Profit Maximization of Delay-Sensitive, Differential Accuracy Inference Services in Mobile Edge Computing

Yuncan Zhang, *Member, IEEE*, Weifa Liang, *Senior Member, IEEE*, Zichuan Xu, *Member, IEEE*, Xiaohua Jia, *Fellow, IEEE*, and Yuanyuan Yang, *Life Fellow, IEEE*

***Abstract*—The integration of Artificial Itelligence (AI) and edge computing has sparked significant interest in edge inference services. In this paper, we consider delay-sensitive, differential accuracy inference services in a Mobile Edge Computing (MEC) network while meeting user stringent delay and accuracy requirements. We formulate two novel profit maximization problems under static and dynamic settings of service request arrivals, with the aim of maximizing the accumulative profit of admitted requests. We assign differential accuracy service requests to the corresponding resolution instances of their requested service models, assuming that each resolution instance can serve up to $L \geq 1$ the same type of service requests. Since the profit maximization problem is NP-hard, we first formulate an Integer Linear Program (ILP) solution if the problem size is small or medium; otherwise, we devise a constant randomized algorithm with high probability. Then, we consider dynamic service request admissions without the knowledge of future request arrivals for a given finite time horizon, for which we develop a simple yet effective prediction mechanism to accurately predict the number of different resolution instances of each model needed, and pre-deploy the predicted number of resolution instances into cloudlets to reduce instantiating delays. We then devise an online algorithm with a provable competitive ratio for the dynamic profit maximization problem by leveraging the primal-dual dynamic updating technique. Finally, we evaluate the performance of the proposed algorithms by simulations. The simulation results demonstrate that the proposed algorithms are promising.**

***Index Terms*—Edge computing, differential-accuracy inferences, approximation algorithms, online algorithms, model resolution instance placements, multi-resolution service models, primal-dual dynamic updating technique, prediction mechanisms.**

Received 29 October 2024; revised 8 January 2025; accepted 5 February 2025. Date of publication 7 February 2025; date of current version 5 June 2025. This work was supported by University Grants Committee in Hong Kong (HK UGC) under CityUHK Grant 7005845, Grant 8730103, Grant 9043510, Grant 9043668, and Grant 9380137. Recommended for acceptance by S. Wang. *(Corresponding author: Weifa Liang.)*

Yuncan Zhang, Weifa Liang, and Xiaohua Jia are with the Department of Computer Science, City University of Hong Kong, Kowloon Tong, Hong Kong (e-mail: yuncan.zhang@cityu.edu.hk; weifa.liang@cityu.edu.hk; csjia@cityu.edu.hk).

Zichuan Xu is with the School of Software, Dalian University of Technology, Dalian 116024, China (e-mail: z.xu@dlut.edu.cn).

Yuanyuan Yang is with the Department of Electrical and Computer Engineering, Stony Brook University, Stony Brook, NY 11794 USA (e-mail: yuanyuan.yang@stonybrook.edu).

Digital Object Identifier 10.1109/TMC.2025.3540017

I. Introduction

DRIVEN by accelerating convergence of AI and the Internet of Things (IoT), we now witness an unprecedented booming of AI-empowered IoT applications and intelligent services that are based on machine learning models such as Deep Neural Networks (DNNs); particularly, the study of service model based inference services has gained tremendous attention from both academia to industrial sectors [18], [40]. Since intelligent service requires large quantities of computing and storage resources for data-intensive computing tasks, the cloud with abundant computing and storage resources is an ideal platform for intelligent service provisioning. However, most inference services for IoT applications are delay-sensitive and need to respond in real time, such as autonomous driving, VR/AR. The remote cloud cannot meet the stringent service delay requirements of users. Fortunately, edge computing pushes computing and storage resources in proximity of users and end devices, which has been envisioned as a promising computing paradigm for delay-sensitive edge inference services [41], [43], [45].

Despite the tremendous benefits of edge networks, it is crucial to recognize that edge nodes are typically equipped with limited resources, which can impair the performance (accuracy) of resource hungry inference service. To meet the heterogeneous user requirements in the resource constrained edge networks, the model resolution adaption is exploited to reduce the resource consumption and service delay on inference services [40]. For example, the state-of-the-art object detection model YOLOv5 [11] offers five different resolutions with different model sizes: nano, small, medium, large, and xlarge. With differential resolutions of each inference model, a mobile user can choose an appropriate model resolution for his requested model service to meet his stringent service delay and service accuracy requirements. Also, several off-the-shelf DNN inference frameworks of choosing different model resolutions have been developed including TensorRT [25], PyTorch [27], and Clipper [5]. Furthermore, there are also several studies that investigated choosing differential accuracy DNN models for inference services in edge computing [8], [32], [33], [40].

Although inference service provisioning has attracted lots of interest of network service providers, how to maximize the provider benefits by properly deploying differential resolution service model instances in edge networks to admit as many

requests as possible while meeting their delay and accuracy requirements has not been well addressed. Most existing studies on differential accuracy service provisioning are based on a single edge server or remote clouds [21], [38], [40], not an edge computing network with multiple edge servers. It must be mentioned that there are two recent studies that focused on collaborative DNN model selections in MEC networks to explore non-trivial trades-off between the inference accuracy and the service delay (or the overhead) [8], [32], by developing reinforcement learning based algorithms for their problems.

In this paper, we consider differential accuracy inference services from a set of DNN models with the aim to maximize the profit of service providers, where there are multiple resolutions for each model and each resolution has a different accuracy level and inference time tailored to meeting different user needs. To provide delay-sensitive differential accuracy inference services in an MEC network for mobile users while meeting their stringent delay and accuracy requirements, it poses the following challenges. First, since the computing capacity on each cloudlet is limited, how to choose an appropriate resolution of the requested service model and where to place the model resolution instance to meet the accuracy and delay requirements of users is challenging. Second, each user usually moves around the network over time, and the services requested by the user at different time slots are very likely to be different. How many resolution instances of each requested service model need to be placed to meet the delay and accuracy requirements of the user is another challenge. Third, to fully utilize limited resources in the MEC network, we assume that each resolution instance of a service model can serve up to L that type of requests. How to group the same type of service requests with differential accuracy requirements into a resolution instance of their requested model is a challenge. Finally, there is no knowledge of future request arrivals for a given monitoring period, how many resolution instances of each service model need to be placed in advance to reduce instantiating delays of different model resolution instance deployments, thereby admitting more service requests, is critical. In this paper, we will address the aforementioned challenges.

The novelty of this paper lies in the study of delay-sensitive, differential accuracy inference services in an MEC network, by jointly choosing different resolutions of inference models, instance placements of chosen model resolutions, and request assignments to different resolution instances of their requested models. Two novel profit maximization problems are formulated under static and dynamic request arrival settings, and performance-guaranteed randomized and online algorithms are devised too.

The main contributions of this paper are summarized as follows. We study delay-sensitive, differential accuracy inference services in an MEC network. We first formulate two novel profit maximization problems under static and dynamic settings of user request arrivals. We then propose an Integer Linear Program (ILP) solution to the profit maximization problem when the problem size is small or medium; otherwise, we develop a constant randomized approximation algorithm with high probability. We also deal with the dynamic profit maximization problem within a finite time horizon, where user requests arrive one by one without the knowledge of future arrivals, for which we develop a simple yet efficient prediction mechanism to predict the number of different resolution instances of each model in each cloudlet needed and then place them in advance. We then devise an online algorithm with a provable competitive ratio for the problem, by adopting the primal-dual dynamic updating technique. We finally evaluate the performance of the proposed algorithms. Simulation results demonstrate that the proposed algorithms are promising.

The rest of the paper is organized as follows. Section II surveys the related work. Section III introduces the system model, notions and notations, and problem definitions. Section IV develops ILP and randomized algorithm for the profit maximization problem. Section V deals with the dynamic profit maximization problem. Section VI evaluates the proposed algorithms, and Section VII concludes the paper.

II. Related Work

Lots of efforts on service provisioning in MEC networks have been taken in the past decades [15], [17], [23], [35]. For example, Ma et al. [23] considered SFC-enabled request admissions in an MEC network with the aim to maximize the total profit, and developed performance-guaranteed algorithms. Li et al. [17] dealt with the profit maximization problem of dynamic SFC-enabled request admissions, and devised a DRL algorithm for the online service instance placement and request assignment problem. Li et al. [15] modeled an inference service by a DNN model and developed an efficient algorithm for task offloading, with the aim to minimize the end-to-end delay of the model training, where the DNN is a directed acyclic graph. Xu et al. [35] considered energy-aware inference-driven DNN offloading by proposing a task offloading algorithm. However, none of these studies considered service provisioning of differential accuracy.

With the advance of AI and machine learning, intelligent service provisioning in MEC networks now becomes a main research topic [4], [13], [14], [39], [44]. For instance, Li et al. [14] devised approximation and online algorithms for dynamic Digital Twin (DT) placements with the assumption of object mobility to provide low age of information services on DT data. Yang et al. [39] studied digital-twin assisted task executions. They presented a two-timescale accuracy-aware online optimization approach to maximize the accuracy of each task execution while meeting its stringent energy and delay constraints. Qiu et al. [28] investigated AI service chain provisioning with the aim to maximize the throughput of admitted requests while minimizing the service deployment cost, by jointly considering reliability, security, and accuracy requirements. Although these works took into account the service accuracy requirement, none of them considered multi-resolution inference service provisions. Due to the heterogeneity of mobile devices and different quality-of-service requirements of users, providing delay-sensitive, differential accuracy inference services in MEC networks is very critical. To strive for the finest trade-off between service accuracy and inference time on each inference model, there are three different approaches, which are given as

follows. (1) The model compression approach that compresses an inference model (e.g., a DNN) in both the number of neurons and the number of layers to mitigate the computing and memory resource demands of the model, with compromises on the model accuracy and running time [7], [38]. (2) The multiple early exit method that exits from one exit point of multiple exit points in an inference model to ensure that the inference accuracy and inference time requirements of users are met [10], [12], [21]. And (3) the multi-resolution method for each inference service model that provides a different accuracy and inference time for each different resolution of the model. For example, Xie et al. [33] dealt with the selection of an inference model with multiple resolutions in a single base station attached to an edge server. They focused on fairness of inference accuracy of different models and operational costs by developing an approximate solution for the problem, through adopting the cooperative game-theoretic approach. Zhao et al. [40] developed an online optimization framework `EdgeAdaptor` for DNN inference services in a single edge server or in a remote cloud. They formulated a cost minimization problem that jointly optimizes the application configuration adaption, DNN model selection, and edge resource provisioning.

Different from [33] and [40] that studied inference service provisioning on a single edge server (cloudlet), there are several works dealing with inference services in an MEC network with multiple cloudlets and base stations. For example, Gao et al. [8] considered deploying DNN models for video analytics at edge nodes to jointly optimize the recognition accuracy and overall delay. They introduced a multi-agent reinforcement learning based framework to dynamically determine DNN model selection, video frame configurations, and the optimal edge node for inference. Xu et al. [32] developed a collaborative DNN model selection scheme for heterogeneous edge computing systems, with the aim to increase the inference accuracy while reducing the overhead of edge systems. The works in [8] and [32] both made use of the reinforcement learning based algorithms for the problems, which require significant amounts of training data and are susceptible to variances during learning process.

Unlike the aforementioned works, in this paper, we will study the profit maximization problem of delay-sensitive, differential accuracy inference services in an MEC network, where each inference model can have up to K different resolutions with each having a different accuracy and running time. We aim to maximize the total profit of the MEC network service provider through admitting user requests, by jointly considering different resolution instance placements of each service model, assignments of same type of service requests (up to L requests) to one of their model instances while meeting their delay and accuracy requirements. We will devise performance-guaranteed approximation and online algorithms for the profit maximization problems under both static and dynamic settings of request arrivals.

III. Preliminaries

In this section, we first introduce the system model, notions and notations. We then define the problems precisely.

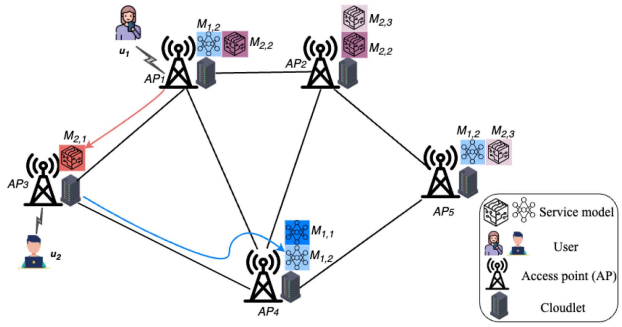


Fig. 1. An illustrative example of an MEC network with two service models M_1 and M_2. M_1 has a high resolution model $M_{1,1}$ and a low resolution model $M_{1,2}$, while M_2 has three different resolution models: a high resolution model $M_{2,1}$, a middle resolution model $M_{2,2}$, and a low resolution model $M_{2,3}$. There are two users requesting for inference services. User u_1 requests service model M_2 with a high accuracy requirement, and the request is assigned to an instance of $M_{2,1}$ hosted by cloudlet 3. User u_2 requests service model M_1 with a low accuracy requirement, and its request is assigned to an instance of $M_{1,1}$ or $M_{1,2}$ that is located at cloudlet 4.

