HierFedML: Aggregator Placement and UE Assignment for Hierarchical Federated Learning in Mobile Edge Computing

Zichuan Xu, *Member, IEEE,* Dapeng Zhao, Weifa Liang, *Senior Member, IEEE,* Omer F. Rana, *Senior Member, IEEE,* Pan Zhou, *Senior Member, IEEE,* Mingchu Li, Wenzheng Xu, *Member, IEEE,* Hao Li, and Qiufen Xia, *Member, IEEE.*

Abstract—Federated learning (FL) is a distributed machine learning technique that enables model development on user equipments (UEs) locally, without violating their data privacy requirements. Conventional FL adopts a single parameter server to aggregate local models from UEs, and can suffer from efficiency and reliability issues – especially when multiple users issue concurrent *FL requests*. Hierarchical FL consisting of a master aggregator and multiple worker aggregators to collectively combine trained local models from UEs is emerging as a solution to efficient and reliable FL. The placement of worker aggregators and assignment of UEs to worker aggregators plays a vital role in minimizing the cost of implementing FL requests in a Mobile Edge Computing (MEC) network. Cost minimisation associated with joint worker aggregator placement and UE assignment problem in an MEC network is investigated in this work. An optimization framework for FL and an approximation algorithm with an approximation ratio for a single FL request is proposed. Online worker aggregator placements and UE assignments for dynamic FL request admissions with uncertain neural network models, where FL requests arrive one by one without the knowledge of future arrivals, is also investigated by proposing an online learning algorithm with a bounded regret. The performance of the proposed algorithms is evaluated using both simulations and experiments in a real testbed with its hardware consisting of server edge servers and devices and software built upon an open source hierarchical FedML (HierFedML) environment. Simulation results show that the performance of the proposed algorithms outperform their benchmark counterparts, by reducing the implementation cost by at least 15% per FL request. Experimental results in the testbed demonstrate the performance gain using the proposed algorithms using real datasets for image identification and text recognition applications.

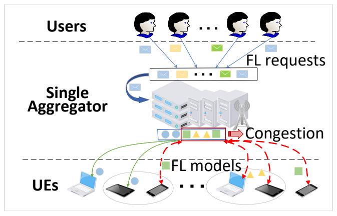
Index Terms—Mobile edge computing; Federated learning; Aggregator placement and UE assignment; hierarchical federated learning framework; approximation algorithms; Online learning algorithms.

1 Introduction

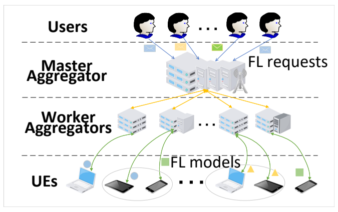
AI applications continuously generate large volume of data in mobile edge computing (MEC) networks, which need be analyzed to obtain valuable patterns for business advantage and decision making [49], [51], [53], [54]. Federated Learning (FL) is a promising distributed learning technique to avoid privacy breaches of data generated by user equipment (UE) – often including sensitive information such as user face images. FL involves training a *local model* on a UE and then aggregating these models to obtain a global machine learning model. Conventional FL frameworks usually adopt a single aggregator (referred to as a "parameter server") that receives all trained local models of UEs to derive the global training model. As reported in many existing studies [3], [26], [37], [59], such a FL framework suffers issues such as inefficient communications and communication failures. For instance, the single aggregator becomes a bottleneck if many UEs send their local models to it; or, if the single aggregator fails to operate, the local models of UEs will not be aggregated, thereby significantly reducing the accuracy of the obtained global model. The above concerns can become more important in an MEC network that consists of heterogeneous cloudlets with different computing resource capacities, spread across a large geographical area. In particular, assuming that the process of obtaining a global model is referred to as an *FL request*, multiple FL requests are issued by users in an MEC network to train different global models. Since the computing resource of each cloudlet is limited, it will incur a performance degradation when a large number of users request to obtain machine learning models. Additionally, if a single request needs to aggregate local models from many UEs, the cloudlet that serves as the aggregator will not be able to respond to other requests if it

- *Z. Xu, D. Zhao, and M. Li are with the Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, School of Software, Dalian University of Technology, Dalian, China, 116621. E-mails: z.xu@dlut.edu.cn, zhaodapeng97@mail.dlut.edu.cn, mingchul@dlut.edu.cn.*
- *W. Liang is with the Department of Computer Science, City University of Hong Kong, Hong Kong. E-mail: weifa.liang@cityu.edu.hk.*
- *O. Rana is with the Physical Sciences and Engineering College, Cardiff University, United Kingdom. Email: RanaOF@cardiff.ac.uk.*
- *P. Zhou is with the School of Cyber Science and Engineering, Huazhong University of Science and Technology, Wuhan, Hubei, China. E-mail: panzhou@hust.edu.cn.*
- *W. Xu is with the Department of Computer Science, Sichuan University, Chengdu 610065, China. Email: wenzheng.xu@scu.edu.cn.*
- *H. Li (Corresponding author) is with the Ningxia Research Institute, China Coal Research Institute, Ning Xia 750004, China. Email: hateif@163.com.*
- *Q. Xia is with the Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, the International School of Information Science & Engineering, Dalian University of Technology, Dalian, China, 116621. E-mail: qiufenxia@dlut.edu.cn.*

Manuscript received XXX; revised XXX.



(a) The conventional FL with a single aggregator.



(b) The hierarchy-based FL with master aggregators and worker aggregators.

Fig. 1. The motivation for using hierarchical FL

is depleted of computing resource – as shown in Fig. 1 (a).

A hierarchical FL with a *master aggregator* and multiple *worker aggregators* to combine trained local models from UEs [4], is adopted to address the afore-mentioned issues of conventional FL frameworks with a single aggregator. Specifically, a hierarchical FL has multiple rounds of training and local model aggregations. In each round, each worker aggregator collects the local models of its UEs. The master aggregator obtains the global model by aggregating the local models collected from all worker aggregators, and returns the updated global model back to all UEs for the next round of local training in each UE. Since the worker aggregators can be distributed to multiple cloudlets of the MEC network, the congestion level and cost of each worker aggregator can be alleviated. The comparison between conventional FL and the hierarchical FL is given in Fig. 1 (b).

1.1 Motivation and Challenges

Under a hierarchical FL framework, the admission cost of FL requests is dominated by aggregator locations and the assignment of UEs to different worker aggregators. If a UE is assigned to a worker aggregator located far from itself, the cost of transmitting its training data may be very high. On one hand, from the perspective of users, a UE is not willing to participate in the FL if its cost is high. On the other hand, a service provider that operates the MEC network will not profit from offering FL services for mobile users. In this paper, we study the problem of joint worker aggregator placement and UE assignment in an MEC network to minimize implementation costs of all FL requests, which poses a number of fundamental challenges, as discussed below.

First, various machine learning models can be developed for a single learning task. For example, machine learning models for object recognition include R-CNN [11], Faster R-CNN [34], and YOLOv5 [19]. Given a specific FL request, the models chosen/adopted are not always certain. To enable learning in UEs, each UE can adaptively prune machine learning models, referred to as submodels, of a full machine learning model. Since each UE makes its own decision, which submodels are selected in UEs cannot be specified in advance. Due to the heterogeneous capabilities available on UEs, submodels of a neural network with different hyper-parameters may be used to adapt to dynamically-changing computing capabilities of UEs [8]. Applying machine learning in MEC networks leads to uncertain local training models at UEs, which are considered as *uncertain models*. Such model uncertainty makes the placements of worker aggregators a difficult optimisation problem. As such, the decision of placing worker aggregators and UE assignment for an FL request has to be made under such model uncertainty.

Second, the capacity constraint on each base station requires worker aggregators to be distributed to different base stations. FL implementation cost may be reduced if a smaller number of worker aggregators are deployed. However, this may result in a higher communication cost as UEs may require longer routing paths to send trained local models to these worker aggregators. Finding a trade-off between communication cost and the number of worker aggregators for each FL request implementation is challenging.

1.2 Novelty and Contributions

Current literature on FL in MEC networks has ignored the joint aggregator placement and UE assignment, and often focuses on optimizing communication and energy costs [35], [55], [58], [64] as well as efficient UE assignment [60]. To the best of our knowledge, unlike these existing studies that considered a single FL request admission only, we are the first to consider the joint worker aggregator placement and UE assignment. Our approach also considers multiple FL requests with model uncertainty in an MEC network. The main contributions of this paper are as follows.

- We propose a hierarchical FL optimization framework for joint worker aggregator placement and UE assignment for a single FL request admission. The optimization framework consists of a well-tuned search process to identify the number of worker aggregators for a request, and an approximation algorithm with an approximation ratio for the worker aggregator placement and UE assignment.
- We consider the online worker aggregator placement and UE assignment for multiple FL request admissions with model uncertainty, by proposing a novel online learning algorithm based on the multi-armed bandits (MAB) method.
- We evaluate the performance of the proposed algorithms by simulation. Results show that the proposed algorithms outperform their counterparts by reducing the implementation cost by at least 15% per FL request.
- We built a testbed for hierarchical FL in an MEC network and implemented a hierarchical FL software framework HierFedML, extending an existing FedML framework [12]. Experiment results of HierFedML demonstrate that the performance of the proposed algorithms are promising, based on real data sets for image identification and text recognition applications.

The remainder of this paper is arranged as follows. Section 2 introduces the system model, notations and problem formulation. Section 3 describes the proposed optimization framework and proposes an approximation algorithm for a single FL request implementation. Section 4 describes mechanisms to support dynamic FL requests, for a sequence of FL requests with model uncertainty, arriving one by one without knowledge of their arrivals. Section 6 evaluates the performance of the proposed algorithms by both simulation and using a physical testbed. Section 7 includes concluding comments.

2 Related work

In this section, we review existing studies and summarize the differences between our work and existing efforts. We focus specifically on efficiency achieved within conventional (distributed) machine learning tasks, optimization for FL and resource allocation for FL.

2.1 Efficiency in Machine Learning

Although there are studies focusing on efficiency of machine learning-based tasks [2], [20], such methods cannot be directly applied to FL in MEC networks, as they mainly consider inference tasks instead of distributed training tasks. Specifically, each FL request involves the collaboration between different edge servers. For example, Acharya *et al.* [2] studied the re-identification of persons in a large-scale video surveillance network, by allocating inference tasks in an MEC network with both edge servers and a remote cloud. The objective is to promote the responsiveness of inference tasks for re-identification. Furthermore, Li *et al.* [20] investigated the problem of task offloading from Internet-of-Things (IoT) devices to edge servers. They assumed each task to be executed on a single edge server; while the FL request requires collaboration between different aggregators in the MEC network. Xu *et al.* [51] focus on distributed person re-identification and optimized the network latency of the distributed model in an MEC network. Recently, FL is emerging as a key distributed training technique that guarantees the privacy requirements of users [7], [9], [27], [28], [42], [63]. However, most of these FL frameworks do not focus on promoting the cost efficiency of FL in MEC networks. For example, Mills *et al.* [28] modified FedAvg to use distributed Adam optimization called CE-FedAvg, greatly reducing the number of rounds of convergence via a novel compression method. Duan *et al.* [9] made use of adaptive data expansion and downsampling to solve the imbalanced training data distribution problem. Since FedAvg is not ideal for training data with highly skewed distribution [62]. Wang *et al.* [42] designed an experience-driven framework to intelligently choose users to participate in FL. Liu *et al.* [25] proposed an adaptive asynchronous federated learning (AAFL) mechanism to achieve less convergence time of training under resource constraints.

2.2 Efficiency Optimization for FL

Due to the distributive nature of FL, many existing studies promote communication efficiency [35], [42], [58], [64], optimize energy consumption [55], and find non-trivial trade-offs between the accuracy and convergence speed [46], [40] of a single FL request admission. However, little attention has been paid to the worker aggregator placement and UE assignment for FL request admissions. Wu *et al.* [47] conducted a detailed analysis of parameter updates on non-i.i.d datasets and compared the difference with i.i.d datasets. They proposed a structure-based communication reduction algorithm. Liu *et al.*[24] proposed an efficient-communication approach, which provides a customized local training strategy for vehicular clients to achieve convergence quickly within fewer communication rounds. Wang *et al.* [44] designed a federated deep-reinforcement-learning method for the problem of cooperative edge caching. Yu *et al.* [57] developed a mobility-aware edge content caching algorithm to allow multiple vehicles to collaborate to learn a model. Yang *et al.* [58] proposed a novel over-the-air computation approach to enable a fast global model aggregation in a wireless multiple-access channel.

2.3 Resource Allocation for FL in MEC Networks

Existing studies on resource allocation for FL in an MEC network assumed that worker aggregators of an FL request implementation are consolidated into a single location, and ignore model uncertainty. They thus cannot be directly applied to the aggregator and UE assignment problem being considered in this paper. For example, Wang *et al.* [43] analyzed the convergence bound of a distributed gradient descent algorithm. Luo *et al.* [26] studied the problem of joint resource allocation and UE assignment problem to minimize the implementation cost of FL requests. Shi *et al.* [38] investigated the problem of joint UE selection and resource allocation to maximize model accuracy under a given training time budget. Though Xia *et al.* [48] investigated the uncertainty of wireless channel state information, they did not consider model uncertainty. Feng *et al.* [10] explored a min-max cost-optimal problem to guarantee the convergence rate of FL in edge networks, to minimize the cost of UEs, subject to the constraints of delay and computing capacity. Ko *et al.* [13] studied the problem of UE selection and bandwidth allocation in a wireless network, with an objective to minimize the average time associated with training rounds while meeting a requirement on the training data type and volume. The placement of worker aggregators was not considered, and the computing resource capacity in the backhaul of the network was ignored in this work. There are also studies focusing on promoting communication efficiency for hierarchical FL in an MEC network [21], [22]. Specifically, [21], [22] focused on incentive mechanisms that could motivate data owners to participate in the FL process such that communication efficiency is promoted. In particular, under the hierarchical FL framework, authors in [22] proposed a hierarchical game, where the lower-level is an evolutionary game for resource allocation and the upper-level is a Stackelberg game to determine a reward for players. In contrast, in this paper we focus on the worker aggregator placement and UE assignment problem by assuming that UEs are willing to participate the FL as long as they are selected. Wang *et al.* [45] also considered a hierarchical FL framework by proposing an algorithm to group UEs into clusters. Each cluster has a single worker aggregator and a master aggregator that collects and aggregates the model from each cluster.

3 Preliminaries

In this section, we first introduce the system model, definitions and notations. We then formally define the optimization problems being considered in this work.

