Stanczak, S. Wang, and E. G. Larsson, "Artificial intelligence enabled wireless networking for 5G and beyond: Recent advances and future challenges," *IEEE Wireless Commun.*, vol. 27, no. 1, pp. 16–23, Feb. 2020.

[12] A. Nazi, W. Hang, A. Goldie, S. Ravi, and A. Mirhoseini, "GAP: Generalizable approximate graph partitioning framework," 2019, *arXiv:1903.00614*.

[13] I. Demirkol, C. Ersoy, M. Caglayan, and H. Delic, "Location area planning and cell-to-switch assignment in cellular networks," *IEEE Trans. Wireless Commun.*, vol. 3, no. 3, pp. 880–890, May 2004.

[14] L. Chen, H.-L. Liu, Z. Fan, S. Xie, and E. D. Goodman, "Modeling the tracking area planning problem using an evolutionary multi-objective algorithm," *IEEE Comput. Intell. Mag.*, vol. 12, no. 1, pp. 29–41, Feb. 2017.

[15] M. Bagaa, T. Taleb, and A. Ksentini, "Efficient tracking area management framework for 5G networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 6, pp. 4117–4131, Jun. 2016.

[16] H. Tabassum, M. Salehi, and E. Hossain, "Fundamentals of mobility-aware performance characterization of cellular networks: A tutorial," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2288–2308, 3rd Quart., 2019.

[17] R. Karmakar, G. Kaddoum, and S. Chattopadhyay, "Mobility management in 5G and beyond: A novel smart handover with adaptive time-to-trigger and hysteresis margin," *IEEE Trans. Mobile Comput.*, vol. 22, no. 10, pp. 5995–6010, Oct. 2023.

[18] X. Huang, J. Zong, and Y. Liu, "Mobility management for novel cooperative two-layer satellite network," in *Proc. Int. Wireless Commun. Mobile Comput. (IWCMC)*, 2023, pp. 1275–1280.

[19] S. M. A. Zaidi, H. Farooq, A. Rizwan, A. Abu-Dayya, and A. Imran, "A framework to address mobility management challenges in emerging networks," *IEEE Wireless Commun.*, vol. 30, no. 4, pp. 90–97, Jan. 2023.

[20] Q. Peng et al., "Mobility-aware and migration-enabled online edge user allocation in mobile edge computing," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, 2019, pp. 91–98.

[21] L. Zhao et al., "MESON: A mobility-aware dependent task offloading scheme for urban vehicular edge computing," *IEEE Trans. Mobile Comput.*, vol. 23, no. 5, pp. 4259–4272, May 2024.

[22] B. W. Kernighan and S. Lin, "An efficient heuristic procedure for partitioning graphs," *Bell Syst. Tech. J.*, vol. 49, no. 2, pp. 291–307, 1970.

[23] D. Van den Bout and T. Miller, "Graph partitioning using annealed neural networks," *IEEE Trans. Neural Netw.*, vol. 1, no. 2, pp. 192–203, 1990.

[24] T. Kawamoto, M. Tsubaki, and T. Obuchi, "Mean-field theory of graph neural networks in graph partitioning," 2018, *arXiv:1810.11908*.

[25] D. Gao, K. Li, R. Wang, S. Shan, and X. Chen, "Multi-modal graph neural network for joint reasoning on vision and scene text," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 12743–12753.

[26] Y. Wu, J. Liu, C. Wang, and Q. Yang, "Graph attention LSTM for load prediction of fine-grained VNFs in SFC," in *Proc. IEEE Int. Conf. Commun.*, 2023, pp. 4899–4904.

[27] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2009.

[28] T. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2017, *arXiv:1609.02907*.

[29] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, *Message Passing Neural Networks*. Cham, Switzerland: Springer Int. Publ., 2020, pp. 199–214.

[30] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *Proc. ICLR*, 2017, p. 20.

[31] S. Kekki et al., "MEC in 5G networks," ETSI, Sophia Antipolis, France, White Paper, 2018.

[32] "Procedures for the 5G system (5GS)," ETSI, Sophia Antipolis, France, document TS 123 502-2021, 2021.