A. System Model

We consider an MEC network $G = (N, E)$ that consists of a set N of Access Points (APs) and a set E of links between APs. Each AP is co-located with a cloudlet and the communication delay between the AP and its co-located cloudlet is negligible. For the sake of convenience, in the rest of this paper, we use an AP and its co-located cloudlet interchangeably if no confusion arises. Denote by C_j the computing capacity of cloudlet $j \in N$. Denote by d_e and ζ_e the transmission delay and cost of a unit data on link $e \in E$, respectively.

There is a set $\mathcal{M}$ of inference models deployed in the MEC network for inference services. Each model in $\mathcal{M}$ has K different resolutions $M_{m,1}, M_{m,2}, \ldots, M_{m,K}$, and each resolution $M_{m,k}$ has accuracy $\epsilon_{m,k}$ and inference time $\tau_{m,k}$, respectively. The amount of computing resource demanded by an instance of inference service $M_{m,k}$ is $s_{m,k}$. The accuracies of different resolutions $M_{m,k}$ of the model M_m satisfy $\epsilon_{m,1} > \epsilon_{m,2} > \ldots > \epsilon_{m,K}$, their inference times meet $\tau_{m,1} > \tau_{m,2} > \ldots > \tau_{m,K}$, and the amounts of their demanded computing resources are proportional to their accuracy levels, that is, $s_{m,1} > s_{m,2} > \ldots > s_{m,K}$.

Given a set U of user requests, each user request $i \in U$ is represented by a tuple $\langle l_i, vol_i, m_i, a_i, D_i \rangle$, where AP_{l_i} is the location of the user, vol_i is the volume of input data of the user i to upload, a_i is the minimum accuracy requirement in its requested service model $M_{m_i} \in \mathcal{M}$, and D_i is the service delay requirement of request i with $l_i \in [1, |N|]$. Each resolution instance of a service model is hosted by a Virtual Machine (VM) or a container in a cloudlet, each instance can serve up to L same type of service requests [33], and the value of L typically is a fixed small integer, e.g., $L = 4$.

Fig. 1 is an illustrative example of the system model with two service models M_1 and M_2, where M_1 has two resolutions and M_2 has three different resolutions.

B. Service Delays of User Requests

To admit a user request, there must exist an instance of its requested service model to process the request while meeting

the delay and accuracy requirements of the request, where the instance can be pre-installed or installed on demand. If an instance of a service model has already been installed, a user requesting for the model can be assigned to the instance for processing immediately; otherwise, an instance of the requested model needs to be launched, which is implemented through creating a new container/VM to host the instance. However, instantiating an instance incurs the instantiating delay, which is the duration of configuring the container and the service model environment. When admitting dynamic user requests, the service provider may remove some existing (idle) model instances to leave room for instantiations of new model resolution instances.

The end-to-end service delay of a user request includes the uploading delay of the input data, the data transmission delay from the user location to the cloudlet hosting its requested service model instance, and the processing delay of the request on its requested service model instance. We assume that the Orthogonal Frequency-Division Multiple Access scheme at each AP is adopted. The bandwidth allocated to each user request under an AP is equal. Specifically, let B_i be the uploading rate of request i to its gateway AP l_i with $B_i = W_{i,l_i} \cdot \log(1 + \frac{P_i \cdot H_{i,l_i}}{\sigma^2 + I_{l_i}})$, where W_{i,l_i} is the amount of bandwidth of AP l_i allocated to user request i, P_i is the transmission power of object v_i, H_{i,l_i} is the channel gain between the locations of request i and AP l_i, and $\sigma^2 + I_{l_i}$ is the average power of the noise and interference over the bandwidth [35].

The data uploading delay of user i through AP l_i then is vol_i/B_i. Assume that request i is assigned to a resolution instance $M_{m_i,k}$ of model M_{m_i} in cloudlet j with accuracy $\epsilon_{i,k} \geq a_i$. Denote by $path_{l_i,j}$ a shortest path in G between user location AP_{l_i} and cloudlet j. The transmission delay per unit data along a routing path $path_{l_i,j}$ is $d(path_{l_i,j}) = \sum_{e \in path_{l_i,j}} d_e$. The end-to-end service delay $d_{i,j,m_i,k}$ of request i requesting for a resolution instance $M_{m_i,k}$ of model M_{m_i} in cloudlet j is

$$d_{i,j,m_i,k} = vol_i/B_i + vol_i \cdot d(path_{l_i,j}) + \tau_{m_i,k} \quad (1)$$

To meet the delay requirement of request i, $d_{i,j,m_i,k} \leq D_i$ if it is processed by an existing instance; otherwise (if it is assigned to a newly instantiating instance), $d_{i,j,m_i,k} + init_{m_i,k} \leq D_i$, where $init_{m_i,k}$ is the instantiating delay of model resolution $M_{m_i,k}$.

C. Profit of Request Admissions

The network service provider typically charges users by admitting their requests in a pay-as-you-go revenue collection scheme [23]. The payment of admitting request i is closely related to (i) its service accuracy and end-to-end service delay requirements, and (ii) the total cost of various resources consumed for the admission of request i.

The monetary payment of a request admission: A request cannot be admitted if its accuracy or delay requirement is violated. We define the payment of request i as a stair-step function $\psi(\cdot,\cdot,\cdot)$, which is defined as follows.

$$pay_i = \psi(m_i, a_i, D_i) \quad (2)$$

where m_i is the index of its requested service model M_{m_i}, a_i is its accuracy requirement, D_i is its service delay requirement. In general, given two requests requesting for the same service model, the request with a higher accuracy and lower delay requirements will pay more than that of another request with a lower accuracy and higher delay requirements. The service provider can adjust the payment function by the market needs. For example, similar to that the accuracy of a service model can be divided into differential accuracy levels, the end-to-end delay of a request can also be divided into Q different delay levels $q_1, q_2, \ldots, q_Q$ with $q_i < q_j$ if $i < j$. The payment pay_i of user i requesting for model M_{m_i} with the actual accuracy no less than $a_i \in (\epsilon_{m_i,p}, \epsilon_{m_i,p-1}]$ with a minimum p on its model resolution $M_{m_i,k}$ and the actual end-to-end service delay $d_{i,j,k}$ no greater than D_i with $D_i \in (q_r, q_{r+1}]$ (where r is the minimum value) is determined by function $\psi(m_i, a_i, D_i)$. *The cost of processing a user request:* There are costs of various resources consumed for inference service provisioning. Denote by $\varphi_{j,m_i,k}$ the inference cost of processing request i on a model resolution instance $M_{m_i,k}$ located at cloudlet j. Then, the total cost of admitting request i by an instance of model resolution $M_{m_i,k}$ in cloudlet j is

$$cost_{i,j,m_i,k} = \xi \cdot \frac{vol_i}{B_i} + vol_i \cdot \sum_{e \in path_{l_i,j}} \zeta_e + \varphi_{j,m_i,k}, \quad (3)$$

where the first term in the RHS of (3) is the cost of uploading the input data of request i, the second term is the transfer cost of transmitting the input data from the user location to the cloudlet hosting the model instance, and the third term is the processing cost of admitting request i against on the model instance. ξ is the data uploading cost per unit time.

The profit of admitting a request: The profit of admitting request i is the difference of its user payment to the total cost of various resources consumed of its admission if it is processed on a resolution instance $M_{m_i,k}$ of model M_{m_i} in cloudlet j, which is defined as follows.

$$profit_{i,j,m_i,k} = pay_i - cost_{i,j,m_i,k}. \quad (4)$$

For the sake of convenience, the symbols used in this paper are summarized in Table I.

D. Problem Definition

Definition 1: Given an MEC network $G = (N, E)$ with computing capacity C_j for each cloudlet $j \in N$, a set $\mathcal{M}$ of service models with K different resolutions of each model, a set U of user service requests with each request $i \in U$ represented by a tuple $\langle l_i, vol_i, m_i, a_i, D_i \rangle$, the profit maximization problem is to admit as many user service requests in U as possible such that the total profit obtained by admitting user service requests in U is maximized, subject to computing capacity on each cloudlet.

Definition 2: Given an MEC network $G = (N, E)$ with computing capacity C_j for each cloudlet $j \in N$, a set $\mathcal{M}$ of service models with K different resolutions of each model, a given finite time horizon $\mathbb{T}$, assume that user service requests arrive one by one without the knowledge of future arrivals within time horizon $\mathbb{T}$, and each arrived request i, represented by a

TABLE I
TABLE OF NOTATIONS

Notation	Descriptions
$G = (N, E)$	An MEC network with a set N of APs (cloudlets) and a set E of links
C_j	The computing capacity on cloudlet j
d_e, ζ_e	The transmission delay and cost of a unit data on link $e \in E$
$\mathcal{M}$	A set of inference service models
K	The number of different resolutions of each model $M_m \in \mathcal{M}$
$M_{m,k}$	The kth resolution of model M_m.
$\epsilon_{m,k}$, $\tau_{m,k}$, $\varphi_{j,m,k}$	The accuracy, inference time, and inference cost of resolution $M_{m,k}$
$s_{m,k}$, $init_{m,k}$	The amounts of computing resource and time of instantiating an instance of resolution $M_{m,k}$
U, U^t	the sets of all user requests and user requests at time slot t
L	The number of service requests that can be served at the same time by an instance of inference model
B_i	The uploading rate of user request i to its gateway AP
l_i	The index of gateway AP of user request i, i.e., i is connected to AP_{l_i}
vol_i	The volume of data of user request $i \in U$
M_{m_i}	The service model requested by user request i
D_i, a_i	The service delay requirement and minimum accuracy requirement on the requested service model of user request i
pay_i	The payment of user request i
$cost_{i,j,m,k}$	The cost of request i processed by an instance of model resolution $M_{m,k}$ at cloudlet j
$profit_{i,j,m,k}$	The profit of admitting request i that is processed by an instance of model resolution $M_{m,k}$ at cloudlet j
$\mathbb{T}$	A finite time horizon
$x_{i,j,m,k}$	A binary variable indicating whether request i is assigned to model $M_{m,k}$ at cloudlet j or not
$\tilde{x}_{i,j,m,k}$, $\widetilde{OPT}$	The real variable and optimal solution of the LP relaxation.
Γ	The parameter used for analyzing the performance of the randomized algorithm
θ	An integer threshold that determines whether an instance of a model resolution can be removed
$n^{(t)}_{j,m,k}$	The number of instances of existing model resolution $M_{m,k}$ in cloudlet j at time slot $t \in \mathbb{T}$
$x^t_{i,j,m,k}$, $y^t_{i,j,m,k}$	A binary variable indicating whether request i is assigned to a newly installed or an existing model resolution instance $M_{m,k}$ in cloudlet j or not at time slot t
$\mathcal{N}^t(i)$, $\mathcal{E}^t(i)$	the sets of pairs of (j,k) that can fulfill the delay and accuracy requirements of request i by a newly instantiated instance and an existing instance of model resolution $M_{m_i,k}$ in cloudlet j
$\alpha^t_j, \beta^t_j, \gamma^t_j, \eta^t_{j,m,k}$	Dual variables of constraints (21), (24), and (27)

tuple $\langle l_i, vol_i, m_i, a_i, D_i \rangle$, must be admitted or rejected immediately, the dynamic profit maximization problem is to admit as many user requests as possible while meeting their service accuracy and delay requirements, such that the accumulative profit obtained by the admitted requests within time horizon $\mathbb{T}$ is maximized, subject to computing capacity on each cloudlet.

It can be seen that the two defined problems are NP-hard by reducing from the well-known NP-complete Knapsack problem.

IV. Algorithms for the Profit Maximization Problem

In this section, we investigate the profit maximization problem. We formulate an ILP solution to the problem when the problem size is small or medium, otherwise we develop a constant randomized approximation algorithm with high probability for it.