3.1 System Model

As shown in Fig. 2, we consider an MEC network $G = (\mathcal{BS} \cup \mathcal{CL}, E)$ with a set $\mathcal{BS}$ of base stations and a set $\mathcal{CL}$ of cloudlets. Cloudlets are located in the backhaul of the MEC network G. Let bs_i and cl_j be a base station and a cloudlet in $\mathcal{BS}$ and $\mathcal{CL}$, respectively. The resources of both base stations and cloudlets have associated capacity constraints. Each cloudlet cl_j or base station bs_i is a potential location Loc_q ($\in \{\mathcal{BS} \cup \mathcal{CL}\}$) for implementing FL requests. Let C_q and B_q be the computing and bandwidth resource capacities at Loc_q, respectively. E is a set of backhaul links interconnecting base stations and cloudlets. Each base station bs_i provides wireless connection for UEs and wired connection to the backhaul network. Therefore, base station bs_i also has a capacity $B(bs_i)$ of wireless bandwidth. Each UE connects to the MEC network via one of its close base stations. Let ue_k and ds_k be a UE and its data, respectively, where k is an integer with $1 \leq k \leq K$ and K is the number of UEs in the MEC network. The volume of data ds_k is denoted by $|ds_k|$.

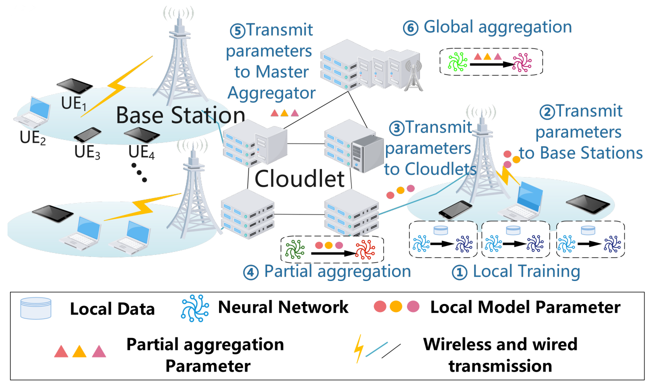

⑤Transmit parameters to Master Aggregator
⑥ Global aggregation
UE1
UE2
UE3
UE4
Base Station
③Transmit parameters to Cloudlets
②Transmit parameters to Base Stations
Cloudlet
④ Partial aggregation
① Local Training
Local Data
Neural Network
Local Model Parameter
Partial aggregation Parameter
Wireless and wired transmission

Fig. 2. An example of a MEC network

3.2 Federated Learning in MEC Networks

An *FL request*, denoted by r_m, is issued to train on the datasets that are distributed in UEs of the MEC network. Each FL request r_m requires an *FL service* for persistent storage to store its learned machine learning model and efficient coordination of UEs to guarantee the trained accuracy. An FL service is usually implemented in a Virtual Machine (VM) or a container of the MEC network. Denote by S_m an FL service. Let $|w_m|$ be the size of a model that is transmitted between a UE and its service S_m.

In each round of FL training, the trained local model of each UE needs to be sent for aggregation. We consider a hierarchical aggregation framework with a *master aggregator* and a set of *worker aggregators*, where the master aggregator is co-located with FL service S_m. Due to limited resources on cloudlets and base stations, the worker aggregators will be distributed to different nodes (cloudlets) in G. Let $A_{m,o}$ be the oth worker aggregator of FL request r_m. We assume that each UE has to register itself to one worker to send its trained local model for aggregation. Let $\mathcal{UE}_m$ be the set of UEs for FL request r_m.

3.3 Cost Models

The implementation of an FL request incurs communication, data aggregation and local training costs [16], [50].

The communication cost is incurred due to the downloading of the global model from service S_m to each UE and uploading the trained local models of UEs. Following existing studies [41], [15], [56], we assume that the communication cost is mainly due to the consumption of bandwidth resources, which is proportional to the amount of data to be transferred. Let $c^t_{k,q,i}$ be the cost of transmitting a unit amount of data from UE ue_k to location Loc_q via base station bs_i. If UE ue_k is assigned to a worker aggregator $A_{m,o}$ in location Loc_q, the communication cost of ue_k for updating its trained model to $A_{m,o}$ of service S_m is

$$c^t_{k,m,o} = \sum\nolimits_{Loc_q \in \mathcal{BS} \cup \mathcal{CL}} \sum\nolimits_{bs_i \in \mathcal{BS}} |w_m| \cdot c^t_{k,q,i} \cdot x_{m,o,q} \cdot z_{k,i},$$

where $x_{m,o,q}$ is a binary variable that shows whether worker aggregator $A_{m,o}$ is placed at location Loc_q, and $z_{k,i}$ indicates whether UE ue_k accesses the MEC network via base station bs_i. Similarly, each worker aggregator sends its aggregated model to service S_m. Let $y_{m,q}$ be an indicator variable that shows whether service S_m is placed at location Loc_q. The communication cost due to model updating from a worker aggregator to the master aggregator in S_m is

$$c^t_{m,o} = \sum\nolimits_{Loc_q, Loc_{q'} \in \mathcal{BS} \cup \mathcal{CL}} |w_m| \cdot x_{m,o,q} \cdot y_{m,q'} \cdot c^t_{q,q'},$$

where $c^t_{q,q'}$ is the cost of transmitting a unit of data from Loc_q to $Loc_{q'}$.

The data aggregation cost is due to local model aggregations in worker aggregators. Typical implementation of an FL request takes an average of the gradient on its local parameters and the current model, and service S_m aggregates the local models by updating its current model. Data aggregation cost of each worker aggregator depends on the number of registered UEs and the location of the aggregation (i.e., with or without accelerators) [25], [52], [61]. Denote by $\mathcal{UE}_{m,o}$ the number of UEs assigned to worker aggregator $A_{o,m}$ of FL request r_m. Let c^a_m be the cost of data aggregation for r_m, we have

$$c^a_m = \sum\nolimits_{A_{m,o} \in \mathcal{A}_m} \sum\nolimits_{Loc_q \in \mathcal{BS} \cup \mathcal{CL}} x_{m,o,q} \cdot \alpha_q \cdot |\mathcal{A}_m|,$$

where $\mathcal{A}_m$ is the set of deployed worker aggregators of FL request r_m and α_q is the cost of aggregating a UE's local model in Loq_q.

The local training cost is due to the model training at each UE. As the stochastic gradient descent method is adopted to optimize a global loss function, the local training cost is proportional to the size of the local dataset. Let c^{unit}_k be the cost of learning a unit data in UE ue_k. The cost c^l_k of local learning in UE ue_k on its dataset ds_k is $c^l_k = c^{unit}_k \cdot |ds_k|$.

3.4 Resource Consumption Models

Each UE, cloudlet or base station consumes wireless and wired bandwidth resources for transmitting trained models. Let η_i be the amount of wireless bandwidth resource that bs_i uses to receive data. The amount of bandwidth resource required to transfer the amount $|w_m|$ of trained local model for FL request r_m is $\eta_i \cdot |w_m|$. Let ξ_i be the amount of wired bandwidth resource assigned to transfer a unit data. The amount of bandwidth resource demanded by bs_i for transferring w_m is $\xi_i \cdot |w_m|$. The worker aggregator located at Loc_q needs bandwidth resource to transfer its aggregated parameters to the master aggregator. Let ξ_q be the amount of bandwidth resource it is assigned to transfer a unit data at Loc_q. Hence an amount $\xi_q \cdot |w_m|$ of bandwidth resource is needed to transfer $|w_m|$ of aggregated parameters is needed.

Work aggregators need computing resources to aggregate the trained models of UEs. Denote by $\mathcal{UE}_{m,o}$ the set of UEs assigned to worker aggregator $A_{m,o}$, the computing resource needed is proportional to the size of the parameters for aggregation, i.e., $CR_{m,o,q} = \gamma_q \cdot |\mathcal{UE}_{m,o}| \cdot |w_m|$, where γ_q is a proportional factor at location Loc_q. Similarly, the amount of computing resource allocated to service S_m is $CR'_{m,q} = \gamma_q \cdot |\mathcal{A}_m| \cdot |w_m|$, if S_m is placed to location Loc_q. Although a single FL service may not consume significant amounts of computing resources, multiple FL services can run out of the entire available computing resource in a base station easily [6], [30].

3.5 Problem Definition

Assuming that the volume of each dataset ds_k of UE ue_k is given apriori, *the joint worker aggregator placement and UE assignment problem for a single FL request admission* in an MEC network is formulated as describe in this section. The problem aims to jointly place service S_m to a location, find a suitable number of worker aggregators and their locations for an FL request r_m, and assign UEs to different base stations and worker aggregators. The objective of the problem is to minimize the cost of implementing the FL request, subject to the resource capacity on each location $Loc_q \in \mathcal{BS} \cup \mathcal{CL}$.

Given a set R of FL requests arriving into the system dynamically without knowing future FL request arrivals, assuming that the model of FL request r_m is uncertain, *the online worker aggregator placement and UE assignment problem for multiple FL requests with model uncertainty* in an MEC network is to find a the number of worker aggregators and their placements for each FL request $r_m \in R$, and assign UEs to base stations and worker aggregators of each r_m, such that the total implementation cost of all FL requests in R is minimized, subject to the resource capacity at each location $Loc_q \in \mathcal{BS} \cup \mathcal{CL}$.

For clarity, the symbols used in this paper are summarized in Table 1.

4 Algorithms for the Joint Worker Placement and UE Assignment Problem with a Single FL Request Admission

In the following we propose an optimization framework for the problem, assuming that the model of an FL request is given.

4.1 An Optimization Framework

The number of worker aggregators deployed for each FL request r_m plays a vital role in minimizing its implementation cost. Specifically, more worker aggregators for FL request r_m imply that these worker aggregators can be distributed to locations closer to its UEs, thereby reducing the cost of updating the trained local model. This however may increase the volume of data transfers from worker aggregators to service S_m, resulting in a high communication cost.

To identify the number of worker aggregators, we adopt binary search. That is, the maximum number of worker aggregators that can be instantiated is determined by the available computing resource in each location. Let n^{min} and n^{max} denote the minimum and maximum numbers of worker aggregators of each FL request r_m, respectively.

We first determine the value of n^{min}. Intuitively, the minimum value of n^{min} is 1, implying that the trained models by all UEs are aggregated by a single worker aggregator. However, the available resource at location Loc_q may not be adequate to aggregate the data of all UEs in $\mathcal{UE}$. This means if locations have the maximum computing resource $\max\{C_q \mid \forall Loc_q\}$ to process the local models of UEs in $\mathcal{UE}$, we will get the minimum number of worker aggregators, i.e., $\lceil \frac{\gamma_q \cdot |w_m| \cdot |\mathcal{UE}_m|}{\max\{C_q \mid \forall Loc_q\}} \rceil$. Similarly, if we consider the bandwidth resource capacity of each location Loc_q, we get the minimum number of worker aggregators as $\lceil \frac{\xi_q \cdot |w_m| \cdot |\mathcal{UE}_m|}{\max\{B_q \mid \forall Loc_q\}} \rceil\}$ Overall, we have

$$n^{min} = \max\{1, \lceil \frac{\gamma_q \cdot |w_m| \cdot |\mathcal{UE}_m|}{\max\{C_q \mid \forall Loc_q\}} \rceil, \lceil \frac{\xi_q \cdot |w_m| \cdot |\mathcal{UE}_m|}{\max\{B_q \mid \forall Loc_q\}} \rceil\}. \quad (1)$$

We then determine the value of n^{max}. The maximum amount of computing resource consumed by worker aggregator $A_{m,o}$ is determined by the number of UEs assigned to it. In the worst case, a worker aggregator only aggregates the local model from a maximum number $|\mathcal{UE}_m|$ of UEs, i.e.,

$$n^{max} = |\mathcal{UE}_m|. \quad (2)$$

We proceed by describing the binary search process. Specifically, we follow a binary search procedure to find an appropriate number n_m of worker aggregators for FL request r_m in the range of $[n^{min}, n^{max}]$. Note that instead of comparing the values of n_m, n^{min}, and n^{max} in each round of the binary search. We here compare the cost of implementing FL request r_m when the number of worker aggregators is selected, which is obtained by invoking algorithm `ApproAG` or `HeuAG` to place worker aggregators and service for r_m. Let $cost(n_m)$ be the cost of the solution delivered by `ApproAG`.

The detailed steps of the proposed optimization framework are shown in **Algorithm** 1, which is referred to as `OptFWK`.

Algorithm 1 `OptFWK`

```
Input: An MEC network $G$, and an FL request $r_m$.
Output: The cost of implementing an FL request $r_m$ in the MEC network $G$.
1: Set values for $n^{min}$ and $n^{max}$ according to equations (1) and (2);
2: while $cost(n'^{min}) < cost(n'^{max})$ do
3:     $n_m \leftarrow \lceil \frac{n'^{min}+n'^{max}}{2} \rceil$;
4:     Invoke algorithm ApproAG to obtain the cost $cost(n_m)$ of implementing request $r_m$ with a number $n_m$ of worker aggregators;
5:     Calculate $cost(n'^{min})$ and $cost(n'^{min})$ if necessary;
6:     if $cost(n_m) \le cost(n'^{min})$, $n'^{min} \leftarrow n_m$; else, $n'^{max} \leftarrow n_m$;
7: end while
```

4.2 An Approximation Algorithm with A Fixed Number of Worker Aggregators

The proposed algorithm aims to map the problem to the minimum-cost multi-commodity flow problem in an auxiliary graph $G' = (V', E')$. We treat the trained model of each UE as a 'commodity' to be routed from its source to destination. However, the commodity may be split into multiple parts and routed along multiple routing paths as a splittable flow in the minimum-cost multi-commodity flow