[33] T. Taleb and A. Ksentini, "Gateway relocation avoidance-aware network function placement in carrier cloud," in *Proc. 16th ACM Int. Conf. Model., Anal. Simul. Wireless Mobile Syst.*, 2013, pp. 341–346.

[34] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. 31st Int. Conf. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1025–1035.

[35] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021.

[36] P. C. Sen, M. Hajra, and M. Ghosh, "Supervised classification algorithms in machine learning: A survey and review," in *Proc. IEM Graph Emerg. Technol. Model. Graph.*, 2020, pp. 99–111.
[37] 2022, "The world's largest open database of cell towers," Data Set, OpenCellID. [Online]. Available: https://www.opencellid.org/#zoom=16&lat=37.77889&lon=-122.41942
[38] X. Liang, J. Zhao, L. Dong, and K. Xu, "Unraveling the origin of exponential law in intra-urban human mobility," *Sci. Rep.*, vol. 3, no. 1, p. 2983, 2013.
[39] X. Jiang, P. Hou, H. Zhu, B. Li, Z. Wang, and H. Ding, "Dynamic and intelligent edge server placement based on deep reinforcement learning in mobile edge computing," *Ad Hoc Netw.*, vol. 145, Jun. 2023, Art. no. 103172.

Jiayi Liu (Member, IEEE) received the Bachelor of Science degree in electronic engineering from Xidian University, Xi'an, China, in 2007, the Master of Science degree in computer science from Télécom-Bretagne, France, in 2009, and the Ph.D. degree from Rennes 1 University, France, in 2013. Since 2014, she has been working as a Lecturer with the School of Telecommunication Engineering and the Guangzhou Institute of Technology, Xidian University. Her research interests include 5/6G network, content distribution, network resource scheduling, and AI enabled network management.

Zhongyi Xu received the B.S. degree in communication engineering from Shandong University, Weihai, China, in 2021, and the M.S. degree in communication and information system from Xidian University, Xi'an, China, in 2024. His research interests include 5G network and mobile edge computing.

Chen Wang received the Bachelor of Science degree in computer science from Tongji University, Shanghai, China, in 2007, the Master of Science degree in computer science from Télécom-Bretagne, France, in 2009, and the Ph.D. degree from the National Institutes of Science and Technology, France, in 2013. He worked as a Research Scientist with Bosch, China, from 2014 to 2017. Since 2017, he has been working as a Research Scientist with Huawei. His current research interests include machine learning algorithms and AI applications.

Xuefang Liu received the B.S. degree in communication engineering from the Lanzhou University of Technology, China, in 1998, and the M.S. and Ph.D. degrees in information and communication systems from Xidian University, China, in 2004 and 2009, respectively. Since 1998, she has been with Xidian University. During this time, she also went on a study tour with the University of Houston from 2016 to 2017. Her current research interest lies in the fields of cooperative communication, content delivery networks, and 6G communication networks.

Xuemei Xie (Senior Member, IEEE) received the M.S. degree in electronic engineering from Xidian University, Xi'an, China, in 1994, and the Ph.D. degree in electrical and electronic engineering from The University of Hong Kong in 2004. She is a Professor with the School of Artificial Intelligence, Xidian University. Her research interests are human action recognition, object detection, scene understanding, video analysis, deep learning, and feature representation.

Guangming Shi (Fellow, IEEE) received the M.S. degree in computer control and the Ph.D. degree in electronic information technology from Xidian University, Xi'an, China, in 1988 and 2002, respectively, where he was the Vice President from 2018 to 2022. He is currently the Vice Dean of Peng Cheng Laboratory and a Professor with the School of Artificial Intelligence, Xidian University. His research interests include artificial intelligence, semantic communications, and human-computer interaction. He was awarded the Cheung Kong Scholar Chair Professor by the Ministry of Education in 2012. He won the Second Prize of the National Natural Science Award in 2017. He is the Chair of IEEE CASS Xian Chapter, a Senior Member of ACM and CCF, and a Fellow of the Chinese Institute of Electronics and of IET.