A. ILP Formulation

Let $x_{i,j,m,k}$ be a binary variable that indicates whether request i is assigned to model $M_{m,k}$ at cloudlet j ($x_{i,j,m,k} = 1$) or not ($x_{i,j,m,k} = 0$). The problem objective is to maximize the total profit of admitted requests, and an ILP solution for it is given as follows.

$$\text{Maximize} \quad \sum_{i \in U} \sum_{j \in N} \sum_{k \in [K]} profit_{i,j,m_i,k} \cdot x_{i,j,m_i,k} \tag{5}$$

s.t.

$$\sum_{m \in M} \sum_{k \in [K]} \left(\frac{\sum_{i \in U} x_{i,j,m,k}}{L} \cdot s_{m,k} \right) \leq C_j, \forall j \in N, \tag{6}$$

$$d_{i,j,m_i,k} \cdot x_{i,j,m_i,k} \leq D_i, \forall i \in U, \forall j \in N, \forall k \in [K], \tag{7}$$

$$x_{i,j,m_i,k} \cdot a_i \leq \epsilon_{m_i,k}, \ \forall i \in U, \forall j \in N, \forall k \in [K] \tag{8}$$

$$\sum_{j \in N} \sum_{k \in [K]} x_{i,j,m_i,k} = q_i, \ \forall i \in U, \tag{9}$$

$$q_i \in \{0,1\}, \ \forall i \in U \tag{10}$$

$$x_{i,j,m,k} = 0, \ \text{if } m \neq m_i, \forall i \in U, \forall j \in N, \forall k \in [K] \tag{11}$$

$$x_{i,j,m,k} \in \{0,1\}, \quad \forall i \in U, \ \forall j \in N, \forall k \in [K] \tag{12}$$

where q_i is an auxiliary binary variable, which ensures that request i is assigned to at most one instance of model M_{m_i}. Constraint (6) ensures that the accumulative amount of computing resource demanded by all different model resolution instances in cloudlet j is no greater than its computing capacity. Constraint (7) ensures that the end-to-end delay of a user request is no greater than its delay requirement. The constraint–> Constraint (8) ensures that the accuracy of a chosen model resolution instance to process the request i is no less than the accuracy requirement a_i of request i. Constraint (9) ensures that each request i is served by one resolution instance of its requested model. The problem is still NP-complete when $L = 1$, and becomes a classic knapsack problem. It can be seen that Constraint (7) and (8) are used to determine whether the kth resolution of model M_{m_i} in cloudlet j can satisfy the delay and

accuracy requirements of request i, respectively. Denote by $\mathcal{E}(i)$ the set of pairs of cloudlet j and resolution index k of model M_{m_i} that can fulfill the delay and accuracy requirements of request i, by an instance of model resolution $M_{m_i,k}$ in cloudlet j, i.e.,

$$\mathcal{E}(i) = \{(j,k) \mid vol_i/B_i + vol_i \cdot d(path_{l_i,j}) + \tau_{m_i,k} \leq D_i,\ \epsilon_{m_i,k} \geq a_i\}, \quad (13)$$

where $\epsilon_{m_i,k} \geq a_i$ in (13) indicates that there is at least one resolution instance of model $M_{m_i,k}$ in cloudlet j for request i. The ILP formula (5) then can be equivalently written as follows.

$$\text{Maximize} \sum_{i \in U} \sum_{(j,k) \in \mathcal{E}(i)} profit_{i,j,m_i,k} \cdot x_{i,j,m_i,k}$$
$$\text{s.t.} \quad (6),\ (9),\ (10),\ (11),\ (12),\ (13) \quad (14)$$

B. Randomized Algorithm

The ILP solution is deliverable when the problem size is small and medium; otherwise, finding the solution is prohibitively large. In the following we develop a scalable, randomized algorithm for the profit maximization problem.

Let $\{\tilde{q}_i, \tilde{x}_{i,j,m_i,k}\}$ be an optimal solution of the Linear Programming (LP) relaxation of the ILP in (14) and $\widetilde{OPT}$ the value of the optimal solution of the LP. We make use of the LP solution to derive an integer solution for the ILP by randomized rounding [22].

Specifically, Let g in `Algorithm 1` be a random number drawn uniformly in interval $[0, 1]$. Let $\{\hat{q}_i, \hat{x}_{i,j,m_i,k}\}$ be a solution of the ILP with high probability, where each random variable $\hat{q}_i$ is set to 1 with probability $\tilde{q}_i$. If variable $\hat{q}_i$ is set to 1, then random variable $\hat{x}_{i,j,m_i,k}$ is set to 1 with probability $\frac{\tilde{x}_{i,j,m_i,k}}{\tilde{q}_i}$ for each pair of $j \in N$ and $k \in [K]$, and this setting is taken by Constraint (9) in an exclusive manner. That is, given each $i \in U$, exactly only one random variable $\hat{x}_{i,j,m_i,k}$ is set to 1 and the rest are set to 0s for each pair of j and k with $\forall j \in N$ and $\forall k \in [K]$. The detailed algorithm for the profit maximization problem is given in `Algorithm 1`.

Algorithm 1: A Randomized Algorithm for the Profit Maximization Problem.

Input: An MEC network $G = (N, E)$ and a set U of requests, and a set $\mathcal{M}$ of inference models with each having K different resolutions.
Output: a scheduling of model resolution instance placements in cloudlets to maximize the total profit of admitted requests in U.

1: Calculate set $\mathcal{E}(i)$ for each request $i \in U$;
2: Solve the linear relaxation LP of the ILP (14);
3: Let $\{\tilde{q}_i, \tilde{x}_{i,j,m_i,k} \mid i \in U, j \in N, k \in [K], m_i \in \mathcal{M}\}$ be the optimal LP solution;
4: $S \leftarrow \emptyset$ /* the solution S */;
5: **for** each $i \in U$ **do**
6: $g \leftarrow$ a randomly generated number $\in [0, 1]$;
7: **if** $\tilde{q}_i \geq g$ **then**
8: $\hat{q}_i \leftarrow 1$ /* request i is admitted */;
9: $h \leftarrow$ a randomly generated number $\in [0, \tilde{q}_i]$;
10: **for all** $j \in N$ **do**
11: **for all** $k \in [K]$ **do**
12: **if** $\sum_{j'=1}^{j} \sum_{k'=1}^{k} \tilde{x}_{i,j',m_i,k'} \geq h$ **then**
13: $\hat{x}_{i,j,m_i,k} \leftarrow 1$ /* request i is assigned to an instance of its requested model $M_{m_i,k}$ in cloudlet j exclusively */;
14: $i \leftarrow i + 1$; Jump to the for loop at line 5;
15: **end if**
16: **end for**
17: **end for**
18: **else**
19: $\hat{q}_i \leftarrow 0$; $\hat{x}_{i,j,m_i,k} \leftarrow 0$
20: **end if** ;
21: $S \leftarrow S \cup \{\langle i, \hat{q}_i, \hat{x}_{i,j,m_i,k} \rangle\}$
22: **end for**
23: **return** S.

C. Algorithm Analysis

In the following, we analyze the approximation ratio and resource capacity violations of the solution delivered by `Algorithm 1`.

Lemma 1: Let $\{\tilde{q}_i, \tilde{x}_{i,j,m_i,k}\}$ be an exact solution of the LP of all $i \in U$. For any $i \in U$, if variable $\hat{q}_i$ is set to 1 with probability $\tilde{q}_i$, then $\hat{x}_{i,j,m_i,k}$ is set to 1 with probability $\frac{\tilde{x}_{i,j,m_i,k}}{\tilde{q}_i}$ exclusively by `Algorithm 1` for each $j \in N$ and $k \in [K]$.

Proof: Following `Algorithm 1`, if $\hat{q}_i = 1$, then request $i \in U$ is admitted, and the probability of $\hat{q}_i = 1$ is $\tilde{q}_i = \sum_{j \in N} \sum_{k \in [K]} \tilde{x}_{i,j,m_i,k}$. Step 12 of `Algorithm 1` ensures that an admitted request is assigned to an instance of its requested model in a cloudlet. That is, there must exist a pair j and k such that condition $\sum_{j'=1}^{j} \sum_{k'=1}^{k} \tilde{x}_{i,j',m_i,k'} \geq h$ in Step 12 holds, as the range of random variable h is in $[0, \tilde{q}_i]$, and random variable $\tilde{x}_{i,j,m_i,k}$ of each valid pair of j and k (i.e., $\tilde{x}_{i,j,m_i,k} \neq 0$) is independent of each other. The probability that $\hat{x}_{i,j,m_i,k} = 1$ then is $\frac{\tilde{x}_{i,j,m_i,k}}{\tilde{q}_i}$. □

Let Γ be a positive value, which is defined as follows.

$$\Gamma = \max\{\max\{profit_{i,j,m,k} \mid i \in U, j \in N, m \in \mathcal{M}, k \in [K]\}, \max\{s_{m,k} \mid m \in \mathcal{M},\ k \in [K]\}\}. \quad (15)$$

Theorem 1: Given an MEC network $G = (N, E)$, a set U of user requests, and a set $\mathcal{M}$ of inference service models with each having K different resolutions, a resolution instance of each model has a different accuracy and inference time that can admit up to L same types of service requests. There is a randomized algorithm, `Algorithm 1`, with approximation ratio $1 - \alpha$ for the profit maximization problem with high probability of $1 - \frac{1}{|U|}$, at the expense of the amount of computing resource consumed by any cloudlet no more than $2 + \kappa$ times its capacity with probability of $1 - \frac{1}{|N|}$. The time complexity of `Algorithm 1` is $O((|U| \cdot |N| \cdot |\mathcal{M}| \cdot K)^{2+\frac{1}{6}} \cdot \log(|U| \cdot |N| \cdot |\mathcal{M}| \cdot K))$, provided that $\min\{C_j \mid j \in N\} \geq \frac{6\Gamma \ln |N|}{L}$,

where Γ is a constant given by (15), $\alpha = \sqrt{\frac{2\Gamma \ln |U|}{\widetilde{OPT}}} < 1$, and $\kappa = \max\left\{\frac{\sum_{m\in\mathcal{M}}\sum_{k\in[K]} s_{m,k}}{C_j} \mid j \in N\right\}$.

Proof: Let OPT and $\widetilde{OPT}$ be the optimal solutions of the ILP formulation (14) and its LP, respectively. We have $\widetilde{OPT} \geq OPT$, since this is the maximization problem. Following `Algorithm 1`, the probabilities of variables $\hat{x}_{i,j,m,k}$ and $\hat{q}_i$ being set to 1s are $\tilde{x}_{i,j,m,k}$ and $\tilde{q}_i$ respectively, as $\mathbf{Pr}[\hat{x}_{i,j,m,k}=1] = \mathbf{Pr}[\hat{x}_{i,j,m,k}=1 \mid \hat{q}_i=1]\mathbf{Pr}[\hat{q}_i=1] = \frac{\tilde{x}_{i,j,m_i,k}}{\tilde{q}_i}\cdot\tilde{q}_i = \tilde{x}_{i,j,m,k}$ and $\mathbf{Pr}[\hat{q}_i=1]=\tilde{q}_i$.

We first analyze the approximation ratio of the randomized algorithm. Let $z_{i,j,m,k} = \frac{profit_{i,j,m,k}}{\Gamma}\cdot\hat{x}_{i,j,m,k}$ be a random variable deriving from random variable $\hat{x}_{i,j,m,k}$. We observe that $z_{i,j,m,k}$ is $\frac{profit_{i,j,m,k}}{\Gamma}$ with probability $\tilde{x}_{i,j,m,k}$; otherwise, $z_{i,j,m,k}$ is 0. Thus, $0 \leq z_{i,j,m,k} \leq 1$, because $\frac{profit_{i,j,m,k}}{\Gamma}\cdot\hat{x}_{i,j,m,k} \leq \frac{profit_{i,j,m,k}}{\max\{profit_{i,j,m,k} \mid i\in U,\ j\in N,\ m\in\mathcal{M},\ k\in[K]\}} \leq 1$. The expected value of $z_{i,j,m,k}$ is $\mathbb{E}[z_{i,j,m,k}] = \frac{profit_{i,j,m,k}}{\Gamma}\cdot\tilde{x}_{i,j,m,k}$. We thus have

$$\mathbb{E}[\sum_{i\in U}\sum_{j\in N}\sum_{m\in\mathcal{M}}\sum_{k\in[K]} z_{i,j,m,k}]$$

$$= \sum_{i\in U}\sum_{j\in N}\sum_{m\in\mathcal{M}}\sum_{k\in[K]}\frac{profit_{i,j,m,k}\cdot\tilde{x}_{i,j,m,k}}{\Gamma} = \frac{\widetilde{OPT}}{\Gamma}. \quad (16)$$

Let α be a constant with $1/2 < \alpha \leq 1$. By Chernoff Bounds [22], we have

$$\mathbf{Pr}[\sum_{i\in U}\sum_{j\in N}\sum_{m\in\mathcal{M}}\sum_{k\in[K]} profit_{i,j,m,k}\cdot\hat{x}_{i,j,m,k} \leq (1-\alpha)\cdot OPT]$$

$$\leq \mathbf{Pr}[\sum_{i\in U}\sum_{j\in N}\sum_{m\in\mathcal{M}}\sum_{k\in[K]} profit_{i,j,m,k}\cdot\hat{x}_{i,j,m,k} \leq (1-\alpha)\cdot\widetilde{OPT}],$$

$$= \mathbf{Pr}[\sum_{i\in U} Z_i \leq (1-\alpha)\cdot\frac{\widetilde{OPT}}{\Gamma}],$$

$$\text{let } Z_i = \sum_{j\in N}\sum_{m\in\mathcal{M}}\sum_{k\in[K]} profit_{i,j,m,k}/\Gamma\cdot\hat{x}_{i,j,m,k}$$

$$\leq exp(\frac{-\alpha^2\cdot\widetilde{OPT}}{2\Gamma}), \quad \text{by the Chernoff Boundst[22].} \quad (17)$$

With $0 < \alpha < 1$, the value of α can be set by assuming that $exp(\frac{-\alpha^2\cdot\widetilde{OPT}}{2\Gamma}) = \frac{1}{|U|}$, and $\alpha = \sqrt{\frac{2\Gamma\ln|U|}{\widetilde{OPT}}}$.