TABLE 1
Symbols

Symbols	Meaning
$G = (\mathcal{BS} \cup \mathcal{CL}, E)$	An MEC network with a set $\mathcal{BS}$ of base stations and a set $\mathcal{CL}$ of Cloudlet.
cl_j, and bs_i	a cloudlet in $\mathcal{CL}$, and a base station in BS.
Loc_q	a potential location for implementing FL requests, which can be either a base station or a cloudlet.
C_q and B_q	the computing and bandwidth resource capacities at Loc_q.
E	a set of backhaul links interconnecting base stations and cloudlets.
$B(bs_i)$	the capacity of wireless bandwidth at bs_i.
ue_k, ds_k	A UE and its dataset, where k is a constant with $1 \leq k \leq$ K.
$\|ds_k\|$	The volume of ds_k.
r_m, R , $\|R\|$ and S_m	an FL request, a set of FL requests, the numbers of FL requests and an FL service.
w_m and $\|w_m\|$	the parameter of a model and the size of the model parameter that is transmitted between a UE and its service S_m.
$A_{m,o}$	the oth worker aggregator of FL request r_m.
$\mathcal{UE}_m$	the set of UEs for FL request r_m.
$c^t_{k,q,i}$	the communication cost of ue_k for updating its trained model to $A_{m,o}$ of service S_m
$y_{m,q}$, $c^t_{q,q'}$ and $c^t_{m,o}$	an indicator variable that shows whether S_m is placed to Loc_q, the cost of transmitting a unit amount of data from Loc_q to $Loc_{q'}$ and from $A_{m,o}$ to the master aggregator in S_m
$\mathcal{A}_m$	the set of deployed worker aggregators of r_m.
α_q and c^a_m	the cost of aggregating a UE's local model in Loq_q and the cost of data aggregation for r_m.
γ_q	the amount of computing resource demanded by aggregating a unit amount of data at location Loc_q.
$\mathcal{UE}_{m,o}$	the set of UEs assigned to worker aggregator $A_{m,o}$.
$CR_{m,o,q}$ and $CR'_{m,q}$	the amount of the computing resource which the set of UEs assigned to worker aggregator $A_{m,o}$ are needed and allocated to service S_m.
η_i, ξ_i and ξ_q	the amount of wireless bandwidth resource bs_i uses to receive data, wired bandwidth resource assigned to transfer a unit amount of data and bandwidth resource it is assigned to transfer a unit amount of data at Loc_q.
n_m, n^{min} and n^{max} the number, the minimum and maximum number of worker aggregators of each FL request r_m respectively.	
$cost(n_m)$	the cost obtained by `ApproAG`.
d_{unit}	a *data unit* as a smaller volume of data compared with that of $\|w_m\|$ for dividing the trained local model of each UE into.
$W_{m,o}$	the set of potential locations for worker aggregator $A_{m,o}$.
$c(\cdot,\cdot)$ and $u(\cdot,\cdot)$	the cost and capacity of edge in auxiliary graph.
$c^t_{k,i}$ and $c^t_{i,q}$	the cost of transmitting a unit amount of data from ue_k to bs_i and The cost of transmitting a unit amount of data from bs_i to Loc_q
f_k	the flow obtained for UE ue_k in an auxiliary graph.
$p_{k,i}$, $p_{k,o}$, $p_{m,o,q}$ and $p_{m,q}$	the probabilities of associating ue_k to bs_i, sending UE ue_k's local models to worker aggregator $A_{m,o}$, placing $A_{m,o}$ to loc_q and placing service S_m to location loc_q.
$Y_{k,q}$, Y', Y'', and Z_m	the events that the local models of UE ue_k are assigned to a worker aggregator at Loc_q, ue_k is assigned to a worker aggregator $A_{m,o}$, $A_{m,o}$ is placed to loc_q, and whether service S_m of FL request r_m is placed to location Loc_q.
Y_q	the number of UEs whose trained models are sent to Loc_q for aggregation.
$\mu(\|w_m\|)$	the average volume of trained local model of each UE ue_k.
χ	a factor for scale-up the average model size of each FL request in algorithm `Online`
ζ	a fixed length discretize the range of χ intervals with.
$\mathcal{VL}$ and $\mathcal{VL}'$	The set of the range of χ discretized and active values for χ.
$num_a(v)$	the number of requests before r_m that choose v as the value of χ.
$\mu_c(v)$	the average cost obtained by the algorithm in the number $num_a(v)$ of selections of v.
$radius_v$ and $ball_v$	the confidence radius and the confidence ball of arms in algorithm `Online` .

problem. In the extreme case, some cloudlets have to initialize a certain amount of computing resource to process a very small amount of data, leading to high overhead. On the other hand, if we consider each local model as a basic routing unit, the remaining resource of a cloudlet may not be fully utilized because its remaining resource may not have enough capacity to aggregate the local model. The MEC network may have many such small 'resource slices' that will not be used for any model aggregation. To avoid such cases, we consider a flexible granularity of routing unit of the local models. Let d_{unit} be the *data unit* of a *split block*. The local model of each UE is divided into $\frac{|w_m|}{d_{unit}}$ *split blocks*, assuming that $|w_m|$ is divisible by d_{unit}. The size of each split block of each UE is considered as the demand of a commodity. Notice that the benefit of introducing d_{unit} allows us to improve resource utilization. For example, when cloudlets are under-loaded, we set $d_{unit} \leftarrow |w|$; otherwise, we set d_{unit} to a value smaller than $|w_m|$. The algorithm proceeds as follows.

Auxiliary graph construction: We jointly place worker aggregators to different locations, assign UEs to base stations, and find a location for service S_m. To this end, we partition nodes in V' into different layers, with each layer corresponding to one of the joint decision making processes. Specifically, the nodes in V' of G' are organized into four layers, namely, the UE layer, base station layer, worker aggregator layer and service layer, as shown in Fig. 3.

The UE layer consists of all UE nodes, i.e. we add each $ue_k \in \mathcal{UE}_m$ into V' via $V' \leftarrow V' \cup \{ue_k \mid 1 \leq k \leq |\mathcal{UE}_m|\}$.

The BS layer consists of a set of base stations. For each base station bs_i, we create two *virtual base stations* and add them to V'. Let bs'_i and bs''_i be the two virtual base stations of base station bs_i, then $V' \leftarrow V' \cup \{bs_i,\ bs'_i,\ bs''_i \mid 1 \leq$

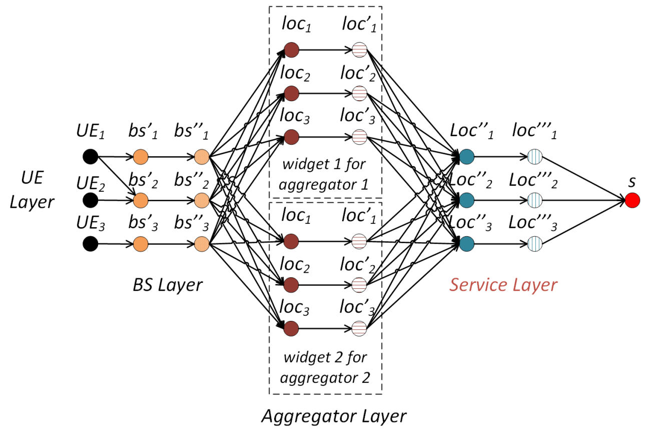


Fig. 3. An example of the auxiliary graph $G' = (V', E')$ with two worker aggregators for an FL request r_m.

$i \leq |\mathcal{BS}|\}$. We then connect bs_i with bs'_i and bs'_i with bs''_i, by adding directed edges $\langle bs_i, bs'_i\rangle$ and $\langle bs'_i, bs''_i\rangle$ to E', i.e., $E' \leftarrow E' \cup \{\langle bs_i, bs'_i\rangle, \langle bs'_i, bs''_i\rangle \mid 1 \leq i \leq |\mathcal{BS}|\}$. The capacity on edge $\langle bs_i, bs'_i\rangle$ is the data unit (each unit has an amount d_{unit} of data) that can be transferred using $B(bs_i)$ amounts of wireless bandwidth of bs_i, i.e.,

$$u(bs_i, bs'_i) = \lfloor B(bs_i)/(d_{unit} \cdot \eta_i)\rfloor. \tag{3}$$

Similarly, the capacity on edge $\langle bs'_i, bs''_i\rangle$ is the maximum number of data units d_{unit} that can be transferred, using the amount B_i of wired bandwidth, i.e., $u(bs'_i, bs''_i) = \lfloor B_i/(d_{unit} \cdot \xi_i)\rfloor$, where ξ_i is the amount of wired bandwidth resource to transfer a unit data to the backhaul of G by base station bs_i. The costs of such edges are set to zeros.

The worker aggregator layer consists of all potential locations of the MEC network for worker aggregators. We create n_m *widgets* with each corresponding to the set of potential locations for each worker aggregator $A_{m,o}$. Let $W_{m,o}$ be such a widget. We then add potential locations with adequate resources to implement $A_{m,o}$ to its widget $W_{m,o}$. Specifically, for each location Loc_q with sufficient computing and bandwidth resources to aggregate and transfer the size $|w_m|$ of trained model, a virtual location node Loc'_q is added to the widget along with Loc_q. That is, $V' = V' \cup \{Loc_q,\ Loc'_q\}$. We then build widget $W_{m,o}$ by adding edges. We add an edge $\langle Loc_q,\ Loc'_q\rangle$ for each Loc_q into the widget. The cost of $\langle Loc_q, Loc'_q\rangle$ is the cost of aggregating d_{unit} data units, i.e., $c(Loc_q, Loc'_q) = d_{unit} \cdot \alpha_q$, and the capacity on the edge thus is

$$u(Loc_q, Loc'_q) = \min\{\lfloor C_q/d_{unit}\gamma_q)\rfloor, \lfloor B_q/(d_{unit}\xi_q)\rfloor\}, \tag{4}$$

which represents the maximum number of data units that can be aggregated by the worker aggregators in location Loc_q. The cost of edge $\langle Loc'_q, d(n_m)\rangle$ is zero, and its capacity is infinity.

In the service layer, we create two virtual nodes Loc''_q and Loc'''_q for each location Loc_q and service S_m. Each pair of nodes is added into a new widget for service S_m. An edge $\langle Loc''_q, Loc'''_q\rangle$ is added to E' and its cost is set to α_q, i.e., $c(Loc''_q, Loc'''_q) = d_{unit} \cdot \alpha_q$. Its capacity is set to the maximum number of data units that can be aggregated by S_m, which is $u(Loc''_q, Loc'''_q) = \min\{\lfloor C_q/(d_{unit} \cdot \gamma_q)\rfloor, \lfloor B_q/(d_{unit} \cdot \xi_q)\rfloor\}$. We finally create a *virtual sink*, denoted by s.

We proceed by interconnecting between the defined layers. First, each UE ue_k in the UE layer is connected to base station bs_i of the BS layer if ue_k is within the transmission range of bs_i. The cost of edge $\langle ue_k, bs_i\rangle$ is the cost of transmitting a unit data from ue_k to bs_i, i.e.,

$$c(ue_k, bs_i) = d_{unit} \cdot c^t_{k,i}, \tag{5}$$

where $c^t_{k,i}$ is the cost of transmitting a unit data from ue_k to bs_i. The capacity of edge $\langle ue_k, bs_i\rangle$ is set to infinity. Second, for each base station bs_i, its virtual base station bs''_i is connected to the source $s(n_m)$ of each widget in the worker aggregation layer. The cost of edge $\langle bs''_i, Loc_q\rangle$ is the cost of transmitting d_{unit} amount of data from bs_i to Loc_q, that is,

$$c(bs''_i, Loc_q) = d_{unit} \cdot c^t_{i,q}, \tag{6}$$

where $c^t_{i,q}$ is the given cost of transmitting a unit data from bs_i to Loc_q that is given in advance. Its capacity is set as the maximum number of data units that can be transferred by the wired bandwidth capacity of base station bs_i if an amount ξ_q of wired bandwidth is allocated to transfer a unit data, i.e.,

$$u(bs_i, Loc_q) = \lfloor B_q/(d_{unit} \cdot \xi_q)\rfloor. \tag{7}$$

Approximation algorithm: We now reduce the joint worker aggregator placement and UE assignment problem for a single FL request to the minimum-cost multi-commodity flow problem in G'. That is, we consider the trained local model of each UE as a commodity with a source ue_k, a sink node s, and a demand of $|w_m|$. The commodity needs to be routed in G' from ue_k to s. To route the commodities of all UEs to their sink node s, we find a splittable multi-commodity flow in G' from each UE to s. Let f_k be the flow obtained for UE ue_k. Flow f_k can be split into different paths in G'. We then adopt a randomized rounding method that converts the obtained flow into probabilities of assigning UEs and placing worker aggregators and service for the FL request admission.

We calculate the probability of assigning a UE ue_k to base station bs_i. In the BS layer, we use $f_k(bs_i)$ to represent the amount of flow that is routed via edges $\langle bs_i, bs'_i\rangle$ and $\langle bs'_i, bs''_i\rangle$. The probability $p_{k,i}$ of associating ue_k to bs_i is

$$p_{k,i} = f_k(bs_i)/(2 \cdot f_k). \tag{8}$$

The data of each UE needs to be sent to a worker aggregator for aggregation. Let $p_{k,o}$ be the probability of sending UE ue_k's trained local model to worker aggregator $A_{m,o}$ for aggregation, then,

$$p_{k,o} = f_k(W_{m,o})/f_k, \tag{9}$$

where $f_k(W_{m,o})$ represents the total amount of flow routed through the edges in widget $W_{m,o}$, i.e., $f_k(W_{m,o}) = \sum_{Loc_q \in W_{m,o}} f_k(Loc_q, Loc'_q)$.

We then calculate the probability of placing a worker aggregator $A_{m,o}$ to location Loc_q. The widget of each $A_{m,o}$ in G' consists of its potential locations. Denote by $W_{m,o}$ the widget for worker aggregator $A_{m,o}$. Let $f_k(Loc_q, W_{m,o})$ be the amount of flow routed via edge $\langle Loc_q, Loc'_q\rangle$ of widget $W_{m,o}$. The probability of placing $A_{m,o}$ to Loc_q is

$$p_{m,o,q} = \frac{\sum_{ue_k \in \mathcal{UE}} f_k(Loc_q, W_{m,o})}{2 \cdot \sum_{Loc_{q'} \in \mathcal{BS} \cup \mathcal{CL}, ue_k \in \mathcal{UE}} f_k(Loc_{q'}, W_{m,o})}. \tag{10}$$

The reason is that the more flow routing via an edge of the widget, the higher probability it will be selected as a location for $A_{m,o}$.

We thirdly calculate the probability of placing service S_m to location Loc_q, according to the amount of flows routed via the service layer of G'. We use $f_k(Loc_q, S_m)$ to denote the amount of flow that is routed via edge $\langle Loc''_q, Loc'''_q \rangle$. Let $p_{m,q}$ be the probability of placing service S_m to location Loc_q, then,

$$p_{m,q} = \frac{\sum_{ue_k \in \mathcal{UE}} f_k(Loc_q, S_m)}{(\frac{|w_{max}|}{d_{unit}} \cdot \sum_{\substack{Loc_{q'} \in \mathcal{BS} \cup \mathcal{CL} \\ ue_k \in \mathcal{UE}}} f_k(Loc_{q'}, S_m)}. \tag{11}$$

We finally assign each ue_k to bs_i with probability $p_{k,i}$, and place worker aggregator $A_{m,o}$ and service S_m to location Loc_q with probabilities $p_{m,o,q}$ and $p_{m,q}$, respectively. The detailed steps of the algorithm are given in algorithm `ApproAG`.

Algorithm 2 `ApproAG`

```
Input: An MEC network $G$, an FL request $r_m$, and $n_m$ worker aggregators for $r_m$.
Output: An implementation of FL request $r_m$.
 1: Construct an auxiliary graph $G'$ as shown in Fig. 3; and set the costs and capacities of each edge accordingly;
 2: Consider the local model $w_m$ of each UE $ue_k$ as a commodity with demand $|w_m|/d_{unit}$;
 3: Find a splittable flow for each UE from $ue_k$ to $s$ in the constructed auxiliary graph $G'$, and let $f_k$ be the flow for the local model $w_m$ if each UE $ue_k$;
 4: Calculate the probabilities of assigning UEs to base stations, worker aggregators and service $S_m$ to locations following Equations (8), (10), and (11);
 5: Randomly assign UE $ue_k$ to $bs_i$ according to probability $p_{k,i}$;
 6: For each worker aggregator, randomly place it in location $Loc_q$ with probability $p_{m,o,q}$;
 7: Randomly place service $S_m$ to location $Loc_q$ with probability $p_{m,q}$;
 8: for Each $ue_k$ do
 9:     In the UE and BS layers of the auxiliary graph, move each split of $f_k$ to the randomly selected base station;
10:     In the worker aggregator layer of $G'$, move each split of $f_k$ to an placed worker aggregator with the minimum cost;
11:     In the service layer, move each split of $f_k$ to the location where its service is placed;
12: end for
13: Transfer the obtained unsplittable flow $f'$ into UE assignments and worker aggregator, service placement decisions;
```

4.3 Algorithm Analysis

We first give the following properties of the proposed approximation algorithm.