Based on (17), we have

$$\mathbf{Pr}\left[\sum_{i\in U}\sum_{j\in N}\sum_{m\in\mathcal{M}}\sum_{k\in[K]} profit_{i,j,m,k}\cdot\hat{x}_{i,j,m,k} \leq (1-\alpha)\cdot OPT\right] \leq \frac{1}{|U|} \quad (18)$$

Thus, the total profit is at least $(1-\alpha)\cdot OPT$ with high probability $1-\frac{1}{|U|}$. Due to that $0<\alpha<1$, it requires that $\sqrt{2\Gamma\ln|U|/\widetilde{OPT}} < 1$, i.e., $\widetilde{OPT} - 2\Gamma\ln|U| > 0$.

We then calculate the computing capacity violations on each cloudlet $j \in N$ as follows. Let $p_{i,j,m,k}$ be a random variable derived from random variable $\hat{x}_{i,j,m,k}$, i.e., $p_{i,j,m,k} = \frac{s_{m,k}}{\Gamma}\cdot\hat{x}_{i,j,m,k}$, and its expectation is $\mathbb{E}[p_{i,j,m,k}] = \frac{s_{m,k}}{\Gamma}\cdot\tilde{x}_{i,j,m,k}$ with $0 \leq p_{i,j,m,k} \leq 1$, because $\frac{s_{m,k}}{\Gamma}\cdot\hat{x}_{i,j,m,k} \leq \frac{s_{m,k}}{\max\{s_{m,k} \mid \forall i\in U,\ k\in[K]\}} \leq 1$. For each cloudlet $j\in N$, we have $\mathbb{E}[\sum_{i\in U}\sum_{m\in\mathcal{M}}\sum_{k\in[K]} p_{i,j,m,k}] = \sum_{i\in U}\sum_{m\in\mathcal{M}}\sum_{k\in[K]}\frac{s_{m,k}\cdot\tilde{x}_{i,j,m,k}}{\Gamma} = \frac{L\cdot\widetilde{C}_j}{\Gamma}$, where $\widetilde{C}_j$ is the amount of computing resource consumed on cloudlet j by the LP and $\widetilde{C}_j \leq C_j$. Let β be constant with $0<\beta\leq 1$. Recall that the computing resource consumption on any cloudlet j is $\sum_{i\in U}\sum_{m\in\mathcal{M}}\sum_{k\in[K]} s_{m,k}\cdot\hat{x}_{i,j,m,k}$ by `Algorithm 1`. Then, the probability of computing capacity violation of cloudlet $j\in N$ is

$$\mathbf{Pr}\left[\bigvee_{j\in N}\sum_{i\in U}\sum_{m\in\mathcal{M}}\sum_{k\in[K]} s_{m,k}\cdot\hat{x}_{i,j,m,k} \geq (1+\beta)\cdot L\cdot C_j\right]$$

$$\leq \mathbf{Pr}[\bigvee_{j\in N}\sum_{i\in U}\sum_{m\in\mathcal{M}}\sum_{k\in[K]} s_{m,k}\cdot\hat{x}_{i,j,m,k} \geq (1+\beta)\cdot L\cdot\widetilde{C}_j],$$

$$\leq \sum_{j\in N}\mathbf{Pr}[\sum_{i\in U}\sum_{m\in\mathcal{M}}\sum_{k\in[K]} p_{i,j,m,k} \geq (1+\beta)\cdot\frac{L\cdot\widetilde{C}_j}{\Gamma}],$$

by $p_{i,j,m,k} = \frac{s_{m,k}}{\Gamma}\cdot\hat{x}_{i,j,m,k}$ and the Union Bound Inequality

$$\leq \sum_{j\in N}\mathbf{Pr}[\sum_{i\in U} Q_{i,j} \geq (1+\beta)\cdot\frac{L\cdot\widetilde{C}_j}{\Gamma}],$$

$$\text{let } Q_{i,j} = \sum_{m\in\mathcal{M}}\sum_{k\in[K]} p_{i,j,m,k}$$

$$\leq |N|\cdot exp\left(\frac{-\beta^2\cdot L\cdot\widetilde{C}_j}{(2+\beta)\cdot\Gamma}\right), \quad \text{by the Chernoff Bounds [22].} \quad (19)$$

Let $exp(\frac{-\beta^2\cdot L\cdot\widetilde{C}_j}{(2+\beta)\cdot\Gamma}) \leq \frac{1}{|N|^2}$. Since $0<\beta\leq 1$, we have $exp(\frac{-\beta^2\cdot L\cdot\widetilde{C}_j}{6\Gamma}) \leq \frac{1}{|N|^2}$, i.e., $\beta \geq \sqrt{\frac{6\Gamma\ln|N|}{L\cdot\widetilde{C}_j}} \geq \sqrt{\frac{6\Gamma\ln|N|}{L\cdot C_j}}$ due to $\widetilde{C}_j \leq C_j$. As $\beta\leq 1$, we have $C_j \geq \frac{6\Gamma\ln|N|}{L}$, i.e., $\min\{C_j \mid j\in N\} \geq \frac{6\Gamma\ln|N|}{L}$. By (19), we have

$$\mathbf{Pr}[\bigvee_{j\in N}\sum_{i\in U} Q_{i,j} \geq (1+\beta)\cdot\frac{L\cdot\widetilde{C}_j}{\Gamma}] \leq |N|\cdot\frac{1}{|N|^2} = \frac{1}{|N|}.$$

Note that the number of instances of model resolution $M_{m,k}$ in cloudlets is at least $\lceil\frac{\sum_{i\in U}\hat{x}_{i,j,m,k}}{L}\rceil$ $(\frac{\sum_{i\in U}\hat{x}_{i,j,m,k}}{L}+1)$, which ensures that the number of instances is an integer. Due to $1+\beta\leq 2$, the computing resource consumption on any cloudlet $j\in N$ for all resolution instances is no more than $2+\kappa$ times its capacity with high probability of $1-\frac{1}{|N|}$, provided that $\min\{C_j \mid \forall j\in N\} \geq 6\Gamma\ln|N|/L$ and $\kappa = \max\{\frac{\sum_{m\in\mathcal{M}}\sum_{k\in[K]} s_{m,k}}{C_j} \mid j\in N\}$.

We finally analyze the time complexity of `Algorithm 1`. The calculation of $\mathcal{E}(i)$ for all requests takes $O(|U|\cdot|N|\cdot K)$

time. The time complexity of an algorithm for the linear relaxation LP is detailed below. According to [3], an LP with n variables can be solved in time $O(n^{2+\frac{1}{6}}\log(\frac{n}{\delta}))$, where δ is the relative accuracy. The LP here contains $|U|\cdot|N|\cdot|\mathcal{M}|\cdot K+|U|$ variables. By setting $\delta=1$ [19], an exact solution for the LP can be found in $O((|U|\cdot|N|\cdot|\mathcal{M}|\cdot K)^{2+\frac{1}{6}}\cdot\log(|U|\cdot|N|\cdot|\mathcal{M}|\cdot K))$ time, and the rounding on the LP solution can be done in $O(|U|\cdot|N|^2\cdot K^2)$ time. The time complexity of `Algorithm 1` thus is $O((|U|\cdot|N|\cdot|\mathcal{M}|\cdot K)^{2+\frac{1}{6}}\cdot\log(|U|\cdot|N|\cdot|\mathcal{M}|\cdot K))$.

The theorem then follows. □

V. Online Algorithm for the Dynamic Profit Maximization Problem

In this section, we deal with the dynamic profit maximization problem. We first formulate an ILP solution to the offline version problem at each time slot t with $t\in\mathbb{T}$, based on the predicted number of different resolution instances of each model. We then devise an online algorithm with a provable competitive ratio for the problem.

A. Prediction on the Number of Different Model Resolution Instances

We deal with the development of effective prediction mechanisms. There are several prediction methods. One is the DRL method, we can adopt the LSTM method to train a CNN model by analyzing historical traces, or the auto-regression method. However, both the algorithms may take a while to deliver a solution, which may not be feasible if the duration of each time slot is not long enough. We adopt a simple yet fast prediction method that keeps the usage record of the number of resolution instances of each model deployed in each cloudlet. That is, given an integral threshold $\theta\geq 1$, for each resolution instance of an existing model in a cloudlet, there is a counter to count the number of usages of the instance. For each instance, if it is not used at the current time slot, its counter is increased by 1; otherwise, its counter is reduced to 0. If an instance of a model resolution in a cloudlet has not been used for $\theta>1$ consecutive time slots, the instance will be removed and its occupied resource will be released back to the system in the end of the current time slot. For the sake of convenience, let $n^t_{j,m,k}$ be the number of instances of existing resolution $M_{m,k}$ of model M_m in cloudlet j at time slot $t\in\mathbb{T}$, which is the number of instances of $M_{m,k}$ in cloudlet j with their counters being less than θ. It can be seen that the accumulative amount of computing resource demanded by these pre-deployed instances in each cloudlet is no larger than the computing capacity on the cloudlet.

B. ILP for the Offline Version of the Problem

Given each time slot t, we formulate an ILP solution for the offline version of the dynamic profit maximization problem, where U^t is the set of requests at time slot t. Let $x^t_{i,j,m_i,k}$ be a binary variable that indicates whether request $i\in U^t$ is assigned to a newly installed instance of model resolution $M_{m_i,k}$ in cloudlet j at time slot t or not, and let $y^t_{i,j,m_i,k}$ be a binary variable that indicates whether request i is assigned to an existing instance of model resolution $M_{m_i,k}$ in cloudlet j or not. Each resolution instance $M_{m_i,k}$ consumes the amount $s_{m_i,k}$ of computing resource. Recall that $n^t_{j,m_i,k}$ is the number of existing resolution instances of $M_{m_i,k}$ in cloudlet j at time slot t. Then, the objective of the problem is to maximize the total profit obtained by admitting requests in U^t. The residual computing capacity C^t_j in the cloudlet j in the time slot t is then $C^t_j=C_j-\sum_{m=1}^{M}\sum_{k=1}^{K}n^t_{j,m,k}\cdot s_{m,k}$.

The ILP solution **LP1** for the admissions of requests in U^t at time slot t is presented as follows.

$$\textbf{LP1}: \text{Maximize} \sum_{i\in U^t}\sum_{j\in N}\sum_{m\in\mathcal{M}}\sum_{k\in[K]} profit_{i,j,m,k}\cdot(x^t_{i,j,m,k}+y^t_{i,j,m,k}) \tag{20}$$

s.t.

$$\sum_{m\in\mathcal{M}}\sum_{k\in[K]}\sum_{i\in U^t}x^t_{i,j,m,k}\cdot s_{m,k}\leq L\cdot C^t_j,\forall j\in N, \tag{21}$$

$$\sum_{j=1}^{|N|}\sum_{k=1}^{K}(d_{i,j,m_i,k}+init_{m_i,k})\cdot x^t_{i,j,m_i,k}\leq D_i,\forall i\in U^t, \tag{22}$$

$$\sum_{j=1}^{|N|}\sum_{k=1}^{K}d_{i,j,m_i,k}\cdot y^t_{i,j,m_i,k}\leq D_i,\forall i\in U^t, \tag{23}$$

$$\sum_{i\in U^t}y^t_{i,j,m,k}\leq L\cdot n^t_{j,m,k},\forall j\in N,\forall m\in\mathcal{M},\forall k\in[K] \tag{24}$$

$$x^t_{i,j,m,k}\cdot a_i\leq\epsilon_{m,k},\forall i\in U^t,j\in N,m\in\mathcal{M},k\in[K] \tag{25}$$

$$y^t_{i,j,m,k}\cdot a_i\leq\epsilon_{m,k},\forall i\in U^t,j\in N,m\in\mathcal{M},k\in[K] \tag{26}$$

$$\sum_{j\in N}\sum_{m\in\mathcal{M}}\sum_{k=1}^{K}(x^t_{i,j,m,k}+y^t_{i,j,m,k})\leq 1,\forall i\in U^t, \tag{27}$$

$$x^t_{i,j,m,k}=0,\quad\text{and}\quad y^t_{i,j,m,k}=0\,\text{if}\,m\neq m_i,$$
$$\forall i\in U^t,\forall j\in N,k\in[K] \tag{28}$$

$$x^t_{i,j,m,k}\in\{0,1\},\forall i\in U^t,\ \forall j\in N,m\in\mathcal{M},k\in[K] \tag{29}$$

$$y^t_{i,j,m,k}\in\{0,1\},\forall i\in U^t,\ \forall j\in N,m\in\mathcal{M},k\in[K], \tag{30}$$

where Constraint (21) ensures that the accumulated computing resource demanded by all newly installed model resolution instances at cloudlet j is no greater than its available capacity C^t_j. Constraints (22) and (23) ensure that the end-to-end delay of a user request is no greater than its delay requirement. Constraint (24) ensures that each existing model resolution instance can serve no more than L same type of service requests. Constraint (25) and (26) ensure that the accuracy of a chosen resolution instance for the processing of request i is no less than its accuracy requirement a_i. Constraint (27) ensures that each request i is served by at most one its requested resolution instance. Notice that the set U^t of requests at each time slot t is not known prior to time slot t.

C. Online Algorithm

In the following, we devise an online algorithm for the dynamic profit maximization problem with a provable competitive ratio, by adopting the primal-dual dynamic updating method [2], [9]. Since there is no knowledge of future request arrivals, we adopt an admission control policy to regulate request admissions to maximize the accumulative profit of admitted requests at each time slot t. For a given request $i \in U^t$, if it is processed by an instance in cloudlet j, the profit brought by its admission is calculated by distinguishing into two cases: (i) there is an existing resolution instance in cloudlet j that meets its accuracy requirement; and (ii) instantiating a new resolution instance of its requested model in cloudlet j meets its accuracy requirement, and such an instantiating delay is taken into account too.

The basic idea of the proposed algorithm is to adopt the primal-dual dynamic updating technique [2]. Let $U = \cup_{t=1}^{|\mathbb{T}|} U^t$ be the sequence of requests that arrive in the system one by one. Specifically, **LP1** can be equivalently rewritten as **LP2** by removing Constraints (22), (23), (25), and (26). Let **LP3** be a linear relaxation of **LP2** and **DP3** be the dual of **LP3**. The dynamic updating of the dual variables in **DP3** will deliver a feasible solution to **LP2**, which in turn will return a feasible solution to **LP1**.

Denote by $\mathcal{N}^t(i)$ and $\mathcal{E}^t(i)$ the sets of pairs of cloudlet j and resolution index k of a model that can fulfill the delay requirement D_i and accuracy requirement a_i of request i by either a newly instantiated instance or an existing resolution instance $M_{m_i,k}$ of model M_{m_i} in cloudlet j, respectively, i.e.,

$$\mathcal{N}^t(i) = \{(j,k) \mid vol_i/B_i + vol_i \cdot d(path_{l_i,j}) + \tau_{m_i,k} + init_{m_i,k}) \le D_i \quad \text{and } \epsilon_{m_i,k} \ge a_i\}, \tag{31}$$

$$\mathcal{E}^t(i) = \{(j,k) \mid vol_i/B_i + vol_i \cdot d(path_{l_i,j}) + \tau_{m_i,k} \le D_i, \; n^t_{j,m_i,k} \ge 1 \quad \text{and } \epsilon_{m_i,k} \ge a_i\}. \tag{32}$$

LP1 then can be rewritten as the following one.

$$\textbf{LP2}: \text{ Maximize } \sum_{i=1}^{|U^t|}\Big(\sum_{(j,k)\in\mathcal{N}^t(i)} profit_{i,j,m_i,k} \cdot x^t_{i,j,m_i,k} + \sum_{(j,k)\in\mathcal{E}^t(i)} profit_{i,j,m_i,k} \cdot y^t_{i,j,m_i,k}\Big),$$
$$\text{subject to } (21), (24), (27), (29), \text{ and } (30). \tag{33}$$

Recall that **LP3** is the linear relaxation of **LP2**, and **DP3** is the dual of **LP3**. Then, **DP3** is given as follows.

$$\textbf{DP3}: \text{Minimize } \sum_{j=1}^{|N|}(L \cdot C^t_j) \cdot \alpha^t_j + \sum_{i=1}^{|U^t|}\gamma^t_i + \sum_{j=1}^{|N|}\sum_{m=1}^{|\mathcal{M}|}\sum_{k=1}^{K} L \cdot n^t_{j,m,k} \cdot \beta^t_{j,m,k}, \tag{34}$$

subject to

$$\alpha^t_j \cdot s_{m_i,k} + \gamma^t_i \ge profit_{i,j,m_i,k}, \forall i,j,k \tag{35}$$

$$\gamma^t_i + \beta^t_{j,m_i,k} \ge profit_{i,j,m_i,k}, \forall i,j,k \tag{36}$$

$$\alpha^t_j \ge 0, \beta^t_{j,m_i,k} \ge 0, \gamma^t_i \ge 0, \tag{37}$$

where α^t_j, $\beta^t_{j,m_i,k}$, and γ^t_j are dual variables of constraints (21), (24), and (27), respectively.

Combining (35) and (36), we have

$$\gamma^t_i \ge \max\{profit_{i,j,m_i,k} - \alpha^t_j \cdot s_{m_i,k}, \; profit_{i,j,m_i,k} - \beta^t_{j,m_i,k} \mid \forall j,k\}, \quad \forall i \tag{38}$$

Define constants $\phi_1, \phi_2, N_{\max}, R_{\max}, a$ and b as follows. $\phi_1 = \max\{\frac{profit_{i,j,m_i,k}}{s_{m_i,k}} \mid i \in U^t,\ j \in N,\ m_i \in \mathcal{M},\ k \in [K],\ t \in \mathbb{T}\}$, $\phi_2 = \min\{\frac{profit_{i,j,m,k}}{s_{m,k}} \mid i \in U^t,\ j \in N,\ m \in \mathcal{M},\ k \in [K],\ t \in \mathbb{T}\}$, $N_{\max} = \max\{\frac{1}{L \cdot n^t_{j,m,k}} \mid j \in N,\ m \in \mathcal{M},\ k \in [K],\ t \in \mathbb{T}\}$, $R_{\max} = \max\{\frac{s_{m,k}}{C^t_j} \mid j \in N,\ m \in \mathcal{M},\ k \in [K], t \in \mathbb{T}\}$, $a = (1+R_{\max})^{\frac{1}{R_{\max}}}$, and $b = (1+N_{\max})^{\frac{1}{N_{\max}}}$.

The *admission control policy* of the proposed online algorithm is given as follows. Given each arrived request $i \in U^t$, if $\gamma^t_i > 0$, then request i is admitted, otherwise rejected. Once i is admitted, the corresponding pair $(j^*,k^*) \in \mathcal{N}^t(i) \cup \mathcal{E}^t(i)$ for the calculation of γ^t_i is determined, where

$$(j^*,k^*) = \underset{j,k}{\text{argmax}}\{\{profit_{i,j,m_i,k} - s_{m_i,k} \cdot \alpha^t_j \mid (j,k) \in \mathcal{N}^t(i)\}, \{profit_{i,j,m_i,k} - \beta^t_{j,m_i,k} \mid (j,k) \in \mathcal{E}^t(i)\}\}. \tag{39}$$

This implies that an instance of model resolution M_{m_i,k^*} in cloudlet j^* will be allocated to request i, all dual variables in constraints of **DP3** will be updated accordingly, and detailed updating rules are presented below.

$$\gamma^t_i \leftarrow \max\{\{profit_{i,j,m_i,k} - s_{m_i,k} \cdot \alpha^t_j \mid (j,k) \in \mathcal{N}^t(i)\}, \{profit_{i,j,m_i,k} - \beta^t_{j,m_i,k} \mid (j,k) \in \mathcal{E}^t(i)\}\}, \tag{40}$$

$$\alpha^t_{j^*} \leftarrow \alpha^t_{j^*} \cdot (1 + \frac{s_{m_i,k^*}}{C^t_{j^*}}) + \frac{\phi_1}{a-1} \cdot \frac{s_{m_i,k^*}}{C^t_{j^*}}, \tag{41}$$

$$\beta^t_{j^*,m_i,k^*} \leftarrow \beta^t_{j^*,m_i,k^*}(1 + \frac{1}{L \cdot n^t_{j^*,m_i,k^*}}) + \frac{profit_{i,j^*,m_i,k^*}}{L \cdot (b-1) n^t_{j^*,m_i,k^*}}. \tag{42}$$

The detailed online algorithm for the dynamic profit maximization problem is presented in `Algorithm 2`.

D. Algorithm Analysis

We first show the feasibility of the solution delivered by the dual linear program, **DP3**, in the following lemma.

Lemma 2: Following updating rules (40), (41), and (42), the dual feasibility of the dual variables is preserved.

Proof: Upon the arrival of request i, all dual variables will be updated when request i is admitted. As dual variables γ^t_i, α^t_j,

Algorithm 2: Online Algorithm for the Dynamic Profit Maximization Problem for Requests in U^t at Time Slot $t \in \mathbb{T}$.

Input: A set $\mathcal{M}$ of service models with each model having K different resolutions, requests U^t, the available computing resource capacity C_j^t on cloudlet j at time slot t, and the number $n_{j,m,k}^t$ of existing instances for model resolution $M_{m,k}$ in cloudlet j at time slot t.

Output: Find a solution to maximize the accumulative profit of requests admitted for time horizon $\mathbb{T}$, subject to the residual computing capacity on each cloudlet.

Initialize $x_{i,j,m_i,k}^t, y_{i,j,m_i,k}^t, \alpha_j^t, \beta_{j,m_i,k}^t, \gamma_i^t \leftarrow 0, \forall i,j,k$;
$A(t) \leftarrow 0$ /* the total profit at time slot t */;
while an arrived request $i \in U^t$ **do**
  for each cloudlet $j \in N$ **do**
    for $k \leftarrow 1$ to K **do**
      if pair (j,k) meets the requirements of $\mathcal{N}^t(i)$ and $\mathcal{E}^t(i)$ **then**
        Add (j,k) to $\mathcal{N}^t(i)$ and $\mathcal{E}^t(i)$;
      end if ;
    end for ;
  end for ;
  $\gamma_i^t \leftarrow \max\{ \{profit_{i,j,m_i,k} - s_{m_i,k} \cdot \alpha_j^t \mid (j,k) \in \mathcal{N}^t(i)\}, \{profit_{i,j,m_i,k} - \beta_{j,m_i,k}^t \mid (j,k) \in \mathcal{E}^t(i)\}\}$, by Rule (40);
  if $\gamma_i^t > 0$ **then**
    Admit request i by assigning it to an instance of model resolution M_{m_i,k^*} in cloudlet j^*;
    $A(t) \leftarrow A(t) + profit_{i,j^*,m_i,k^*}$;
    if $(j^*,k^*) \in \mathcal{N}^t(i)$ **then**
      $x_{i,j^*,m_i,k^*}^t \leftarrow 1$;
      $\alpha_{j^*}^t \leftarrow \alpha_{j^*}^t \cdot (1 + \frac{s_{m_i,k^*}}{C_{j^*}^t}) + \frac{\phi_1}{a-1} \cdot \frac{s_{m_i,k^*}}{C_{j^*}^t}$, by Rule (41);
    else
      $y_{i,j^*,m_i,k^*}^t \leftarrow 1$;
      $\beta_{j^*,m_i,k^*} \leftarrow \beta_{j^*,m_i,k^*} \cdot (1 + \frac{1}{L \cdot n_{j^*,m_i,k^*}^t}) + \frac{profit_{i,j^*,m_i,k^*}}{L \cdot (b-1) n_{j^*,m_i,k^*}^t}$, by Rule (42);
    end if
  end if
end while ;
return $A(t)$

and $\beta_{j,m_i,k}^t$ are initialized as 0s when $t = 0$, the updated value of variable γ_i^t must be no less than 0 by the updating rule (40), and the updated values of variables α_j^t and $\beta_{j,m_i,k}^t$ increase by updating rules (41) and (42). Thus, for all dual variables, the updating rules always keep them non-negative, and the dual constraint (37) is met. By updating rule (40), the dual constraints (35) and (36) are met, too. The fractional solution of the dual linear program **DP3** of **LP3** thus is a feasible solution to **DP3**. □

We then analyze the net gain relationship of the objective function values between **LP3** and **DP3** after the admission of request $i \in U^t$ by the following lemma.

Lemma 3: Let $\Delta P(t)$ and $\Delta D(t)$ be the value gains of the primary linear program **LP3** and its dual **DP3** by online algorithm `Algorithm 2` to respond to request $i \in U^t$ at time slot t, respectively. Then, $\Delta D(t) \leq \Delta P(t) \cdot H_{\max}$, where

$$H_{\max} = \max\{ \frac{1}{b-1} + 1, \frac{1}{a-1} \cdot \frac{\phi_1}{\phi_2} + 1\}. \quad (43)$$

Proof: When request $i \in U^t$ is admitted by a resolution instance of its requested model M_{m_i} in cloudlet j^*, the dual variables for cloudlet j^* are updated by distinguishing into two cases.

Case 1: If request i is admitted by an existing resolution instance M_{m_i,k^*} in cloudlet j^*, i.e., $(j^*,k^*) \in \mathcal{E}^t(i)$, then the values of dual variables γ_i^t and β_{j^*,m_i,k^*}^t are updated by rules (40) and (42), respectively, and $\Delta\beta_{j^*,m_i,k^*}^t = \frac{\beta_{j^*,m_i,k^*}^t}{L \cdot n_{j^*,m_i,k^*}^t} + \frac{profit_{i,j^*,m_i,k^*}}{L \cdot (b-1) n_{j^*,m_i,k^*}^t}$. Then,

$$\begin{aligned}
\Delta D(t) &= L \cdot n_{j^*,m_i,k^*}^t \cdot \Delta\beta_{j^*,m_i,k^*}^t + \gamma_i^t \\
&= L \cdot n_{j^*,m_i,k^*}^t \cdot \left(\frac{\beta_{j^*,m_i,k^*}^t}{L \cdot n_{j^*,m_i,k^*}^t} + \frac{profit_{i,j^*,m_i,k^*}}{L \cdot (b-1) \cdot n_{j^*,m_i,k^*}}\right) \\
&\quad + (profit_{i,j^*,m_i,k^*} - \beta_{j^*,m_i,k^*}^t) \\
&= profit_{i,j^*,m_i,k^*} \cdot \left(\frac{1}{b-1} + 1\right).
\end{aligned}$$

As $\Delta P(t) = profit_{i,j^*,m_i,k^*}$, we have $\frac{\Delta D(t)}{\Delta P(t)} = \frac{1}{b-1} + 1$.