Property 1: $p_{k,i}$ is lower bounded by $\frac{d_{unit}}{2 \cdot |w_m|}$. Recall that one unit of flow in the auxiliary graph G' represents an amount $|w_{min}|$ of local model. We then have a lower bound on the probability

$$f_k(bs_i)/(2 \cdot f_k) \geq d_{unit}/(2 \cdot |w_m|), \tag{12}$$

since edge $\langle bs'_i, bs''_i \rangle$ has at least one unit of flow routed via itself, if its flow is non-negative.

Property 2: The probability $p_{m,o,q}$ of placing worker aggregator $A_{m,o}$ to location Loc_q is bounded by $d_{unit}/(|w_m| \cdot |\mathcal{UE}|)$, i.e.,

$$p_{m,o,q} = f_k(Loc_q, W_{m,o})/2 \sum_{Loc_{q'} \in \mathcal{BS} \cup \mathcal{CL}} f_k(Loc_{q'}, W_{m,o}) \tag{13}$$

$$\geq d_{unit}/(2 \cdot |w_m| \cdot |\mathcal{UE}|),$$

if the traffic of all UEs is routed via a single widget of G'.

Property 3: $p_{m,q}$ is lower bounded by $\frac{d_{unit}}{|w_m| \cdot |\mathcal{UE}|}$. The minimum size of local model that is routed via an edge $\langle Loc''_q, Loc'''_q \rangle$ is d_{unit} while the total amount of data routed through the service layer of G' is $|w_m| \cdot |\mathcal{UE}|$, then

$$p_{m,q} \geq d_{unit}/(2 \cdot |w_m| \cdot |\mathcal{UE}|). \tag{14}$$

Lemma 1. Assuming that $B(bs_{min})/(\eta_i \cdot |w_m|) \geq |\mathcal{UE}|/\delta$ and $\min\{B_i \mid 1 \leq i \leq |\mathcal{BS}|\}/(\xi_i \cdot |w_m|) \geq |\mathcal{UE}|/\kappa$, the wireless and wired bandwidth resource capacities of each base station may be violated with a low probability of $\frac{((2 \cdot |w_m|)/(d_{unit})-1)}{(1/\delta-1)^2|\mathcal{UE}|}$ and $\frac{((2 \cdot |w_m|)/(d_{unit})-1)}{(1/\kappa-1)^2|\mathcal{UE}|}$, respectively, where δ and κ are given constants in the range of $(0, 1]$.

Proof: We first show the bound on the violation of wireless bandwidth of base station bs_i. Recall that each UE ue_k is associated with a base station bs_i with probability $p_{k,i}$. Let $X_{k,i}$ be the event that ue_k is assigned to bs_i. We have $X_{k,i} = 1$ with probability $p_{k,i}$. Let X_i be the number of UEs assigned to base station bs_i, we have $X_i = \sum_{ue_k \in \mathcal{UE}} X_{k,i}$, and the expectation $E(X_i)$ of X_i is $E(X_i) = \sum_{ue_k \in \mathcal{UE}} E(X_{k,i}) = \sum_{ue_k \in \mathcal{UE}} p_{k,i}$. Its variance $Var(X_i)$ then is $Var(X_i) = E((X_i)^2) - (E(X_i))^2$. Since UEs are assigned to base stations randomly, the wireless bandwidth capacity on each base station may be violated, and the probability of such a violation is $Pr(X_i \geq B(bs_i)/(\eta_i \cdot |w_m|))$, which can be rewritten as $Pr(|X_i - E(X_i)| \geq B(bs_i)/(\eta_i \cdot |w_m|) - E(X_i))$. Let $B(bs_{min})$ denote the minimum wireless bandwidth capacity on base stations, following Chebyshev's inequality [29], we have

$$Pr(X_i \geq \frac{B(bs_i)}{\eta_i \cdot |w_m|}) \leq \frac{E((X_i)^2) - (E(X_i))^2}{(B(bs_i)/(\eta_i \cdot |w_m|) - \sum_{ue_k \in \mathcal{UE}} p_{k,i})^2}$$
$$\leq \frac{\sum_{ue_k \in \mathcal{UE}} p_{k,i}(1)^2 - (\sum_{ue_k \in \mathcal{UE}} p_{k,i})^2}{(B(bs_i)/(\eta_i \cdot |w_m|) - \sum_{ue_k \in \mathcal{UE}} p_{k,i})^2}$$
$$\leq \frac{\sum_{ue_k \in \mathcal{UE}} (\frac{1}{p_{k,i}} - 1)}{(\frac{B(bs_{min})}{p_{k,i}\eta_i \cdot |w_m|} - |\mathcal{UE}|)^2}$$
$$\leq \frac{\sum_{ue_k \in \mathcal{UE}} (\frac{1}{p_{k,i}} - 1)}{(\frac{B(bs_{min})}{\eta_i \cdot |w_m|} - |\mathcal{UE}|)^2}, \text{ since } p_{k,i} \leq 1$$
$$\leq (\sum_{ue_k \in \mathcal{UE}} (1/p_{k,i} - 1))/((1/\delta - 1)^2 |\mathcal{UE}|^2), \tag{15}$$

assuming that $\frac{B(bs_{min})}{\eta_i \cdot |w_m|} \geq \frac{1}{\delta} \sum_{ue_k \in \mathcal{UE}} 1 = \frac{|\mathcal{UE}|}{\delta}$ with $0 < \delta \leq 1$, and the rationale of this assumption is that a base station can host a single UE; it however may not be able to host all UEs. Since $p_{k,i} \geq d_{unit}/(2 \cdot |w_m|)$, we have

$$(15) \leq \frac{\sum_{ue_k \in \mathcal{UE}} (\frac{1}{p_{k,i}} - 1)}{(\frac{1}{\delta} - 1)^2 |\mathcal{UE}|^2} \leq \frac{|\mathcal{UE}|(\frac{2 \cdot |w_m|}{d_{unit}} - 1)}{(\frac{1}{\delta} - 1)^2 |\mathcal{UE}|^2}$$
$$\leq \frac{(\frac{2 \cdot |w_m|}{d_{unit}} - 1)}{(\frac{1}{\delta} - 1)^2 |\mathcal{UE}|}. \tag{16}$$

Therefore, the probability of violating the wireless bandwidth capacity on each base station bs_i is $\frac{((2 \cdot |w_m|)/(d_{unit})-1)}{(1/\delta-1)^2|\mathcal{UE}|}$. This probability is very small if many UEs participate in FL.

We then analyze the probability of violating the wired bandwidth resource capacity on each base station bs_i. The derivation is similar as that in Inequalities (15) and (16). We only need to replace $B(bs_i)$ with B_i, which is omitted for the sake conciseness. We thus conclude that $\frac{((2\cdot|w_m|)/(d_{unit})-1)}{(1/\kappa-1)^2|\mathcal{UE}|}$, where $\frac{\min\{B_i \mid 1\leq i\leq|\mathcal{BS}|\}}{\xi_i\cdot|w_m|} \geq \frac{|\mathcal{UE}|}{\kappa}$. □

Lemma 2. Assuming that $\frac{C_q}{\gamma_q\cdot|w_m|} \geq |\mathcal{UE}|/\Psi$, the computing resource capacity of each location $Loc_q \in \mathcal{BS}\cup\mathcal{CL}$ may be violated with a very low probability of $\frac{2|\mathcal{UE}|^2|V||w_{max}|^2/d_{unit}}{(\frac{|\mathcal{UE}|}{\Psi}-|\mathcal{UE}|^2|V|\frac{|w_{max}|}{d_{unit}}-|\mathcal{UE}\cup\mathcal{CL}|)^2}$, where Ψ is a given constant.

Proof: We analyze the probability of violating the computing resource capacity on each location Loc_q. Recall that the total amount of computing resource assigned to aggregate local models of UEs depends on the number of UEs that are assigned to location Loc_q and whether service S_m is placed to Loc_q. Let $Y_{k,q}$ be an event that the local model of UE ue_k is assigned to a worker aggregator at Loc_q. This event depends on two sub-events of (1) ue_k is assigned to a worker aggregator $A_{m,o}$, which is denoted by random variable Y', and (2) $A_{m,o}$ is placed to Loc_q, represented by random variable Y''. We then have the expectation of $Y_{k,q}$ by

$$\begin{aligned} E(Y_{k,q}) &= E(Y'' \mid Y'=1)\\ &= \sum\nolimits_{Loc_q\in\mathcal{BS}\cup\mathcal{CL}} p_{m,o,q}\cdot p_{k,o}, \end{aligned} \tag{17}$$

considering that events Y' and Y'' are two independent events.

Let Y_q be the number of UEs whose data is sent to Loc_q for aggregation. We have

$$E(Y_q) = \sum\nolimits_{ue_k\in\mathcal{UE}} Y_{k,q} = \sum\nolimits_{ue_k\in\mathcal{UE},Loc_q\in\mathcal{BS}\cup\mathcal{CL}} p_{m,o,q}\cdot p_{k,o}.$$

Denote by Z_m the event that whether service S_m of FL request r_m is placed to location Loc_q. Its expectation is $E(Z_m) = \sum_{Loc_q\in\mathcal{BS}\cup\mathcal{CL}} p_{m,q}$. Clearly, if all the worker aggregators and S_m are placed to location Loc_q, the computing capacity of Loc_q may be violated maximally. Following the similar derivation in Ineq. (15), the probability of violating the computing resource capacity on each Loc_q is

$$\begin{aligned} &Pr(Y_{k,q}+Z_m \geq \frac{C_q}{\gamma_q\cdot|w_m|})\\ &\leq \frac{Var(Y_{k,q})+Var(Z_m)}{(\frac{C_q}{\gamma_q\cdot|w_m|}-E(Y_{k,q})-E(Z_m))^2}\\ &\leq \frac{E((Y_{k,q})^2)-(E(Y_{k,q}))^2+E((Z_m)^2)-(E(Z_m))^2}{(\frac{C_q}{\gamma_q\cdot|w_m|}-\sum_{ue_k,Loc_q}p_{m,o,q}\cdot p_{k,o}-\sum_{Loc_q}p_{m,q})^2}\\ &\leq \frac{\sum_{ue_k,Loc_q}p_{m,o,q}\cdot p_{k,o}(1)^2-(\sum_{ue_k,Loc_q}p_{m,o,q}\cdot p_{k,o})^2}{(\frac{C_q}{\gamma_q\cdot|w_m|}-\sum_{ue_k,Loc_q}p_{m,o,q}\cdot p_{k,o}-\sum_{Loc_q}p_{m,q})^2}\\ &\quad+\frac{\sum_{Loc_q}p_{m,q}(1)^2-(\sum_{Loc_q}p_{m,q})^2}{(\frac{C_q}{\gamma_q\cdot|w_m|}-\sum_{ue_k,Loc_q}p_{m,o,q}\cdot p_{k,o}-\sum_{Loc_q}p_{m,q})^2}\\ &\leq \frac{\sum_{ue_k,Loc_q}(p_{m,o,q}\cdot p_{k,o}-(p_{m,o,q}\cdot p_{k,o})^2)}{(\frac{C_q}{\gamma_q\cdot|w_m|}-\sum_{ue_k,Loc_q}p_{m,o,q}\cdot p_{k,o}-\sum_{Loc_q}p_{m,q})^2}\\ &\quad+\frac{\sum_{Loc_q}(p_{m,q}-(p_{m,q})^2)}{(\frac{C_q}{\gamma_q\cdot|w_m|}-\sum_{ue_k,Loc_q}p_{m,o,q}\cdot p_{k,o}-\sum_{Loc_q}p_{m,q})^2} \end{aligned}$$

$$\begin{aligned} &\leq \frac{\sum_{ue_k,Loc_q}(\frac{1}{p_{m,o,q}\cdot p_{k,o}}-1)}{(\frac{C_q}{\gamma_q\cdot|w_m|}-|\mathcal{UE}|\cdot|V|-|\mathcal{UE}\cup\mathcal{CL}|)^2}\\ &\quad+\frac{\sum_{Loc_q}(\frac{1}{p_{m,q}}-1)}{(\frac{C_q}{\gamma_q\cdot|w_m|}-\sum_{ue_k,Loc_q}\frac{p_{m,o,q}\cdot p_{k,o}}{p_{m,q}}-|\mathcal{UE}\cup\mathcal{CL}|)^2} &(18)\\ &\leq \frac{\sum_{ue_k,Loc_q}(\frac{1}{p_{m,o,q}\cdot p_{k,o}}-1)}{(\frac{C_q}{\gamma_q\cdot|w_m|}-|\mathcal{UE}|\cdot|V|-|\mathcal{UE}\cup\mathcal{CL}|)^2}\\ &\quad+\frac{\sum_{Loc_q}(\frac{1}{p_{m,q}}-1)}{(\frac{C_q}{\gamma_q\cdot|w_m|}-\frac{|\mathcal{UE}|\cdot|w_{max}|}{d_{unit}}\sum_{ue_k,Loc_q}p_{m,o,q}p_{k,o}-|\mathcal{UE}\cup\mathcal{CL}|)^2} &(19)\\ &\leq \frac{\sum_{ue_k,Loc_q}(\frac{1}{p_{m,o,q}\cdot p_{k,o}}-1)}{(\frac{C_q}{\gamma_q\cdot|w_m|}-|\mathcal{UE}|^2|V|\frac{|w_{max}|}{d_{unit}}-|\mathcal{UE}\cup\mathcal{CL}|)^2}\\ &\quad+\frac{\sum_{Loc_q}(\frac{1}{p_{m,q}}-1)}{(\frac{C_q}{\gamma_q\cdot|w_m|}-|\mathcal{UE}|^2|V|\frac{|w_{max}|}{d_{unit}}-|\mathcal{UE}\cup\mathcal{CL}|)^2}\\ &\leq \frac{|\mathcal{UE}|^2|V||w_{max}|^2/d_{unit}}{(|\mathcal{UE}|/\Psi-|\mathcal{UE}|^2|V||w_{max}|/d_{unit}-|\mathcal{UE}\cup\mathcal{CL}|)^2}\\ &\quad+\frac{|V||\mathcal{UE}||w_{max}|/d_{unit}}{(|\mathcal{UE}|/\Psi-|\mathcal{UE}|^2|V||w_{max}|/d_{unit}-|\mathcal{UE}\cup\mathcal{CL}|)^2},\\ &\quad\text{since } C_q/(\gamma_q|w_m|)\geq|\mathcal{UE}|/\Psi &(20)\\ &\leq \frac{2|\mathcal{UE}|^2|V||w_{max}|^2/d_{unit}}{(|\mathcal{UE}|/\Psi-|\mathcal{UE}|^2|V||w_{max}|/d_{unit}-|\mathcal{UE}\cup\mathcal{CL}|)^2}. &(21) \end{aligned}$$

Note that the derivation from Ineq. (18) to (19) is due to the fact that the minimum non-negative flow goes to edge $\langle Loc_q'', Loc_q'''\rangle$ is 1.