Case 2: If request i is admitted by a new resolution instance of M_{m_i,k^*} in cloudlet j^*, i.e., $(j^*,k^*) \in \mathcal{N}^t(i)$, then the dual variables γ_i^t and $\alpha_{j^*}^t$ are updated by rules (40) and (41), respectively. The gain value of the dual is

$$\begin{aligned}
\Delta D(t) &= \Delta\alpha_{j^*}^t \cdot C_{j^*}^t + \gamma_i^t \\
&= \left(\alpha_{j^*}^t \cdot \frac{s_{m_i,k^*}}{C_{j^*}^t} + \frac{\phi_1}{a-1} \cdot \frac{s_{m_i,k^*}}{C_{j^*}^t}\right) \cdot C_{j^*}^t \\
&\quad + profit_{j^*,m_i,k^*} - s_{m_i,k^*} \cdot \alpha_{j^*}^t \\
&= \frac{\phi_1}{a-1} \cdot s_{m_i,k^*} + profit_{i,j^*,m_i,k^*}.
\end{aligned}$$

Since $\Delta P(t) = profit_{i,j^*,m_i,k^*}$, we have

$$\begin{aligned}
\frac{\Delta D(t)}{\Delta P(t)} &= \frac{\phi_1}{a-1} \cdot \frac{s_{m_i,k^*}}{profit_{i,j^*,m_i,k^*}} + 1 \\
&\leq \frac{\phi_1}{a-1} \cdot \frac{1}{\phi_2} + 1.
\end{aligned}$$

By Case 1 and 2 and the definition of $H_{\max}$ in (43), $\Delta D(t) \leq \Delta P(t) \cdot H_{\max}$, and the lemma follows. □

Finally, we analyze the usage of residual computing resources and the number $n_{j,m,k}^t$ of existing resolution instances of model $M_{m,k}$ in cloudlet j after admitting request i at time slot t by the following three lemmas.

Lemma 4: After the arrival of request i, the dual variable α_j^t corresponding to the computing capacity on cloudlet $j \in N$ satisfies

$$\alpha_j^t \geq \frac{\phi_1}{a-1} \cdot (a^{\frac{\sum_{i \in U^t} \sum_{k=1}^{K} x_{i,j,m_i,k}^t \cdot s_{m_i,k}}{C_j^t}} - 1) \tag{44}$$

Proof: We show the lemma by induction. As all dual variables are set to 0 s initially, (44) holds.

We assume that (44) holds for all requests in U^t with indices no greater than i. We now admit request $i' \in U^t$ by a resolution instance $M_{m_{i'},k'}$ of model $M_{m_{i'}}$ in cloudlet j with $i < i'$. Let $\alpha_j^t(start)$ and $\alpha_j^t(end)$ be the values of α_j^t before and after the admission of request i'. Then,

$$\alpha_j^t(end) = \alpha_j^t(start) \cdot (1 + \frac{s_{m_{i'},k'}}{C_j^t}) + \frac{\phi_1}{a-1} \cdot \frac{s_{m_{i'},k'}}{C_j^t}$$

$$\geq \frac{\phi_1}{a-1} \cdot (a^{\frac{\sum_{i<i'} \sum_{k=1}^{K} x_{i,j,m_i,k}^t \cdot s_{m_i,k}}{C_j^t}} - 1) \cdot (1 + \frac{s_{m_{i'},k'}}{C_j^t}) + \frac{\phi_1}{a-1} \cdot \frac{s_{m_{i'},k'}}{C_j^t} \tag{45}$$

$$= \frac{\phi_1}{a-1}((1 + \frac{s_{m_{i'},k'}}{C_j^t}) \cdot a^{\frac{\sum_{i<i'} \sum_{k=1}^{K} x_{i,j,m_i,k}^t \cdot s_{m_i,k}}{C_j^t}} - 1) \tag{46}$$

$$\geq \frac{\phi_1}{a-1}(a^{\frac{\sum_{i \in U^t} \sum_{k=1}^{K} x_{i,j,m_i,k}^t \cdot s_{m_i,k}}{C_j^t}} - 1) \tag{47}$$

By the induction assumption, (45) holds. Now, since $\frac{1}{x}\log(1+x)$ is a decreasing function with $x > 0$ and $R_{\max} \geq \frac{s_{m_{i'},k'}}{C_j^t}$, the inequality from (46) to (47) holds by the following arguments.

$$\log a = \frac{1}{R_{\max}} \log(1 + R_{\max}) \leq \frac{C_j^t}{s_{m_{i'},k'}} \log(1 + \frac{s_{m_{i'},k'}}{C_j^t}). \tag{48}$$

Ineq. (48) can be rewritten as $a^{\frac{s_{m_{i'},k'}}{C_j^t}} \leq 1 + \frac{s_{m_{i'},k'}}{C_j^t}$. The lemma then follows. □

Lemma 5: After the arrival of request $i \in U^t$, the dual variable $\beta_{j,m_i,k}^t$ corresponding to the number $n_{j,m_i,k}^t$ of existing instances of model resolution $M_{m_i,k}$ in cloudlet j satisfies

$$\beta_{j,m_i,k}^t \geq \frac{profit_{i,j,m_i,k}}{b-1} \cdot (b^{\frac{\sum_{i \in U^t} y_{i,j,m_i,k}^t}{L \cdot n_{j,m_i,k}^t}} - 1) \tag{49}$$

Proof: The proof is similar to Lemma 4, omitted. □

Lemma 6: In the solution of `Algorithm 2`, for any cloudlet $j \in N$, the violation on the residual computing capacity C_j^t is bounded by $\sum_{m \in \mathcal{M}} \sum_{k \in [K]} s_{m,k} + \max\{s_{m,k} \mid 1 \leq m \leq M, 1 \leq k \leq K\}$, and the violation on the number of different resolution instances of each model is at most one of that type instance, respectively.

Proof: We first analyze the usage of C_j^t. By Lemma 4, if the computing capacity constraint (21) is violated, i.e., $\sum_{i \in U^t} \sum_{m \in \mathcal{M}} \sum_{k \in [K]} x_{i,j,m_i,k}^t \cdot s_{m_i,k} > C_j^t$, then $\alpha_j^t \geq \phi_1$. As a result,

$$profit_{i,j,m_i,k} - \alpha_j^t \cdot s_{m_i,k} \leq profit_{i,j,m_i,k} - s_{m_i,k} \cdot \phi_1 \leq profit_{i,j,m_i,k} - s_{m_i,k} \cdot \frac{profit_{i,j,m_i,k}}{s_{m_i,k}} = 0,$$

By the updating rules (40) and (41), all the parameters related to the computing capacity of cloudlet j will not be updated anymore, and for all later arriving requests i' requesting a resolution instance of model $M_{m_{i'}}$ in cloudlet j, we have $x_{i',j,m_{i'},k} = 0$. Constraint (21) thus can be violated by at most once. Since the number of instances of model resolution $M_{m,k}$ is at least $\lceil \frac{\sum_{i \in U} \hat{x}_{i,j,m,k}}{L} \rceil$ ($\frac{\sum_{i \in U} \hat{x}_{i,j,m,k}}{L} + 1$), the computing capacity violation on any cloudlet j is upper bounded by $\sum_{m \in \mathcal{M}} \sum_{k \in [K]} s_{m,k}$ plus the maximum amount of computing resource consumed by deploying a model instance, i.e., $\sum_{m \in \mathcal{M}} \sum_{k \in [K]} s_{m,k} + \max\{s_{m,k} \mid 1 \leq m \leq M, 1 \leq k \leq K\}$.

The violation on the number of different resolution instances of each model can be analyzed similar to the capacity violation, omitted. □

Theorem 2: Given an MEC network $G = (N, E)$, there is an online algorithm, `Algorithm 2`, with a competitive ratio of $\frac{1}{H_{\max}}$ for the dynamic profit maximization problem, at the expense of bounded violations on the number of pre-installed resolution instances of each model and on the capacity of each cloudlet. The running time of `Algorithm 2` takes $O(|N| \cdot K)$ time for the admission per request at each time slot $t \in \mathbb{T}$, where $H_{\max}$ is defined in (43).

Proof: Let OPT_1, OPT_2 and OPT_3 be the optimal solutions for **LP1**, **LP2**, and **LP3**, respectively. Since **LP3** is the LP relaxation of the maximization problem **LP2**, $OPT_3 \geq OPT_2$. Denote by $\mathcal{P}$ and $\mathcal{D}$ the feasible solutions of **LP3** and **DP3** delivered by `Algorithm 2`, respectively. By Lemma 3, we have that $\mathcal{D} \leq \mathcal{P} \cdot H_{\max}$, and by the weak duality property, we have that $\mathcal{D} \geq OPT_3$. Then

$$\mathcal{P} \geq \frac{\mathcal{D}}{H_{\max}} \geq \frac{OPT_3}{H_{\max}} \geq \frac{OPT_2}{H_{\max}} = \frac{OPT_1}{H_{\max}}.$$

It can be seen that the resource violations in the solution by `Algorithm 2` is shown in Lemma 6. The time complexity of `Algorithm 2` is analyzed as follows. For each arrived request $i \in U^t$ at time slot t, finding $\mathcal{N}^t(i)$ and $\mathcal{E}^t(i)$ takes $O(|N| \cdot K)$ time by examining all potential pairs of (j,k). If request i is admitted, the updatings of dual variables γ_i^t, α_j^t, and $\beta_{j,m_i,k}^t$ take O(1) time. `Algorithm 2` thus takes $O(|N| \cdot K)$ time per request at each time slot $t \in \mathbb{T}$. □

VI. Performance Evaluation

In this section, we evaluated the performance of the proposed algorithms and investigated the impacts of important parameters on the performance of the proposed algorithms.

TABLE II
ACCURACY OF EACH RESOLUTION FOR DIFFERENT INFERENCE SERVICE MODELS

Model	Inference accuracy			
	240p	360p	480p	720p
R-FCN [6]	0.102	0.315	0.512	0.817
SSD [20]	0.103	0.252	0.491	0.818
YOLOv2 [29]	0.111	0.265	0.512	0.865

TABLE III
PARAMETERS OF EACH MODEL RESOLUTION INSTANCE

Function type	240p	360p	480p	720p
CPU cores	0.5vCPU	0.5vCPU	1vCPU	1vCPU
Processing time	[20, 30]ms	[40, 50]ms	[60, 70]ms	[80, 90]ms
Inference cost	[$0.01, $0.05]	[$0.06, $0.10]	[$0.11, $0.15]	[$0.16, $0.20]

A. Experimental Environment Settings

We considered an MEC network that consists of 20 APs, which is generated by NetworkX [24]. Each cloudlet j has computing capacity C_j that is randomly drawn between 30 virtual CPU (vCPU) cores and 35 vCPU cores [26]. The bandwidth capacity of each AP is randomly chosen from 2,000 Mbps to 3,000 Mbps at each time slot [36]. The cost ξ of a unit bandwidth resource (Mbps) is $0.001 [34]. The transmission delay for transmitting one MB of data along a link $e \in E$ is drawn in $[0.01, 0.05]$ millisecond [37]. The communication cost ζ_e on each link $e \in E$ is randomly drawn from $0.0001 to $0.0005 per MB [16]. The signal-to-interference-plus-noise ratio $\frac{P_i \cdot H_{i,l_i}}{\sigma^2 + I_{l_i}}$ between a user and an AP is randomly drawn in the range of $[10, 30]$ dB [42]. We adopted three popular inference service models: R-FCN [6], SSD [20], and YOLOv2 [29]. Each model has four different resolutions with each resolution having a different accuracy and inference time. Specifically, the four different resolutions of each model are 240p, 360p, 480p, and 720p ($K = 4$), respectively, and the accuracy of a different resolution of each model is shown in Table II.

We conducted the simulation through image detection as shown in Table III, each service request i contains images with data volume randomly drawn from $[0.5, 2]$ MB [37] with the delay requirement D_i randomly drawn between 100 and 450 milliseconds [4]. Request i requests an inference model M_{m_i} and its minimum accuracy requirement a_i is randomly chosen from 0 to the maximum accuracy of its requested inference model. Each resolution instance of a model can serve up to L (=2) same type of requests with their accuracy requirements no greater than the accuracy of this resolution instance. For a resolution instance of an inference service, the range of processing delay, the allocated amount of computing resource, and the range of inference cost $\varphi_{m,k}$ are listed in Table III [1], [30]. We further assume that the instantiating delay $init_{m,k}$ of each new resolution instance of a model is randomly drawn in [100, 200] milliseconds [31]. We calculate the payment pay_i of request $i \in U$ by adopting a stair-step function, where both model resolution accuracy and service delay are divided into four levels. Thus, the payments of different accuracy and delay levels for a requested model are given by these these three parameters of the stair-step function.

To evaluate `Algorithm 1` for the profit maximization problem, the proposed ILP solution for the problem is served as one benchmark. Also, following the suggestion by one anonymous referee, there is another ILP solution without resource violation for the problem, referred to as ILP_A, by introducing an auxiliary variable $X_{j,m,k}$ to transform Constraint (6) into the following two constraints:

$$\sum_{m=1}^{|\mathcal{M}|} \sum_{k=1}^{K} s_{m,k} \cdot X_{j,m,k} \leq C_j, \forall j \in N$$

$$\sum_{i \in U} x_{i,j,m_i,k} \leq L \cdot X_{j,m_i,k}, \forall j \in N, \forall M_m \in \mathcal{M}, \forall k \in [K],$$

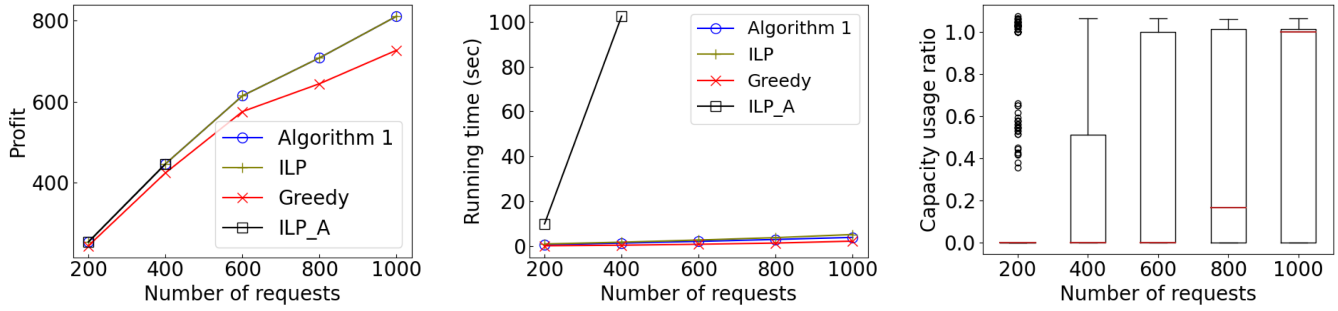
where $X_{j,m,k}$ is an integer variable, representing the number of VM instances of model resolution $M_{m,k}$ in cloudlet j. In addition to ILP_A as benchmark for the problem, we also proposed a greedy algorithm `Greedy` as benchmark as well. `Greedy` admits requests one by one greedily. Within each iteration, only the request with the maximum ratio of the profit it brought to the computing capacity of its requested model instance will be admitted, i.e., if request i is admitted and processed by instance $M_{m_i,k}$ in cloudlet j while satisfying $d_{i,j,m_i,k} \leq D_i$ and $a_i \leq \epsilon_{m_i,k}$, then its ratio $\frac{profit_{i,j,m_i,k}}{s_{m_i,k}}$ must be the maximum one among yet-to-be admitted requests.

For the dynamic profit maximization problem, we assumed that time horizon $\mathbb{T}$ consists of 50 time slots. To evaluate `Algorithm 2` for the dynamic profit maximization problem, two benchmark algorithms `No_control` and `No_pre` are adopted, which are the variants of `Algorithm 2` without the admission control policy and without pre-installed instances of any model resolutions in the network. In `Algorithm 2` and `No_control`, the number of pre-installed instances of each model resolution at each time slot is at least one to avoid the denominator becoming 0 when calculating $N_{\max}$. The threshold θ is set to 2 by default.

The value of each figure is the mean of the results out of 25 network topological instances with the same size. The running time of an algorithm is obtained on a desktop equipped with an Intel(R) Xeon(R) Platinum 8280 2.70 GHz CPU, with 10 GB RAM. The aforementioned parameters will be adopted by default unless otherwise specified.

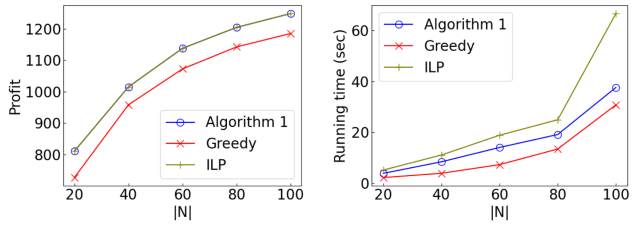
B. Performance of Different Algorithms for the Profit Maximization Problem

We first investigated the performance of the proposed algorithm `Algorithm 1` against those of `Greedy`, ILP, and the ILP_A, by varying the number of requests from 200 to 1000 while fixing L at 2. With the growth on the number of requests, we can observe from Fig. 2(a) and (b) that the performance of `Algorithm 1` is almost identical to the ILP solution and ILP_A solution, and the performance of `Greedy` is the worst one among them but with the shortest running time. When the problem size becomes large, ILP_A solution cannot be delivered within a practical running time. It can also be seen from Fig. 2(c) that the capacity usage ratio of cloudlets by `Algorithm 1` increases with the increase on the number of requests, as more service model instances need to be deployed for admitted requests. The empirical results in Fig. 2(c) indicates that the probability of resource violation is very small.



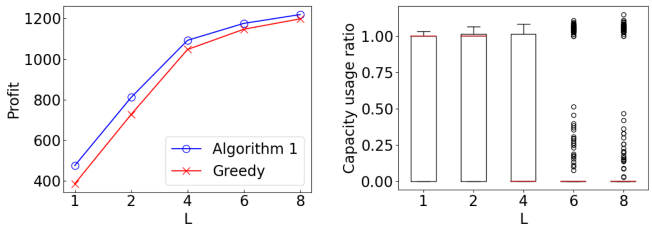
(a) Profit in dollars (b) Running time (c) Capacity usage ratio of Algorithm 1

Fig. 2. Performance of different algorithms for the profit maximization problem, by varying the number of requests.



(a) Accumulative profit in dollars (b) Running time

Fig. 3. Impact of network size $|N|$ on the performance of different algorithms for the profit maximization problem.



(a) Accumulative profit in dollars (b) Capacity usage ratio of Algorithm 1

Fig. 4. Impact of parameter L on the performance of different algorithms for the profit maximization problem.

We then studied the impact of network size $|N|$ on the performance of Algorithm 1, ILP, and Greedy while fixing the number of requests at 1,000. It can be seen from Fig. 3(a) that the accumulative profit of solution delivered by Algorithm 1 is at least 5% higher than that of Greedy for each given network size. Fig. 3(b) plots that the running time curves of different algorithms. With the increase on $|N|$, the running time of ILP rapidly increases, compared with the other comparison algorithms including Algorithm 1. In other words, Algorithm 1 exhibits a better scalability than the ILP.

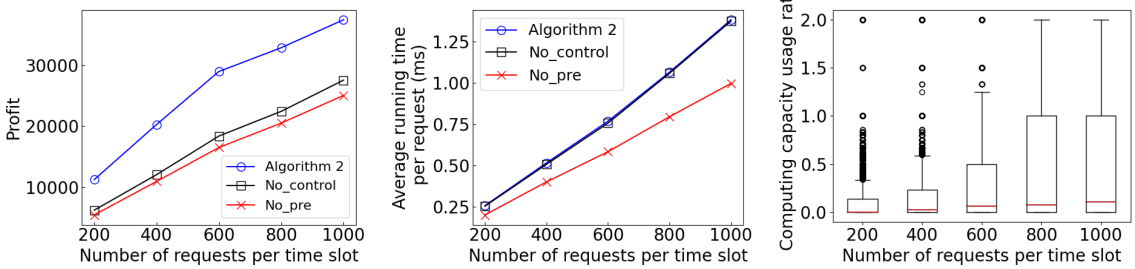
We also studied the impact of parameter L on the performance of Algorithm 1 and Greedy while fixing the number of requests at 1,000. Fig. 4(a) depicts that the accumulative profit of solution delivered by Algorithm 1 is higher than that of Greedy for each given value of L. It can be seen from Fig. 4(b) that with the increase on L, the capacity usage ratios of cloudlets by Algorithm 1 decrease, because more requests can be processed by a single resolution instance.

C. Performance of Different Algorithms for the Dynamic Profit Maximization Problem

We then evaluated the performance of Algorithm 2 against those of algorithms No_control and No_pre, by varying the number of arrived requests per time slot from 200 to 1,000. We can observe from Fig. 5(a) that the accumulative profit of the solution delivered by Algorithm 2 is at least 35% higher than that of No_control, and their profit gap becomes larger and larger as the number of arrived requests increases. This indicates that Algorithm 2 admits requests intelligently through its admission control policy while No_control does not. Fig. 5(c) shows the residual computing capacity usage ratios, i.e., the usage ratios of C_j^t, of cloudlets in Algorithm 2.

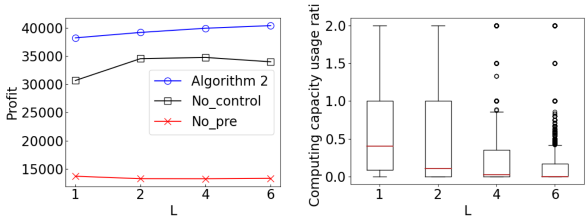
What followed is to investigate the impact of parameter L on the performance of three comparison algorithms Algorithm 2, No_control, and No_pre, while fixing the number of requests at 1,000 per time slot. Fig. 6(a) showss that Algorithm 2 has the highest accumulative profit while No_pre has the lowest one for different values of L. Fig. 6(b) depicts that the residual capacity usage ratios in Algorithm 2 decrease with the increase on the value of L. This is because a larger L means that each instance can process more requests and thus less instances are created for request processing.

We studied the impact of threshold θ on the performance of Algorithm 2, by drawing its value as 2, 4, 8, and 16 while fixing the number of requests per time slot at 1,000. Fig. 7(a) shows the impact of θ on the performance of Algorithm 2 for requests with shorter delay requirement in [100, 200] ms. It can be seen from Fig. 7(a) that with the increase on the value of θ, the accumulative profit delivered by Algorithm 2 increases first and then gradually decreases. With a larger θ, more idle instances without instantiating delays can be used to process requests, and thus more requests can be admitted by meeting their delay requirements. However, when θ is too large, the



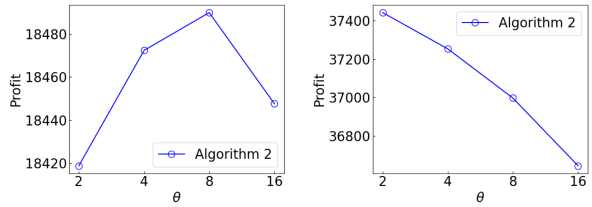
(a) Accumulative profit in dollars (b) Running time (c) Capacity usage ratio of Algorithm 2

Fig. 5. Performance of the dynamic profit maximization problem, by varying the number of requests.



(a) Accumulative profit in dollars (b) Capacity usage ratio of Algorithm 2

Fig. 6. Impact of L on the performance of algorithms for the dynamic profit maximization problem.

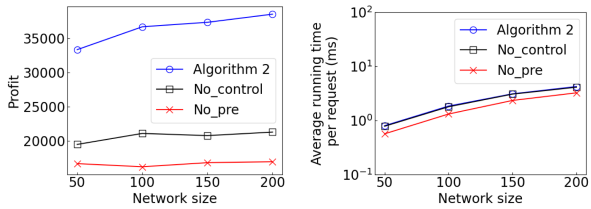


(a) Requests with shorter delay requirements (b) Requests with larger delay requirements

Fig. 7. Impact of θ on the performance of Algorithm 2 for the dynamic profit maximization problem.

instance deployment cannot be effectively adjusted according to the needs of user requests. Fig. 7(b) shows the impact of θ on the performance of Algorithm 2 for requests with larger delay requirement in [300, 450] ms. It can be seen from Fig. 7(b) that the accumulative profit delivered by Algorithm 2 decreases with the increase on the value of θ. This is because users with larger delay requirements can tolerate the installation delays, and the increase of θ makes the deployment of model instances unable to be adjusted in time according the user needs.

We finally evaluated the impact of the network size $|N|$ on the performance of Algorithm 2 against No_control and No_pre, by varying the network size while fixing the number of requests at 600 per time slot. It can be seen from Fig. 8(b) that with the increase on the network size, the accumulative profit of the solution by Algorithm 2 increases. This is because when users are distributed in a larger network, there is sufficient bandwidth resource and computing resource. This enables more requests to be admitted while their requirements can be met.



(a) Accumulative profit in dollars (b) Running time

Fig. 8. Performance of the dynamic profit maximization problem, by varying the network size $|N|$.

VII. Conclusion

In this article, we studied delay-sensitive, differential accuracy inference service provisioning in an edge computing network. We formulated two novel profit maximization problems for service request admissions while meeting stringent delay and accuracy requirements of users. Specifically, we first proposed an ILP solution for the profit maximization problem if the problem size is small or medium; otherwise, we proposed a constant randomized approximation algorithm with high probability. Meanwhile, we considered dynamic admissions of requests by developing a simple yet efficient prediction mechanism to predict the number of different resolution instances of models of each cloudlet needed in the future. We then pre-deployed these predicted numbers of model resolution instances into cloudlets to reduce their instantiating times, thereby admitting more requests through meeting the delay requirements of the requests. Built upon the pre-deployed instances, we third devised an online algorithm with a provable competitive ratio for the dynamic profit maximization problem, by adopting the primal-dual dynamic updating technique. Finally, we evaluated the performance of the proposed algorithms through simulations. The simulation results demonstrate that the proposed algorithms are promising.