The violation of bandwidth resources at a location Loc_q can be derived similarly; omitted. □

Theorem 1. The approximation ratio of algorithm ApproAG is $2\epsilon|w_m|/(|w_m|-d_{unit})$ with probability $1/(\epsilon|\mathcal{UE}|)$, where ϵ is constant with $0<\epsilon\leq 1$.

Proof: Considering that algorithm ApproAG is a randomized algorithm, we show that its approximation ratio is α with high probability. That is, $Pr(C \geq \alpha\cdot OPT)$, where C is the cost by ApproAG and OPT is the optimal solution to the problem. Let OPT' be the fractional solution of OPT that splits the flow to route along different routing paths in the auxiliary graph G'. We then have $Pr(C\geq\alpha\cdot OPT)\leq Pr(C\geq\alpha\cdot OPT')$ since $OPT'\leq OPT$. Due to the Markov's inequality,

$$Pr(C\geq\alpha OPT)\leq Pr(C\geq\alpha OPT')\leq E(C)/(\alpha OPT'),$$

where $E(C)$ is the expected cost of algorithm ApproAG. Algorithm ApproAG consists of two stages: (1) splitting the trained local model w_m of each UE into $\frac{|w_m|}{d_{unit}}$ numbers of data units; and (2) finding a splittable flow in auxiliary graph G', and then assigning different splits of each UE to the selected base station and worker aggregator. These two stages push-up the costs in comparison with the optimal fractional solution OPT' of the problem.

For Stage (1), we divide the local model of each UE into numbers of data units with each having a volume of d_{unit}. However, the optimal fractional solution OPT' to the problem may consider smaller split blocks (smaller values for d_{unit}), because smaller split blocks can fit into locations with low costs and small amount of available resources. We thus have

$$\begin{aligned} &Pr(C \geq \alpha \cdot OPT') \leq E(C)/(\alpha \cdot OPT') \\ &\leq \frac{d_{unit}/(2|w_m||\mathcal{UE}|)OPT'}{\alpha \cdot OPT'} = \frac{d_{unit}/(2|w_m|\mathcal{UE}|)}{\alpha}, \quad (22) \end{aligned}$$

due to the bounds of probabilities $p_{i,k}$, $p_{m,o,q}$, and $p_{m,q}$. This is due to: (1) the local model w_m of each UE is split into $\frac{|w_m|}{d_{unit}}$ data units; and (2) the number of data units of each UE may be split into different paths of the auxiliary graph. Let OPT be the optimal solution to the original problem.

The obtained splittable flow in Stage (2) may be infeasible. To make it become feasible, we reallocate the split flows of each UE to the associated base station, placed worker aggregators and service S_m. Such reallocation of the split flows push up the costs of routing the trained local models of UEs in $\mathcal{UE}$. Denote by C the total cost of the solution delivered by approximation algorithm ApproAG. For simplicity, we distinguish such the cost increase into the following three cases: (i) the cost push-ups due to moving partial data units of a UE to its associated base station bs_i; (ii) the cost push-ups because of moving partial data units of a UE to its assigned worker aggregator in location Loc_q; and (iii) the push-ups incurred by moving data units to its assigned service S_m in location $Loc_{q'}$.

Case (i). Following the construction of the auxiliary graph G', each UE ue_k may has its data units being split into $|\mathcal{BS}|$ base stations in the worst case. Let bs_i be the selected base station for ue_k by algorithm ApproAG. It is selected with probability $p_{k,i}$. Let C_1' be the cost incurred by the splittable flow f. In the worst case, for each ue_k, its $\frac{|w_m|}{d_{unit}} - 1$ data units are moved to bs_i. Then, Eq. (22) can be rewritten as

$$\begin{aligned} &Pr(C \geq \alpha \cdot OPT') \leq E(C)/(\alpha \cdot OPT') \\ &\leq \frac{(|w_m|/d_{unit} - 1)d_{unit}/(2 \cdot |w_m| \cdot |\mathcal{UE}|)OPT'}{\alpha \cdot OPT'} \\ &= (|w_m| - d_{unit})/(2 \cdot |w_m| \cdot |\mathcal{UE}|) \cdot \alpha. \quad (23) \end{aligned}$$

Cases (ii) and Case (iii) can be discussed similarly, omitted.

Let $|w_m| \geq 2 \cdot d_{unit}$ and $\alpha = 2\epsilon|w_m|/(|w_m| - d_{unit})$, we have $Pr(C \geq \alpha \cdot OPT) \leq 1/(\epsilon|\mathcal{UE}|)$. Namely, the approximation ratio of algorithm ApproAG is $\frac{2\epsilon|w_m|}{|w_m|-d_{unit}}$ with probability $\frac{1}{\epsilon|\mathcal{UE}|}$, where ϵ is a constant with $0 < \epsilon \leq 1$. □

5 An Online Learning Algorithm with Uncertain Model

In this section we propose an online learning algorithm for the online worker aggregator placement and UE assignment problem in an MEC network.

5.1 Basic Idea

If the model size is uncertain, a feasible method is to implement each FL request r_m according to the average model size. Let $\mu(|w_m|)$ be the average size of local model of each UE of request r_m. That is, we modify algorithm OptFWK by first calculating the minimum number n^{min} of worker aggregators, according to the average value $\mu(|w_m|)$ instead of the actual value $|w_m|$, and then considering each commodity in ApproAG with a demand of $|w_m|$. However, the actual model size could be much larger than the average one. This may lead to a violation on the resource capacity at a location in the MEC network. To avoid such a resource violation, we scale-up the average model size of each FL request by a factor of χ, where χ is a real value in the range of $[1, \frac{|w_{max}|}{\mu(|w_m|)}]$. Specifically, we calculate n^{min} in OptFWK by

$$\begin{aligned} &n^{min} \\ &= \max\{1, (\gamma_q(1+\chi)\mu(|w_m|)|\mathcal{UE}_m|)/(\max\{C_q \mid \forall Loc_q\}), \\ &(\xi_q \cdot (1+\chi)\mu(|w_m|) \cdot |\mathcal{UE}_m|)/\max\{B_q \mid \forall Loc_q\}\} \quad (24) \end{aligned}$$

Also each commodity of UE ue_k of r_m has a demand of

$$(1+\chi) \cdot \mu(|w_m|)/d_{unit}. \quad (25)$$

To determine the value of χ, we can simply adopt a fix value for χ and perform a one-time optimization. This however may lead to the case where the actual performance of the algorithm may deviate far from the optimal one if the actual model size of each FL request deviates significantly from its expected one. We thus learn the value of χ adaptively as requests are arriving into the system, by adopting a multi-armed bandit-based online learning method with a fully customized zooming algorithm [39].

5.2 Online Learning Algorithm

We devise an online learning algorithm based on a customized zooming algorithm, by adjusting the scale up factor χ according to the current mean cost of implementing FL requests. Specifically, we have infinite choices of χ in its range of $[1, |w_{max}|/|w_{min}|]$. Besides, there is not a clear monotonic relationship between χ and the mean cost of implementing FL requests. To find an appropriate value for χ, we discretize the range of χ into ζ intervals with an equal length of $len = (|w_{max}|/|w_{min}| - 1)/\zeta$. Letting i be an integer in range $[1, \zeta]$, we obtain a finite set of candidate values for χ, i.e., $\mathcal{VL} = \{1 + (i-1) \cdot len \mid 1 \leq i \leq \zeta\}$.

Given the set $\mathcal{VL}$ of candidate values for χ, we adaptively find a proper value for the scale-up factor χ in the candidate value set $\mathcal{VL}$, by considering each value in $\mathcal{VL}$ as an *arm* of the MAB method.

We design a novel *activation and selection* scheme to decide which arms can be activated and selected. Specifically, we first activate a set of arms in $\mathcal{VL}$, which is referred to as *activated set* $\mathcal{VL}'$. The arms in $\mathcal{VL}'$ are the representatives of the arms in set $\mathcal{VL}$. We then select an arm from the activated set $\mathcal{VL}'$. The benefit of such an activation and selection approach is to group arms with similar costs, and thereby reducing the arm selection space and enhancing the efficiency.

Arm activation: To this end, we keep a set of active arms for the values in χ. Let $\mathcal{VL}'$ be the set of active arms. Once a value in $\mathcal{VL}$ is activated, it is put into set $\mathcal{VL}'$. Values of χ with small differences may incur similar implementation cost of FL request r_m. The activation of such values with similar costs can reduce the efficiency of the algorithm. We thus define a *confidence radius* for each arm $v \in \mathcal{VL}'$, to active a value v' in $\mathcal{VL}$ and not in the confidence radius of v into candidate value set $\mathcal{VL}'$.

For a value v and an FL request r_m, we use $num_a(v)$ to denote the number of requests before r_m that have chosen v as their value of χ. Let $\mu_c(v)$ be the average cost obtained by the algorithm in the number $num_a(v)$ of selections of v, and $|R|$ the number of FL requests. The confidence radius is defined as

$$radius_v = \sqrt{2|R|/(num_a(v)+1)}, \tag{26}$$

and the confidence ball of v by $ball_v = \{x \in \mathcal{VL} \mid |\mu_c(v) - \mu_c(x)| \leq radius_v\}$, which is a condition for the Lipschitz bandits, that is, $|\mu_c(v) - \mu_c(x)| \leq radius_v$.

Ideally, the radii of arms in $\mathcal{VL}'$ should cover every value in $\mathcal{VL}$. Initially, the coverage radius of each arm is large, since it has not been selected (meaning that $num_a(v) = 0$). As requests arrive, the confidence ball of each arm is shrinking, leaving some values in $\mathcal{VL}$ uncovered. To ensure all values in $\mathcal{VL}$ covered by the arms in $\mathcal{VL}'$, we add those uncovered values into set $\mathcal{VL}'$.

Arm selection: Having obtained a set of active arms, we select an arm from $\mathcal{VL}'$ on the arrival of each FL request r_m. Ideally, we would like to select an arm that not only achieves a lower cost but also covers more values in $\mathcal{VL}$. We then select the arm with the highest value of $1/\mu_c(v) + 2 \cdot radius_v$, and its corresponding value is selected for χ. The detailed steps of the proposed algorithm is shown in **Algorithm** 3.

Algorithm 3 ONLINE

```
Input: An MEC network $G$, a set $R$ of FL requests that arrives one by one without the knowledge of future arrivals and their model sizes.
Output: Implementation of each arrived FL request.
1:  Discretize the range of $\chi$ into $\zeta$ intervals with a fixed length of $len = (|w_{max}|/|w_{min}| - 1)/\zeta$, and let $\mathcal{VL}$ be the obtained finite set of values;
2:  $\mathcal{VL}' \leftarrow \emptyset$; /*The set of values that are selected from $\mathcal{VL}$*/
3:  Greedily select a number of values in set $\mathcal{VL}$ that can cover all values in $\mathcal{VL}$ by their radii, and add them to set $\mathcal{VL}'$;
4:  for each arrived request $r_m \in R$ do
5:      Select a value from set $\mathcal{VL}'$ with the highest rank of $1/\mu_c(c) + 2 \cdot radius_v$;
6:      Invoke algorithm FWK with $n_{min}$ being set following Eq. (24) and the demand of each commodity set by Eq. (25);
7:      Update the number of selections of each value in $\mathcal{VL}'$;
8:      if There are values that are not covered by the ones in $\mathcal{VL}'$ then
9:          Add these uncovered values in $\mathcal{VL}'$;
10:     end if
11: end for
```

Theorem 2. Assuming the costs satisfy the Lipschitz condition, i.e., $|E(C(r_m)) - E(C(r_{m'}))| \leq L \cdot |C(r_m) - C(r_{m'})|$, the regret of algorithm ONLINE is $\big(\frac{2\epsilon|w_{min}|}{|w_{min}|-d_{unit}} \cdot \frac{1}{\epsilon|\mathcal{UE}|} + L + 1\big)2\sqrt{2R}$, where L is a given constant.

Proof: Let H_m be the random process of the model size of FL request r_m, and its average value is $\mu(|w_m|)$. Let C_χ be the solution cost delivered by algorithm ONLINE. Denote by m' the index of a previously implemented request $r_{m'}$. Assuming that FL request r_m is the to-be-implemented one, the regret ret_m of dynamically adjusting the scale up factor χ after its admission is

$$ret_m = E[C(r_m))] - \min_{1 \leq m' \leq m}\{C_{opt}(r_{m'})\}. \tag{27}$$

To obtain an upper bound on ret_m, we consider the following three cases: (i) $|w_m| = (1+\chi)\mu(|w_m|)$; (ii) $|w_m| < (1+\chi)\mu(|w_m|)$; and (iii) $|w_m| > (1+\chi)\mu(|w_m|)$.