Acknowledgment

The authors appreciate for the three anonymous referees and the Associate Editor for their constructive comments and invaluable suggestions, which help us to improve the quality and presentation of the paper greatly.

REFERENCES

[1] Amazon Web Services, Inc, Amazon EC2 instance configuration. Accessed: Nov. 2023. [Online]. Available: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html

[2] N. Buchbinder, K. Jain, and J. Naor, "Online primal-dual algorithms for maximizing ad-auctions revenue," in *Proc. Eur. Symp. Algorithms*, 2007, pp. 253–264.

[3] M. B. Cohen, Y. T. Lee, and Z. Song, "Solving linear programs in the current matrix multiplication time," *J. ACM*, vol. 68, no. 1, pp. 1–39, 2021.

[4] J. Chen and X. Ran, "Deep learning with edge computing: A review," in *Proc. IEEE*, vol. 107, no. 8, pp. 1655–1674, Aug. 2019.

[5] D. Crankshaw, X. Wang, G. Zhou, M. J. Franklin, J. E. Gonzalez, and I. Stoica, "Clipper: A low-latency online prediction serving system," in *Proc. Usenix Symp. Netw. Syst. Des. Implementation*, 2017, pp. 613–627.

[6] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object detection via region based fully convolutional networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 379–387.

[7] L. Deng, G. Li, S. Han, L. Shi, and Y. Xie, "Model compression and hardware acceleration for neural networks: A comprehensive survey," in *Proc. IEEE*, vol. 108, no. 4, pp. 485–532, 2020.

[8] G. Gao, Y. Dong, R. Wang, and X. Zhou, "EdgeVision: Towards collaborative video analytics on distributed edges for performance maximization," *IEEE Trans. Multimedia*, vol. 23, pp. 9083–9094, 2024.

[9] M. X. Goemans and D. P. Williamson, "The primal-dual method for approximation algorithms and its application to network design problems," in *Book chapter of Approximation Algorithms for NP-hard Problems*, Boston, MA, USA: PWS publishing, pp. 144–191, 1997.

[10] J. Huang, Y. Gao, and W. Dong, "Elastic DNN inference with unpredictable exit in edge computing," in *Proc. 43rd Int. Conf. Distrib. Comput. Syst.*, 2023, pp. 293–304.

[11] G. Jocher and Jacksonargo, YoLov 5, 2020. [Online] Available: https://github.com/uttralytics/yolov5

[12] W. Ju, D. Yuan, W. Bao, L. Ge, and B. Zhou, "eDeepSave: Saving DNN inference using early exit during handovers in mobile edge environment," *ACM Trans. Sensor Netw.*, vol. 17, no. 3, pp. 1–28, 2021.

[13] J. Li et al., "Digital twin-enabled service provisioning in edge computing via continual learning," *IEEE Trans. Mobile Comput.*, vol. 23, no. 6, pp. 7335–7350, Jun. 2024.

[14] J. Li et al., "AoI-aware user service satisfaction enhancement in digital twin-empowered edge computing," *IEEE/ACM Trans. Netw.*, vol. 32, no. 2, pp. 1677–1690, Apr. 2024.

[15] J. Li, W. Liang, Y. Li, Z. Xu, X. Jia, and S. Guo, "Throughput maximization of delay-aware DNN inference in edge computing by exploring DNN model partitioning and inference parallelism," *IEEE Trans. Mobile Comput.*, vol. 22, no. 5, pp. 3017–3030, May 2023.

[16] J. Li, W. Liang, W. Xu, Z. Xu, Y. Li, and X. Jia, "Service home identification of multiple-source IoT applications in edge computing," *IEEE Trans. Mobile Comput.*, vol. 16, no. 2, pp. 1417–1430, Mar./Spr. 2023.

[17] Y. Li, W. Liang, and J. Li, "Profit driven service provisioning in edge computing via deep reinforcement learning," *IEEE Trans. Netw. Service Manag.*, vol. 19, no. 3, pp. 3006–3019, Sep. 2022.

[18] Y. Li, W. Liang, Z. Xu, W. Xu, and X. Jia, "Budget-constrained digital twin synchronization and its application on fidelity-aware queries in edge computing," *IEEE Trans. Mobile Comput.*, vol. 24, no. 1, pp. 165–182, Jan. 2025.

[19] J. Liu, G. Zhao, H. Xu, P. Yang, B. Wang, and C. Qiao, "Toward a service availability-guaranteed cloud through VM placement," *IEEE/ACM Trans. Netw.*, vol. 32, no. 5, pp. 3993–4008, Oct. 2024.

[20] W. Liu et al., "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 21–37.

[21] Z. Liu, Q. Lan, and K. Huang, "Resource allocation for multiuser edge inference with batching and early exiting," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 4, pp. 1186–1200, Apr. 2023.

[22] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2005.

[23] Y. Ma, W. Liang, Z. Xu, and S. Guo, "Profit maximization for admitting requests with network function services in distributed clouds," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 5, pp. 1143–1157, May 2019.

[24] NetworkX. Accessed: Jul., 2022. [Online]. Available: https://networkx.org/

[25] Nvidia NIVIDIAtensorrt, 2021. [Online]. Available: https://developer.nvidia.com/tensorrt

[26] T. Ouyang, X. Chen, L. Zeng, and Z. Zhou, "Cost-aware edge resource probing for infrastructure-free edge computing: From optimal stopping to layered learning," in *Proc IEEE Real-Time Syst. Symp.*, 2019, pp. 380–391.

[27] A. Paszke et al., "Pytorch: An imperative style, high-performance deep learning library," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 8024–8035.

[28] Y. Qiu, J. Liang, V. C. M. Leung, and M. Chen, "Online security-aware and reliability-guaranteed AI service chains provisioning in edge intelligence cloud," *IEEE Trans. Mobile Comput.*, vol. 23, no. 5, pp. 5933–5948, May 2024.

[29] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 6517–6525.

[30] K. Sheshadri and J. Lakshmi, "QoS aware FaaS platform," in *Proc 21st Int. Symp. Cluster, Cloud Internet Comput.*, 2021.

[31] X. Wei et al., "No provisioned concurrency: Fast RDMA-codesigned remote fork for serverless computing," in *Proc. USENIX Conf. Operating Syst. Des. Implementation*, 2022, pp. 497–517.

[32] A. Xu et al., "CoMS: Collaborative DNN model selection for heterogeneous edge computing systems," *IEEE Trans. Veh. Technol.*, early access, Sep. 27, 2024, doi: 10.1109/TVT.2024.3469281.

[33] J. Xie, Z. Zhou, T. Ouyang, X. Zhang, and X. Chen, "Fair: DNN Model Selection in Edge AI Via a Cooperative Game Approach. *Proc. IEEE 43rd Int. Conf. Distrib. Comput. Syst.*, 2023, pp. 383–394.

[34] Z. Xu et al., "HierFedML: Aggregator placement and UE assignment for hierarchical federated learning in mobile edge computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 34, no. 1, pp. 328–345, Jan. 2023.

[35] Z. Xu et al., "Energy-aware inference offloading for DNN-driven applications in mobile edge clouds," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 4, pp. 799–814, Apr. 2021.

[36] Z. Xu et al., "Energy-aware collaborative service caching in a 5G-enabled MEC with uncertain payoffs," *IEEE Trans. Commun.*, vol. 70, no. 2, pp. 1058–1071, Feb. 2022.

[37] Z. Xu et al., "Stateful serverless application placement in MEC with function and state dependencies," *IEEE Trans. Comput.*, vol. 72, no. 9, pp. 2701–2716, Sep. 2023.

[38] M. Yin, Y. Sui, S. Liao, and B. Yuan, "Towards efficient tensor decomposition-based DNN model compression with optimization framework," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 10674–10683.

[39] Y. Yang, Y. Shi, C. Yi, J. Cai, J. Kang, and D. Niyato, "Dynamic human digital twin deployment at the edge for task execution: A two-timescale accuracy-aware online optimization," *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 12262–12279, Dec. 2024.

[40] K. Zhao et al., "EdgeAdaptor: Online configuration adaption, model selection and resource provisioning for edge DNN inference serving at scale," *IEEE Trans. Mobile Comput.*, vol. 22, no. 10, pp. 5870–5886, Oct. 2023.

[41] Y. Zhang and W. Liang, "Cost-aware digital twin migration in mobile edge computing via deep reinforcement learning," in *Proc. 23rd IFIP Netw. Conf.*, 2024, pp. 441–447.

[42] Y. Zhang, W. Liang, Z. Xu, and X. Jia, "Mobility-aware service provisioning in edge computing via digital twin replica placements," *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 11295–11311, Dec. 2024.

[43] Y. Zhang, W. Liang, W. Xu, Z. Xu, and X. Jia, "Cost minimization of digital twin placements in mobile edge computing," *ACM Trans. Sensor Netw.*, vol. 20, no. 3, pp. 1–26, 2024.

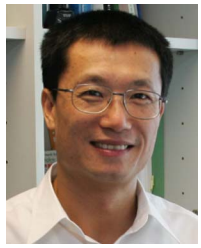
[44] Y. Zhang, W. Liang, Z. Xu, W. Xu, and M. Chen, "AoI-aware inference services in edge computing via digital twin network slicing," *IEEE Trans. Serv. Comput.*, vol. 17, no. 6, pp. 3154–3170, Nov./Dec. 2024.

[45] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, "Edge intelligence: Paving the last mile of artifical intelligence with edge computing. *Proc. IEEE*, vol. 107, no. 8, pp. 1738–1762, Aug. 2019.

Yuncan Zhang (Member, IEEE) received the PhD degree from Kyoto University, Kyoto, Japan, in 2021, the ME degree from the University of Science and Technology of China, Hefei, China, in 2016, and the BE degree from the Dalian University of Technology, Dalian, China, in 2013. She now is a post-doc with the Department of Computer Science at City University of Hong Kong. She worked with Baidu, Beijing, China, from 2016 to 2017. Her research interests include mobile edge computing, network function virtualization, and optimization problems.

Weifa Liang (Senior Member, IEEE) received the BSc degree from Wuhan University, China, in 1984, the ME degree from the University of Science and Technology of China, in 1989, and the PhD degree from the Australian National University, in 1998, all in computer science. He is a full professor with the Department of Computer Science, City University of Hong Kong. Prior to that, he was a full professor with the Australian National University. His research interests include design and analysis of energy efficient routing protocols for wireless ad hoc and sensor networks, mobile edge computing (MEC), network function virtualization (NFV), Internet of Things and digital twins, design and analysis of parallel and distributed algorithms, approximation algorithms, and graph theory. He currently serves as an editor for *IEEE Transactions on Communications*.

Zichuan Xu (Member, IEEE) received the BSc and ME degrees from the Dalian University of Technology, China, in 2008 and 2011, and the PhD degree from Australian National University, in 2016, all in computer science. From 2016 to 2017, he was a research associate with the Department of Electronic and Electrical Engineering, University College London, U.K.. He is currently a full professor and PhD advisor with the School of Software, Dalian University of Technology. His research interests include mobile edge computing, serverless computing, network function virtualization, algorithmic game theory, and optimization problems.



Xiaohua Jia (Fellow, IEEE) received the BSc and ME degrees from the University of Science and Technology of China, in 1984 and 1987, respectively, and the DSc degree in information science from the University of Tokyo, in 1991. He is currently a chair professor with the Department of Computer Science at City University of Hong Kong. His research interests include cloud computing and distributed systems, computer networks, wireless sensor networks and mobile wireless networks. He is an editor of *IEEE Transactions on Computers*, *IEEE Transactions on Parallel and Distributed Systems* (2006-2009), *Journal of World Wide Web*, *Wireless Networks*, *Journal of Combinatorial Optimization*, and so on. He is the general chair of ACM MobiHoc 2008, TPC co-chair of IEEE MASS 2009, area-chair of IEEE INFOCOM 2010, TPC co-chair of IEEE GlobeCom 2010, Ad Hoc and Sensor Networking Symposium, and Panel co-chair of IEEE INFOCOM 2011.

Yuanyuan Yang (Life Fellow, IEEE) received the BEng and MS degrees in computer science and engineering from Tsinghua University, Beijing, China, and the MSE and PhD degrees in computer science from Johns Hopkins University, Baltimore, MD, USA. She is currently a SUNY distinguished professor of computer engineering and computer science with Stony Brook University, Stony Brook, NY, USA. She is also on leave with the National Science Foundation as the program director. She has published more than 460 papers in major journals and refereed conference proceedings and holds seven U.S. patents in these areas. Her research interests include edge computing, data center networks, cloud computing, and wireless networks. She is currently the editor-in-chief for *IEEE Transactions on Cloud Computing* and an associate editor for *IEEE Transactions on Parallel and Distributed Systems* and *ACM Computing Surveys*. She has served as the associate editor-in-chief for *IEEE Transactions on Cloud Computing*, the associate editor-in-chief and an associate editor of *IEEE Transactions on Computers*, and an associate editor of *IEEE Transactions on Parallel and Distributed Systems*. She has also served as the general chair, program chair, or vice chair for several major conferences, and a program committee member for numerous conferences.