Case (i). We implement request r_m by its actual model size. Let $C^{opt}(r_m)$ and $C^{opt}_\chi(r_m)$ be the optimal solutions for request being implemented according to model sizes of $|w_m|$ and $(1+\chi)\cdot\mu(|w_m|)$. Following Theorem 1, we have

$$\frac{C(r_m)}{C^{opt}(r_m)} = \frac{C_\chi(r_m)}{C^{opt}_\chi(r_m)} \leq \frac{2\epsilon(1+\chi)\mu(|w_m|)}{(1+\chi)\mu(|w_m|) - d_{unit}} \tag{28}$$

with probability $1/(\epsilon|\mathcal{UE}|)$. From Ineq. (28), we have

$$\frac{C(r_m) - C^{opt}(r_m)}{C^{opt}(r_m)} \leq \frac{2\epsilon(1+\chi)\mu(|w_m|)}{(1+\chi)\mu(|w_m|) - d_{unit}} - 1, \tag{29}$$

which means that

$$C(r_m) \leq \Big(\frac{2\epsilon(1+\chi)\mu(|w_m|)}{(1+\chi)\mu(|w_m|) - d_{unit}} - 1\Big)C^{opt}(r_m) + C^{opt}(r_m). \tag{30}$$

Assuming that request $r_{m'}$ has the lowest implementation cost, the regret for Case (i) is

$$\begin{aligned}
&E(C(r_m)) - C^{opt}(r_{m'}) \\
&\leq E\big(\big(\frac{2\epsilon(1+\chi)\mu(|w_m|)}{(1+\chi)\mu(|w_m|) - d_{unit}} - 1\big)C^{opt}(r_m) + C^{opt}(r_m)\big) \\
&\qquad - C^{opt}(r_{m'}) \\
&\leq \big(\frac{2\epsilon(1+\chi)\mu(|w_m|)}{(1+\chi)\mu(|w_m|) - d_{unit}} - 1\big)C^{opt}(r_m)\frac{1}{\epsilon|\mathcal{UE}|} + \\
&\qquad E(C^{opt}(r_m)) - C^{opt}(r_{m'}).
\end{aligned} \tag{31}$$

Due to the Lipschitz condition, Ineq. (31) can be rewritten as

$$\begin{aligned}
(31) &\leq \big(\frac{2\epsilon(1+\chi)\mu(|w_m|)}{(1+\chi)\mu(|w_m|) - d_{unit}} - 1\big)C^{opt}(r_m)\frac{1}{\epsilon|\mathcal{UE}|} + \\
&\qquad E(C^{opt}(r_m)) - C^{opt}(r_{m'}) \qquad (32) \\
&\leq \big(\frac{2\epsilon(1+\chi)\mu(|w_m|)}{(1+\chi)\mu(|w_m|) - d_{unit}} - 1\big)C^{opt}(r_m)\frac{1}{\epsilon|\mathcal{UE}|} + \\
&\qquad L \cdot C^{opt}(r_m) + E(C^{opt}(r_{m'})) \\
&\leq \big(\frac{2\epsilon(1+\chi)\mu(|w_m|)}{(1+\chi)\mu(|w_m|) - d_{unit}} - 1 + L\big)C^{opt}(r_m)\frac{1}{\epsilon|\mathcal{UE}|} + \\
&\qquad + E(C^{opt}(r_{m'})). \qquad (33)
\end{aligned}$$

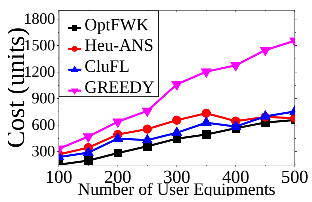
In each round of algorithm Online, each activated value is treated as a representative of the values within its radius $radius_v$. In the worst case, each value is activated and the value with the highest cost is selected in a ball of each arm (value) v may be selected. This means

$$\begin{aligned}
&(33) \\
&\leq \big(\frac{2\epsilon(1+\chi)\mu(|w_m|)}{(1+\chi)\mu(|w_m|) - d_{unit}} - 1 + L\big)2\sqrt{2R}\frac{1}{\epsilon|\mathcal{UE}|} + 2\sqrt{2R} \\
&\leq \big(\frac{2\epsilon(1+\chi)\mu(|w_m|)}{(1+\chi)\mu(|w_m|) - d_{unit}} \cdot \frac{1}{\epsilon|\mathcal{UE}|} + L + 1\big)2\sqrt{2R}.
\end{aligned} \tag{34}$$

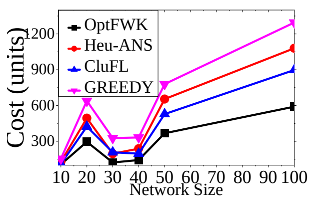
Case (ii), It can be seen that each request is implemented by adopting a larger model size compared instead of its actual model size. By Theorem 1, we have $C(r_m)/C^{opt}(r_m) < C_\chi(r_m)/C^{opt}_\chi(r_m)$. The derivation from inequalities (28) to (33) thus holds.

Case (iii). We have $C_\chi(r_m)/C^{opt}_\chi(r_m) < C(r_m)/C^{opt}(r_m) \leq 2\epsilon|w_m|/(|w_m| - d_{unit})$. Following the similar derivation from (28) to (33), we have $ret_m \leq \big((2\epsilon|w_{min}|)/((|w_{min}| - d_{unit})\cdot\epsilon|\mathcal{UE}|) + L + 1\big)2\sqrt{2R}$.

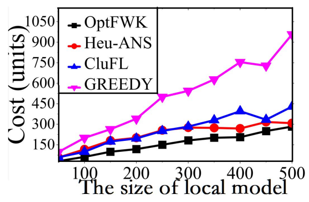
In summary, we have a regret of $\big(\frac{2\epsilon|w_{min}|}{|w_{min}|-d_{unit}} \cdot \frac{1}{\epsilon|\mathcal{UE}|} + L + 1\big)2\sqrt{2R}$. □



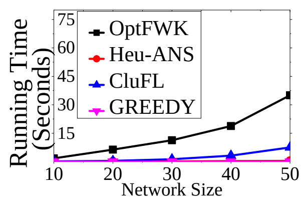
(a) The average cost by varying the number UEs of each FL request.



(b) The average cost by varying network sizes ($|\mathcal{BS} \cup \mathcal{CL}|$).

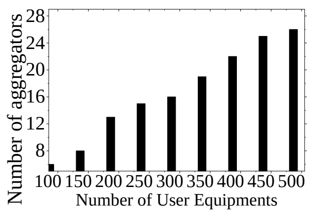


(c) The total cost by varying the size of local model.

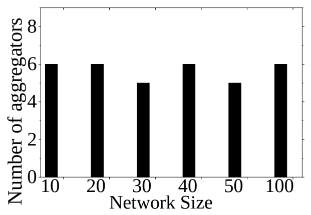


(d) Running time (seconds).

Fig. 4. The performance of algorithms `OptFWK`, `Heu-ANS`, `CluFL`, and `GREEDY`



(a) The average number of worker aggregators by varying the number UEs of each FL request.



(b) The average number of aggregators by varying network sizes.

Fig. 5. The impact of the number of UEs and the number of worker aggregators on the performance of algorithm `OptFWK`.

6 Experiments

In this section, we evaluated the performance of the proposed algorithms by both simulations and experiments in a testbed.

6.1 Parameter Settings

We consider an MEC network generated by tool GT-ITM [14]. 20% of locations in each network contain cloudlets or base stations serving as potential locations for worker aggregators of FL requests. The computing capacity of each potential location is randomly drawn from [1,000 MHz, 5,000 MHz] [32]. The wireless and wired bandwidth capacity on base stations and cloudlets are drawn from [1 Mpbs, 5 Mbps] and [1 Mpbs, 20 Mbps] [32], respectively. The number of FL requests varies from 5 to 30. The size of the trained model parameter of each UE is randomly drawn in the range of [0.05 MB, 0.5 MB] [30], which is calculated and validated based on existing neural networks, such as CNN, ResNet, etc. The cost of transmitting 1MB of data is drawn from $[0.1, 0.3]$ $units$ via wireless links, and $[0.05, 0.2]$ $units$ in wired links. The cost of aggregating one MB of data is withdrawn from $[0.1, 0.3]$ $units$ [1].

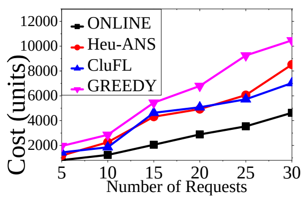
We evaluated the proposed algorithms against the listed three comparison algorithms:

- The first one is a heuristic algorithm in [36], referred to as `Heu-ANS`. It greedily finds a location for each worker aggregator with the minimum weighted sum of the communication latency and energy consumption of edge devices.
- The second one is a hierarchical FL framework, referred to as algorithm `CluFL` in [45]. Algorithm CluFL groups UEs into clusters, and each cluster has a single worker aggregator, using the K-means method, where the cluster head of each cluster has a worker aggregator. There is also a master aggregator that collects and aggregates the model from each cluster.
- The third one, referred to as `GREEDY`, consolidates all worker aggregators into a single location.

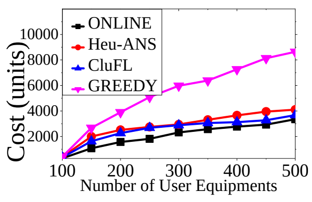
6.2 Simulation Results

We first evaluated algorithm `OptFWK` against its counterparts `Heu-ANS`, `CluFL` and `GREEDY` for the problem of joint worker aggregator placement and UE assignment problem with a single FL request in an MEC network, as shown in Fig. 4.

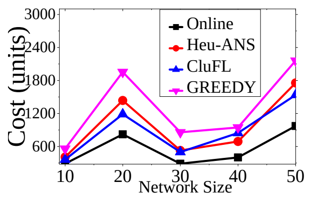
Fig. 4 (a) depicts the cost curves of implementing an FL request, by varying the number of UEs from 100 to 500 while fixing the network size at 20. We can see that algorithm `OptFWK` has the lowest implementation cost among the four comparison algorithms. The reasons are twofold. On one hand, `OptFWK` leverages a binary search process to find the appropriate number of worker aggregators for each FL request. On the other hand, it places the worker aggregators to multiple locations, which results in that the worker aggregators can be distributed to the locations close to UEs with high probability. We can also see that the implementation cost of `GREEDY` increases with the growth of the number of UEs. This is due to the fact that the use of more UEs means aggregating larger volumes of local models from these UEs. Fig. 4 (b) shows the implementation cost of an FL request by varying network size from 10 to 100. It can be seen that the implementation cost fluctuates. Specifically, the cost increases when the network size grows from 10 to 20, because the communication cost grows with the increase on the network size. However, the cost decreases afterwards when the network size varies from 20 to 40. The reason is that more worker aggregators are placed to the network to reduce the cost of transmitting local models to worker aggregators. Then, the cost increases when the network size keeps growing. The rationale behind is that the worker aggregators may be placed sparsely in a large network, thereby pushing up the implementation cost of



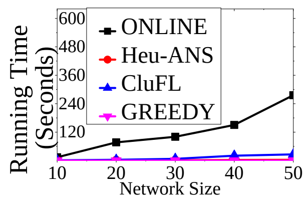
(a) The total cost by varying the number of requests.



(b) The average cost by varying the number of UEs.



(c) The total cost by varying network sizes.



(d) Running time.

Fig. 6. The performance of algorithms `ONLINE`, `Heu-ANS`, `CluFL`, and `GREEDY` for the online worker aggregator placement and UE assignment problem with model uncertainty.

updating local models to the master aggregator. Fig. 4 (c) describes the cost of implementing an FL request by varying $|w|$ from 50 to 500. The costs increase as $|w|$ grows, because a larger model size normally implies a higher communication cost. Fig. 4 (d) shows the running times of the comparison algorithms. We can see that algorithm `OptFWK` has a slightly longer running time than the rest of the algorithms as the solution delivered by it has a better quality.

Fig. 5 shows the impact of the number of UEs and the number of worker aggregators on the solution of algorithm `OptFWK`. Fig. 5 (a) depicts the number of worker aggregators of implementing an FL request, by varying the number of UEs from 100 to 500 while fixing network size at 20. The number of worker aggregators increases with the increase on the number of UEs. The reason is with more UEs, more worker aggregators are to be distributed in the MEC network to reduce the communication cost of local model updating from UEs to their assigned worker aggregators. In addition, there is a significant growth in the number of worker aggregators when the number of UEs increases from 300 to 350. This is because that the computing resources of potential locations are limited, the worker aggregators may be dispersed in more locations, increasing the total number of worker aggregators needed. Fig. 5 (b) depicts the number of worker aggregators of an FL request by varying network size from 10 to 100. It can be seen that the number of worker aggregators fluctuates with the growth of network size. The rationale behind is that to reduce the communication cost, the number of worker aggregators is not needed to change much, as the worker aggregators may be placed into locations that are close to UEs.

We then studied the performance of algorithms `ONLINE` against those of algorithms `Heu-ANS`, `CluFL`, and `GREEDY` for the online worker aggregator placement and UE assignment problem with the uncertainty on the size of trained model. The results are shown in Fig. 6. Note that algorithms `Heu-ANS`, `CluFL`, and `GREEDY` are not inherently online in nature. We thus consider that algorithms `Heu-ANS`, `CluFL`, and `GREEDY` performs 'one-shot' optimization at each time slot.

Fig. 6 (a) illustrates the total cost of implementing all FL requests by varying the number of requests from 5 to 30 while fixing the network size at 20. We can see that the implementation cost of algorithm `Online` is the lowest one among the four comparison algorithms. Besides, the total cost of implementing all FL requests increases when the number of arrived requests keeps increasing, since more requests need more resources to aggregate local models. Fig. 6 (b) shows the total costs of implementing all requests by varying the number of UEs of each FL request from 100 to 500. Results show that the total costs of `Online`, `Heu-ANS` and `CluFL` fluctuate at much lower levels than that of algorithm `GREEDY`. The reason is that `Online`, `Heu-ANS` and `CluFL` place the worker aggregators to multiple locations, while `GREEDY` consolidates all worker aggregators into a single location. Also, the total cost of `GREEDY` increases as the number of UEs grows. The rationale behind is that all the local models of a large number of UEs normally needs to be aggregated by more worker aggregators. Further, since `GREEDY` consolidates all worker aggregators into a single location, the choices of lower-cost locations become less and less with the growth of the number of worker aggregators. Fig. 6 (c) depicts the total costs of the algorithms by varying the network size from 10 to 50. Algorithm `Online` has the lowest implementation cost among the three comparison algorithms. Also, the total costs of the algorithms increase when the network size increases from 10 to 20, decreases when the network size varies from 20 to 40, and increases afterwards. The reason is that as the network size grows, more worker aggregators are needed to cover UEs, thereby increasing the processing cost of worker aggregators. However, as the network size keeps increasing, there is a higher probability of distributing worker aggregators to the locations close to UEs, decreasing the communication costs. From Fig. 6 (d), it can be seen that the running times of all algorithms increase with the network size. Although algorithm `Online` has a higher running time than the other three algorithms, the obtained cost is the lowest.

Fig. 7 demonstrates the impact of the number of requests on a scale-up factor χ of algorithm `ONLINE`. Results show that the value of χ when $\zeta = 100$ is higher than the value when $\zeta = 1000$, when $\mu(|\omega|) \in [50, 200]$. The reason is that a lower number of intervals of the range $[1, |w_{max}|/|w_{min}|]$ means a larger range of each interval. This indicates that χ has a higher probabilities of selecting a larger value. Similar trends can be found for other ranges of $\mu(|\omega|)$.

We finally dealt with the impact of parameter d_{unit} on the performance of algorithm `OptFWK` by varying d_{unit} from 0.005 to 5 and the number of UEs from 100 to 500.

Fig. 8 (a) describes the cost of implementing an FL request. For different numbers of UEs, the minimum cost of algorithm `OptFWK` can be obtained when the value of d_{unit} is in the range of $[0.01, 0.05]$. This is because when the value of d_{unit} is large, the number of flows in the auxiliary graph in `OptFWK` is small, the paths in the auxiliary graph cannot be fully utilized, so the final distribution probability is not optimal.

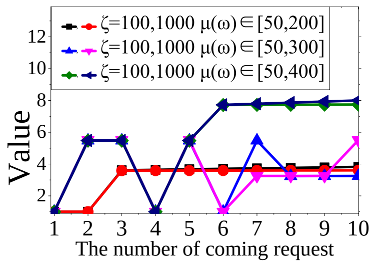


Fig. 7. The χ 's transformation of algorithms `Online`.

Conversely, when the value of d_{unit} is small, the number of flows in the auxiliary graph is large, and each flow has a higher probability of being routed via sub-optimal paths in the auxiliary graph. Similar trends can be found in Fig. 8 (b), which shows the cost of implementing an FL request by varying d_{unit} from 0.005 to 5 while changing the network size from 10 to 50. It can be discovered that with different network sizes, the minimum cost of `OptFWK` can be obtained when the value of d_{unit} is $[0.01, 0.05]$. In addition, Fig. 8 (c) describes the cost of implementing an FL request by varying the d_{unit} from 0.05 to 5 while changing the size of local model from 100 to 500. We can find that with different model sizes, the minimum cost can be obtained when the value of d_{unit} is[0.01, 0.05]. It must be mentioned that the value setting of d_{unit} is also realistic in real scenarios, as it matches the setting in our test-bed in Section 6.3. This is also the reason that we set d_{unit} in the range of $[0.01, 0.05]$ in the rest of our experiments.

6.3 Test-Bed Implementations

We designed a new optimization framework for the hierarchy FL based on the existing FL framework, and evaluated the proposed algorithms for image identification and text recognition applications on real datasets in a real test-bed. We now describe the hardware and software of our test-bed for the hierarchy FL in an MEC network in the following.

Hardware: We built a test-bed for the MEC network with both real hardware devices and virtual network elements as shown in Fig. 9. The hardware includes physical switches, edge devices, edge servers, and virtual switches. There are five Huawei S5720-32C-HI24S-AC switches. Netconf and SNMP protocols are used to manage the physical switches and the links that interconnect them. The edge devices include 3 NVIDIA Jetson AGX Xavier, 3 NVIDIA Jetson TX2, and 3 NVIDIA Jetson Nano. These devices are connected to the MEC network as UEs via WiFi access points. Each edge server is equipped with an Intel(R) Xeon(R) Gold 5118 CPU and two NVIDIA 2080Ti GPUs. The afore-mentioned edge servers and physical switches form a physical underlay, which can be seen as a resource pool with computing and bandwidth resources. Built upon such a underlay, we construct a overlay consisting of a number of virtual switches whose topology corresponds to that of the MEC network.

Software framework: We implemented a hierarchical FL training framework as Fig. 9 shows based on an open source FL framework, i.e., FedML, which is referred to as HierFedML. The framework includes three modules: (1) an **optimizer** that implements the proposed algorithms. It accepts FL requests as inputs to the algorithms, and the algorithms returns configurations that include the number of UEs, the datasets for training, the number of worker aggregators, and the locations of worker aggregators; (2) a **controller** which parses the configurations by the algorithms and transfers the configuration into FL training plans; and (3) a **monitor** monitors a hierarchy FL process based on the interface provided by FedML, by providing necessary states of the current FL processes.

Models and Datasets: Each FL is performed based on datasets MNIST [17], FedEMNIST [33], Shakespeare [7] and HARBox [31] with three different models: 1) Logistic regression on the MNIST dataset; 2) CNN on the FedEMNIST dataset; and 3) RNN on the Shakespeare dataset. By default, we use the FedAvg based on FedEMNIST dataset. The training model is the AlexNet architecture of CNN. Data samples are assigned to each UE uniformly. For all experiments, the batch size, the learning rate, and the model optimizer are set to 10, 0.03 and SGD, respectively. The values of global communication round, the local epoch of each UE and the group communication round are 10, 5 and 10, respectively. The default setting of the numbers of UEs and network size are 200 and 10, respectively.

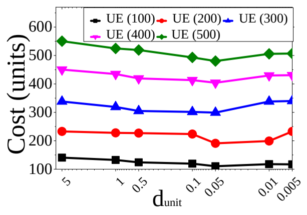
Real and virtual UEs: Note that in our experiments the number of real edge devices is small. To evaluate the performance of the proposed algorithms when the number of UEs is large, we use a container running in an edge server as a virtual UE. Fig. 10, Fig. 11 and Fig. 12 shows the results with virtual UEs. Fig. 13 and Fig. 14 illustrate the results with only real edge devices.

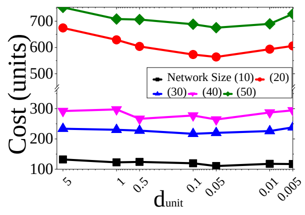
Results: We evaluated the performance of `OptFWK` in the customized FedML framework by varying the number of UEs from 100 to 500. Fig. 10 (a) and (b) show the accuracy and loss of CNN on the FeDEMNIST dataset, from which we can see that the proposed algorithms converge to a high accuracy and low loss within 100 rounds of training for different numbers of UEs. Also, the accuracy increases with the growth on the number of UEs. The reason is that more UEs means the training of more data, and a higher number of local models will be aggregated in each round, leading to a higher accuracy.

We then studied the performance of algorithms `OptFWK`, `Heu-ANS`, `CluFL`, and `GREEDY` in FedFL framework while fixing the number of UEs at 200 and the network size at 30. Fig. 11 (a) and (b) show the accuracy and loss of CNN on the FedEMNIST dataset, from which we can see that `GREEDY` has the fastest convergence followed by `OptFWK`, `Heu-ANS`, and `CluFL`, since `GREEDY` has only one aggregator. However, `CluFL` has the slowest convergence, since it groups UEs into a number of distributed clusters. Also, `OptFWK` can guarantee a relatively faster convergence while obtaining the minimum cost among the four algorithms.

We finally studied the performance of algorithm `OptFWK` to test its applicability to different machine learning algorithms LR, CNN and RNN, using different real datasets. From Fig. 12, we can see that LR has a faster convergence speed due to its simple structure, and RNN has the slowest convergence due to long delays in transmitting large number of parameters.

We evaluated the performance of `OptFWK` by varying the d_{unit} from 0.005 to 5 while changing the model size from 100 to 500 and fixing the number of UEs at 15. Note that to obtain


(a) The cost by varying d_{unit} with different numbers of UEs.


(b) The cost by varying d_{unit} with different network sizes.

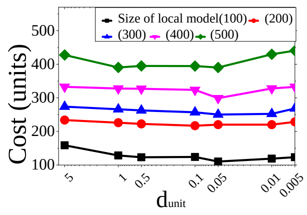

(c) The cost by varying d_{unit} with different sizes of local model.

Fig. 8. The effect of varying d_{unit} on the algorithm OptFWK under different conditions.

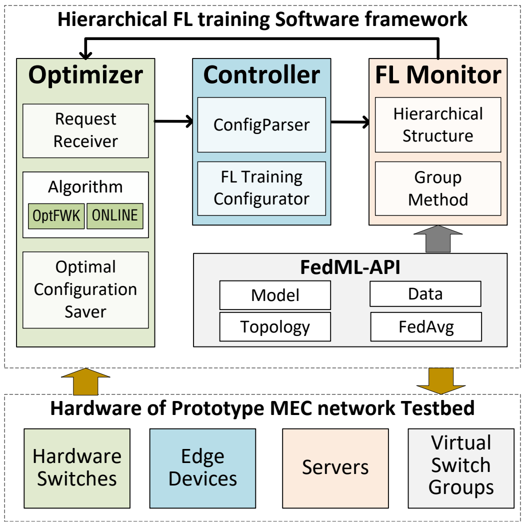


Fig. 9. A hierarchical FL training framework based on an open source FL framework, referred to as HierFedML.

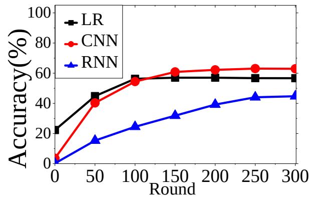


Fig. 12. The performance of algorithms OptFWK in FedML with machine learning algorithms LR, CNN and RNN based on different real datasets.

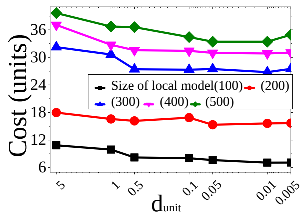
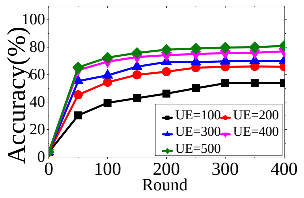


Fig. 13. The performance of algorithms OptFWK by varying d_{unit} with different sizes of local model in FedFL framework with hardware testbed.


(a) The accuracy by varying the number UEs of each FL request.

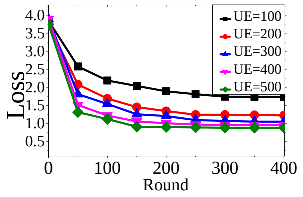
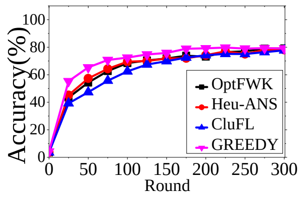

(b) The loss by varying the number UEs of each FL request.

Fig. 10. The performance of algorithms OptFWK by varying the number of UEs in FedFL framework


(a) The accuracy by changing the algorithms of each FL request.

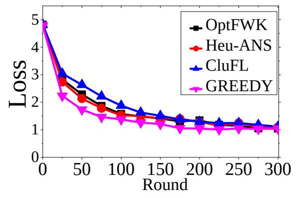

(b) The loss by changing the algorithms of each FL request.

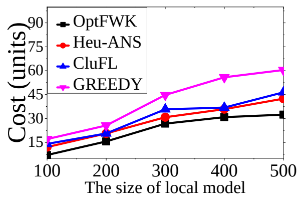
Fig. 11. The performance of algorithms OptFWK, Heu-ANS, CluFL, and GREEDY in FedFL framework.

the right settings for d_{unit} we only use real edge devices. Fig. 13 describes the cost of implementing an FL request. We can find that with different model sizes, the minimum cost can be obtained when the value of d_{unit} is [0.01, 0.05]. Note that this result is consistent with the simulation results in Fig. 8, and this is the reason that we set d_{unit} in the range of $[0.01, 0.05]$ in all other experiments.

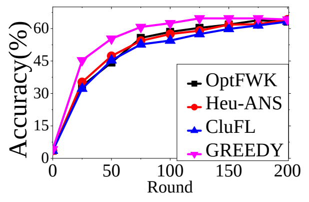
We finally evaluated the performance of algorithms OptFWK, Heu-ANS, CluFL, and GREEDY in FedFL framework with only physical UEs. Fig. 14 (a) describes that, with the increase of the sizes of local models, the costs of the four algorithms increase accordingly. At the same time, the cost of OptFWK is the smallest compared to other algorithms, which echoes the results of our simulation experiments and shows that OptFWK can also guarantee the smallest cost in the actual environment. Fig. 14 (b) shows the accuracy and the convergence speed of the algorithms, by fixing the size of local model as 300 of each FL request. We can see that the convergence speed of GREEDY is the fastest, OptFWK and Heu-ANS are moderate, and CluFL is the slowest. Since there are a limited number of physical edge devices, the performance gap is not large.

7 Conclusions

In this paper, we investigated the problem of joint worker aggregator placement and UE assignment in an MEC network for different FL requests with model uncertainty. We first proposed an optimization framework and an approximation algorithm with an approximation ratio for a single FL request admission. We then studied the problem of online worker aggregator placements and UE assignments when a sequence of FL requests arrives one by one without the knowledge of future arrivals, assuming that the size



(a) The cost varying the size of local model of each FL request.



(b) The accuracy by fixing the size of local model as 300 of each FL request.

Fig. 14. The performance of algorithms `OptFWK`, `Heu-ANS`, `CluFL`, and `GREEDY` in FedFL framework with hardware testbed.

of local model from each UE is uncertain, for which we proposed an online learning algorithm with a bounded accumulative regret, via leveraging the multi-armed bandits technique. We finally evaluated the performance of the proposed algorithms by simulations, and simulation results show that the performance of the proposed algorithms are promising. We also implemented a hierarchical FL training framework HierFedML, and experiments based on real datasets in HierFedML demonstrate that the performance of the proposed algorithms indeed are promising.

Acknowledgments

We appreciate the three anonymous referees and the associate editor for their constructive comments and valuable suggestions, which helped us improve the quality and presentation of the paper greatly. The work by Zichuan Xu and Qiufen Xia is funded by the National Natural Science Foundation of China (NSFC) with grant numbers 62172068, 62172071, and the "Xinghai scholar" program. The work by Weifa Liang was supported by a grant from City University of Hong Kong with project No: 9380137/CS.

References

[1] Amazon Pricing. https://aws.amazon.com/emr/pricing/, accessed in Apr 2021.

[2] A. Acharya, Y. Hou, Y. Mao, *et al.* Workload-aware task placement in edge-assisted human re-identification. *Proc. of SECON*, IEEE, 2019.

[3] M. S. H. Abad, E. Ozfatura, D. Gunduz , *et al.*. Hierarchical federated learning across heterogeneous cellular networks *IEEE International Conference on Acoustics Speech and Signal Processing*, pp. 8866–8870, 2020.

[4] K. Bonawitz, H. Eichner, W. Grieskamp, *et al.*. T. V. Overveldt, D. Petrou, D. Ramage, and J. Roselander. Towards Federated Learning at Scale: System Design. *Proc. of SysML*, 2019.

[5] S. Caldas, Peter Wu, Tian Li *et al.* LEAF: A Benchmark for Federated Settings. [online] Available: https://arxiv.org/abs/1812.01097.

[6] Y. Chen, X. Qin, J. Wang *et al.* FedHealth: A Federated Transfer Learning Framework for Wearable Healthcare *IEEE Intelligent Systems*, Vol. 35, No. 4, pp. 83–93, 2020.

[7] H. Chai, S. Leng, Y. Chen , *et al.*. A Hierarchical Blockchain-Enabled Federated Learning Algorithm for Knowledge Sharing in Internet of Vehicles *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 3975–3986,2021.

[8] E. Diao, J. Ding, and V. Tarokh. Computation and communication efficient federated learning for heterogeneous clients. *Proc. of ICLR*, 2021.

[9] M. Duan, D. Liu, X. Chen, *et al.* Self-Balancing Federated Learning With Global Imbalanced Data in Mobile Systems. *IEEE Transactions on Parallel and Distributed Systems*, Vol. 32, No. 1, pp. 59–71, 2021.

[10] J. Feng, L. Liu, Q. Pei,*et al.* Min-Max Cost Optimization for Efficient Hierarchical Federated Learning in Wireless Edge Networks. *IEEE Transactions on Parallel and Distributed Systems*, doi: 10.1109/TPDS.2021.3131654.

[11] R. Girshick, J. Donahue, T. Darrell, *et al.* Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580–587, 2014.

[12] C. He, S. Li, J. So, *et al.* FedML: A Research Library and Benchmark for Federated Machine Learning. *Proc. of NeurIPS*, MIT Press, 2020.

[13] H. Ko, J. Lee, S. Seo, *et al.* Joint Client Selection and Bandwidth Allocation Algorithm for Federated Learning. *IEEE Transactions on Mobile Computing*, doi: 10.1109/TMC.2021.3136611.

[14] GT-ITM. http://www.cc.gatech.edu/projects/gtitm/.

[15] L. Ibraimi, M. Selimi and F. Freitag BePOCH: Improving Federated Learning Performance in Resource-Constrained Computing Devices *IEEE Global Communications Conference*, IEEE, 2021, doi: 10.1109/GLOBECOM46510.2021.9685095.

[16] A. Imteaj, U. Thakker, S. Wang, *et al.* A Survey on Federated Learning for Resource-Constrained IoT Devices. *IEEE Internet of Things Journal*, Vol. 9, No. 1, pp. 1–24, 2022.

[17] Y. Lecun, L. Bottou, Y. Bengio ,*et al.* Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, Vol. 86, No. 11, pp. 2278-2324, IEEE, 1998.

[18] S. Luo, X. Chen, Q. Wu, *et al.* HFEL: Joint Edge Association and Resource Allocation for Cost-Efficient Hierarchical Federated Edge Learning. *IEEE Transactions on Wireless Communications*, Vol.19, No.10, pp. 6535–6548, 2020.

[19] Z. Li, W. Xie, L. Zhang, *et al.* Toward efficient safety helmet detection based on YoloV5 with hierarchical positive sample selection and box density filtering *IEEE Transactions on Instrumentation and Measurement*, Vol. 71, pp. 1–14, 2022.

[20] B. Li, W. Dong, G. Guan, *et al.* Queec: QoE-aware edge computing for IoT devices under dynamic workloads. *ACM Transactions on Sensor Networks*, Vol. 17, No. 3, Article 27, 2021.

[21] W. Lim, J. Ng, Z. Xiong, *et al.* Dynamic edge association and resource allocation in self-organizing hierarchical federated learning networks. *IEEE Journal on Selected Areas in Communications*, Vol. 39, No. 12, pp. 3640–3653, 2021.

[22] W. Lim, J. Ng, Z. Xiong, *et al.* Decentralized edge intelligence: A dynamic resource allocation framework for hierarchical federated learning. *IEEE Transactions on Parallel and Distributed Systems*, Vol. 33, No. 3, pp. 536–550, 2022.

[23] J. Liu, H. Xu, L. Wang *et al.* Adaptive Asynchronous Federated Learning in Resource-Constrained Edge Computing *IEEE Transactions on Mobile Computing*, doi: 10.1109/TMC.2021.3096846.

[24] S. Liu, J. Yu, X. Deng, *et al.* FedCPF: An Efficient-Communication Federated Learning Approach for Vehicular Edge Computing in 6G Communication Networks. *IEEE Transactions on Intelligent Transportation Systems*, doi: 10.1109/TITS.2021.3099368.

[25] J. Liu, H. Xu, L. Wang *et al.* Adaptive Asynchronous Federated Learning in Resource-Constrained Edge Computing *IEEE Transactions on Mobile Computing*, doi: 10.1109/TMC.2021.3096846.

[26] S. Luo, X. Chen, Q. Wu, *et al.* HFEL: Joint Edge Association and Resource Allocation for Cost-Efficient Hierarchical Federated Edge Learning. *IEEE Transactions on Wireless Communications*, Vol.19, No.10, pp. 6535–6548, 2020.

[27] H. B. McMahan, E. Moore, D. Ramage, *et al.* Communication-Efficient Learning of Deep Networks from Decentralized Data. *Proc. of AISTATS*, 2016.

[28] J. Mills, J. Hu and G. Min. Communication-Efficient Federated Learning for Wireless Edge Intelligence in IoT. *IEEE Internet of Things Journal*, Vol. 7, No. 7, pp. 5986–5994, 2020.

[29] M. Mitzenmacher and E. Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.

[30] M. N. H. Nguyen, N. H. Tran, K. Yan, *et al.* Toward Multiple Federated Learning Services Resource Sharing in Mobile Edge Networks. [online] Available: https://arxiv.org/abs/2011.12469.

[31] X. Ouyang, Z. Xie, J. Zhou, *et al.* Clusterfl: a similarity aware federated learning system for human activity recognition. *Proc. of MobiSys*, ACM, 2021.

[32] Y. Qian, L. Hu, J. Chen, *et al.* Privacy-aware Service Placement for Mobile Edge Computing via Federated Learning. *Information Sciences*, Vol. 505, pp. 562–570, Elsevier, 2019.

[33] S. Reddi, Z. Charles, M. Zaheer,*et al.* Adaptive Federated Optimization. [online] Available: https://arxiv.org/abs/2003.00295.

[34] S. Ren, K. He, R. Girshick , *et al.* Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 39, No. 6, pp. 1137–1149, 2017.

[35] D. Rothchild, A. Panda, E. Ullah, *et al.* FetchSGD: Communication-Efficient Federated Learning with Sketching. *Proc. of ICML*, 2020.
[36] R. Saha, S. Misra, and P. K. Deb. FogFL: Fog Assisted Federated Learning for Resource-constrained IoT Devices. To appear in *IEEE Internet of Things Journal*, 2020, DOI: 10.1109/JIOT.2020.3046509.
[37] L. Shi, J. Shu, W. Zhang *et al.* HFL-DP: Hierarchical federated learning with differential privacy. *IEEE Global Communications Conference*, 2021, DOI: 10.1109/GLOBECOM46510.2021.9685644.
[38] W. Shi, S. Zhou, Z. Niu, *et al.* Joint Device Scheduling and Resource Allocation for Latency Constrained Wireless Federated Learning. To appear in *IEEE Transactions on Wireless Communications*, IEEE, 2020, DOI: 10.1109/TWC.2020.3025446.
[39] A. Slivkins. Introduction to Multi-armed Bandits. 2019. [Online]. Available: http://arxiv.org/abs/1904.07272
[40] N. H. Tran, W. Bao, A. Zomaya, *et al.* Federated Learning over Wireless Networks: Optimization Model Design and Analysis. *Proc. of INFOCOM*, 2019.
[41] W. Wen, Z. Chen, H. H. Yang, *et al.* Joint Scheduling and Resource Allocation for Hierarchical Federated Edge Learning. *IEEE Transactions on Wireless Communications*, 2022, DOI: 10.1109/TWC.2022.3144140.
[42] H. Wang, Z. Kaplan, D. Niu, *et al.* Optimizing Federated Learning on Non-IID Data with Reinforcement Learning. *Proc. of INFOCOM*, 2020.
[43] S. Wang, T. Tuor, T. Salonidis, *et al.* Adaptive Federated Learning in Resource Constrained Edge Computing Systems. *IEEE Journal on Selected Areas in Communications*, Vol. 37, No. 6, pp. 1205–1221, 2019.
[44] X. Wang, C. Wang, X. Li, *et al.* Federated Deep Reinforcement Learning for Internet of Things With Decentralized Cooperative Edge Caching. *IEEE Internet of Things Journal*, Vol. 7, No. 10, pp. 9441–9455, 2020.
[45] Z. Wang, H. Xu, J. Liu, *et al.* Resource-efficient federated learning with hierarchical aggregation in edge computing. *Proc. of INFOCOM*, IEEE, 2021.
[46] W. Wu, L. He, W. Lin, *et al.* SAFA: a Semi-Asynchronous Protocol for Fast Federated Learning with Low Overhead. To appear in *IEEE Transactions on Computers*, 2020, DOI: 10.1109/TC.2020.2994391.
[47] X. Wu, X. Yao and C. L. Wang. FedSCR: Structure-Based Communication Reduction for Federated Learning. *IEEE Transactions on Parallel and Distributed Systems*, Vol. 32, no. 7, pp. 1565–1577, 2021.
[48] W. Xia, T. Q. S. Quek, K. Guo, *et al.* Multi-Armed Bandit-Based Client Scheduling for Federated Learning. *IEEE Transactions on Wireless Communications*, Vol. 19, No.11, pp. 7108–7123, 2020.
[49] Q. Xia, W. Ren, Z. Xu, X. Wang, and W. Liang. When edge caching meets a budget: Near optimal service delivery in multi-tiered edge clouds. *IEEE Transactions on Services Computing*, to appear, 2021.
[50] J. Xu and H. Wang. Client Selection and Bandwidth Allocation in Wireless Federated Learning Networks: A Long-Term Perspective. *IEEE Transactions on Wireless Communications*, Vol. 20, No. 2, pp. 1188–1200, 2021.
[51] Z. Xu, J. Wu, Q. Xia, P. Zhou, J. Ren and H. Liang. Identity-aware attribute recognition via real-time distributed inference in mobile edge clouds. *Proc. of ACM Multimedia*, 2020.
[52] S. Xie, Y. Xue, Y. Zhu and Z. Wang Cost Effective MLaaS Federation: A Combinatorial Reinforcement Learning Approach *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications*, pp. 2078–2087, doi: 10.1109/INFOCOM48880.2022.9796701.
[53] Z. Xu, L. Zhou, S. Chau, W. Liang, Q. Xia and P. Zhou. Collaborate or separate? Distributed service caching in mobile edge clouds. *Proc. of IEEE INFOCOM*, 2020.
[54] Z. Xu, L. Zhao, Weifa Liang, Omer F. Rana, Pan Zhou, Qiufen Xia, Wenzheng Xu, and Guowei Wu. Energy-aware inference offloading for DNN-driven applications in mobile edge clouds. IEEE Transactions on Parallel and Distributed Systems , Vol. 32, No. 4, pp. 799-814, IEEE, 2021.
[55] Z. Yang, M. Chen, W. Saad, *et al.* Energy efficient federated learning over wireless communication Networks. [online] Available: https://arxiv.org/abs/1911.02417.
[56] Z. Yang, M. Chen, W. Saad, *et al.* Energy Efficient Federated Learning Over Wireless Communication Networks. *IEEE Transactions on Wireless Communications*, Vol. 20, No. 3, pp. 1935–1949, 2021.
[57] Z. Yu, J. Hu, G. Min, *et al.* Mobility-Aware Proactive Edge Caching for Connected Vehicles Using Federated Learning. To appear in *IEEE Transactions on Intelligent Transportation Systems*, 2020, DOI:10.1109/TITS.2020.3017474.
[58] K. Yang, T. Jiang, Y. Shi *et al.* Federated Learning via Over-the-Air Computation. *IEEE Transactions on Wireless Communications*, Vol.19, No. 3, pp. 2022–2035, 2019.
[59] Q. Yang, Y. Liu, T. Chen, *et al.* Federated Machine Learning *ACM Transactions on Intelligent Systems and Technology*, Vol. 10, No.2, pp. 1–19, 2019.
[60] R. Zeng, S. Zhang, J. Wang, *et al.* FMore: An Incentive Scheme of Multi-dimensional Auction for Federated Learning in MEC. [online] Available: https://arxiv.org/abs/2002.09699.
[61] J. Zhao, X. Zhu, J. Wang and J. Xiao, Efficient Client Contribution Evaluation for Horizontal Federated Learning *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3060–3064, doi: 10.1109/ICASSP39728.2021.9413377.
[62] Y. Zhao, M. Li, L. Lai, *et al.* Federated Learning with Non-IID Data, 2018 [online] Available: https://arxiv.org/abs/1806.00582.
[63] Z. Zhong, Y.Zhou, D.Wu, *et al.*, P-FedAvg: Parallelizing Federated Learning with Theoretical Guarantees. *IEEE INFOCOM*, pp. 1-10, doi: 10.1109, 2021.
[64] G. Zhu, Y. Wang, and K. Huang. Broadband Analog Aggregation for Low-Latency Federated Edge Learning *IEEE Transactions on Wireless Communications*, Vol. 19, No. 1, pp. 491–506, 2019.

Zichuan Xu (M'17) received his PhD degree from the Australian National University in 2016, ME degree and BSc degree from Dalian University of Technology in China in 2011 and 2008, all in Computer Science. From 2016 to 2017, he was a Research Associate at Department of Electronic and Electrical Engineering, University College London, UK. He is currently a full professor and PhD advisor in School of Software at Dalian University of Technology. He is also a 'Xinghai Scholar' in Dalian University of Technology. His research interests include mobile edge computing, serverless computing, network function virtualization, software-defined networking, algorithmic game theory, and optimization problems.

Dapeng Zhao received the B.S. degree in the School of Software from Dalian University of Technology, China, in 2019. He is currently pursuing his M.S. degree in the School of Software, Dalian University of Technology, Dalian, China. His current research interests include federated Learning, mobile-edge computing, and resource allocation.

Weifa Liang (M'99–SM'01) received the PhD degree from the Australian National University in 1998, the ME degree from the University of Science and Technology of China in 1989, and the BSc degree from Wuhan University, China in 1984, all in Computer Science. He is currently a Professor at the Department of Computer Science, City University of Hong Kong, Hong Kong, prior to that position, he was a Professor at the Australian National University. His research interests include design and analysis of energy efficient routing protocols for wireless ad hoc and sensor networks, the Internet of Things, Mobile Edge Computing (MEC), Network Function Virtualization (NFV), Software-Defined Networking (SDN), design and analysis of parallel and distributed algorithms, approximation algorithms, combinatorial optimization, and graph theory. He currently serves as an Associate Editor for the IEEE Transactions on Communications, and he is a senior member of the IEEE.

Omer F. Rana received the B.S. degree in information systems engineering from the Imperial College of Science, Technology and Medicine, London, U.K., the M.S. degree in microelectronics systems design from the University of Southampton, Southampton, U.K., and the Ph.D. degree in neural computing and parallel architectures from the Imperial College of Science, Technology and Medicine. He is a Professor of performance engineering with Cardiff University, Cardiff, U.K. His current research interests include problem solving environments for computational science and commercial computing, data analysis and management for large-scale computing, and scalability in high performance agent systems.

Pan Zhou (S'07-M'14) received the B.S. degree in the Advanced Class of Huazhong University of Science and Technology (HUST), Wuhan, China, in 2006, and the Ph.D. degree from the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA, in 2011. He is currently an Associate Professor with the School of Electronic Information and Communications, HUST, Wuhan, China. He was a Senior Technical Member with Oracle, Inc., American, from 2011 to 2013, Boston, MA, USA. His current research interests include security and privacy, machine learning and big data analytics, and information networks.

Mingchu Li received the B.S. degree in mathematics from Jiangxi Normal University, Nanchang, China, in 1983, the M.S. degree in applied science from the University of Science and Technology Beijing, Beijing, China, in 1989, and the Ph.D. degree in mathematics from the University of Toronto, Toronto, ON, Canada, in 1998.

He was an Associate Professor with the University of Science and Technology Beijing from 1989 to 1994. From 2002 to 2004, he was a Full Professor with the School of Software, Tianjin University, Tianjin, China. Since 2004, he has been a Full Professor and the Vice-Dean of the School of Software Technology, Dalian University of Technology, Dalian, China. His main research interests include theoretical computer science and information security, trust models and cooperative game theory.

Wenzheng Xu received the BSc, ME and PhD degrees in computer science from Sun Yat-Sen University, Guangzhou, PR China, in 2008, 2010, and 2015, respectively. He currently is a Special Associate Professor at the Sichuan University and was a visitor at the Australian National University. His research interests include wireless ad hoc and sensor networks, mobile computing, approximation algorithms, combinatorial optimization, online social networks, and graph theory. He is a member of the IEEE.

Hao Li is currently an associate researcher in the Institute of Ningxia at China Coal Research Institute. Dr. Li's research interests include Coal mine intellectualization, automation of fully mechanized mining face, Edge computing, AI and Internet of Things.

Qiufen Xia received her PhD degree from the Australian National University in 2017, the ME degree and BSc degree from Dalian University of Technology in China in 2012 and 2009, all in Computer Science. She is currently an Associate Professor at Dalian University of Technology. Her research interests include mobile cloud computing, query evaluation, big data analytics, big data management in distributed clouds, and cloud computing.