Inference Service Fidelity Maximization in DT-Assisted Edge Computing

Jing Li, Jianping Wang, *Fellow, IEEE*, Weifa Liang, *Senior Member, IEEE*, Xiaohua Jia, *Fellow, IEEE*, and Albert Y. Zomaya, *Fellow, IEEE*

***Abstract*—Digital twin (DT) technology enables smooth integrations of cyber and physical worlds in alignment with the Industry 4.0 initiative. DTs are virtual presentations of physical objects. Through synchronizations with physical objects in real-time, DTs can reflect the states of their objects with high fidelity. Orthogonal to the DT technology, mobile edge computing (MEC) is a promising computing paradigm that shifts computing power to the edge network, which is appropriate for delay-sensitive intelligent services. In this paper, we study fidelity-aware inference services in a DT-assisted MEC environment, where machine learning-based inference models must be continuously retrained using updated DT data in order to provide high-fidelity services for consumers. To this end, we first formulate two novel optimization problems: the initial DT and model placement problem with the aim of minimizing the total cost of various resources consumed for the placements, and the cumulative fidelity maximization problem to maximize the long-term cumulative fidelity of all service models while minimizing the cost of resource consumption on enhancements of service model fidelitiess over a given time horizon, through jointly scheduling mobile devices to synchronize with their DTs by uploading their update data and determining whether DTs and/or models to be migrated at each time slot. We then develop an efficient algorithm for the initial DT and model placement problem, through a reduction to a series of minimum-cost maximum matching problems in auxiliary graphs. We also devise an online algorithm with a provable competitive ratio for the cumulative fidelity maximization problem, by designing an elegant service request admission strategy. Finally, we evaluate the performance of the proposed algorithms via simulations. Simulation results demonstrate that the proposed algorithms are promising, and outperform their baselines by no less than 28%.**

***Index Terms*—Digital twins (DTs), DT-assisted edge computing, mobile devices, DT and model placements, service fidelity maximization, DT and service model migrations, cost modeling.**

Received 12 March 2025; revised 5 June 2025; accepted 16 August 2025. Date of publication 19 August 2025; date of current version 3 December 2025. The work of Jing Li, Jianping Wang, and Weifa Liang were supported by Hong Kong Research Grants Council (RGC) under the Collaborative Research Fund (CRF) under Grant C1042-23GF. The work of Jing Li was supported by the Post-Doc Fellowship Award from the Hong Kong RGC under Grant CityU PDFS2425-1S02. The work of Weifa Liang was supported by the Research Grants Council of the Hong Kong Special Administrative Region, China under Grant CityU 11202723, Grant CityU 11202824, Grant 7005845, Grant 8730094, and Grant 9380137, respectively. Recommended for acceptance by L. Bai. *(Corresponding author: Weifa Liang.)*

Jing Li, Jianping Wang, Weifa Liang, and Xiaohua Jia are with the Department of Computer Science, City University of Hong Kong, Kowloon Tong Hong Kong (e-mail: jing.li@cityu.edu.hk; jianwang@cityu.edu.hk; weifa.liang@cityu.edu.hk; csjia@cityu.edu.hk).

Albert Y. Zomaya is with the School of Computer Science, University of Sydney, Camperdown, NSW 2006, Australia (e-mail: albert.zomaya@sydney.edu.au).

Digital Object Identifier 10.1109/TMC.2025.3600390

I. Introduction

Accompanied by the widespread deployment and rapid adoption of 5G and beyond 5G networks, significant research endeavors are underway to drive the development of the next-generation network, known as the sixth generation (6G), via the Internet of Things (IoT) towards a fully intelligent future world [4], [26]. Holding immense potential in realizing the digitization goals of 6G, the concept of digital twins (DTs) is transitioning from a conceptual idea to a tangible simulation technology recently and continues to garner significant interest across various fields, including healthcare, autonomous driving, and manufacturing [25]. A DT is a digital representation of a physical object, which enables diverse functionalities such as simulation, analysis, prediction and optimization, due to advances in technologies, such as artificial intelligence and Big Data analytics [13]. DTs offer an opportunity to portray and model the physical world in a virtualized manner, serving as a connection between the physical and virtual realms.

Mobile edge computing (MEC) presents an excellent solution for building DTs to enhance intelligent services in future 6G networks, through bringing data processing and service model training to the network edge, in the proximity of users and IoT devices [11]. DTs are able to record the dynamic state information of their physical counterparts in real-time, while MEC facilitates ubiquitous low-latency intelligent services to handle the high update frequency of DTs with large amounts of update data and boost data processing efficiency [13].

Model-driven inference service provisioning in MEC environments has emerged as a recent prominent research focus [18]. In this work, we consider an IoT application scenario illustrated in Fig. 1, where there are multiple service models to provide different inference services, and each service model consists of multiple attributes (features), and the data of each attribute in the service model comes from the DT data of an object (a mobile device). Since each DT stores historical data of its physical object, these DT data are important for continuous model training in order to enable providing accurate model-driven services including predictive behaviors of their objects [5]. One such an example is that there are several service models providing inference results of testing autonomous vehicles in an area through simulations, using DTs of the vehicles for model training. Later simulations indicate that the service models based on DTs of different vehicles in an area can provide inference results to their users and facilitate the users for decision-making.

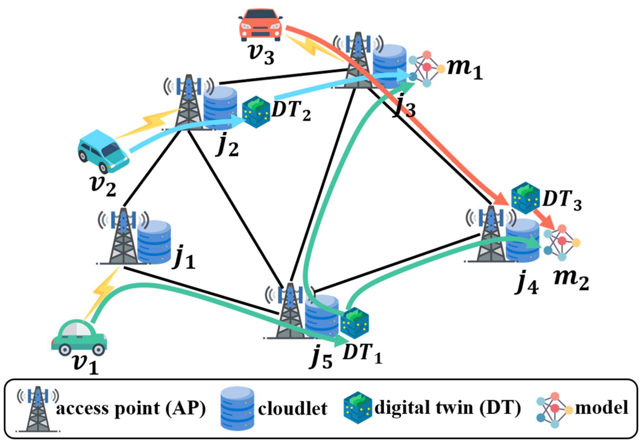


Fig. 1. An illustrative example of fidelity-aware inference service provisioning in a DT-assisted edge computing network, where there is a co-located cloudlet with each access point (AP). Objects v_1, v_2 and v_3 have their DTs, DT_1, DT_2 and DT_3, placed in cloudlets j_5, j_2, and j_4, respectively. There are two inference service models, m_1 and m_2, that are placed in cloudlets j_3 and j_4 respectively; model m_1 consists of attributes DT_1 and DT_2, while model m_2 consists of attributes DT_1 and DT_3, respectively.

Since most service models, including the above example of autonomous driving, require to be trained continuously to maintain their service fidelity, using the update data from their sources, how to maintain the high fidelity of service models while utilizing limited computing and storage resources in edge computing poses the following challenges. First, as the DTs of different objects provide the input data of service models for model training, what is the impact of the DT update data on the fidelity of service models? and how to measure the fidelity contribution of the update data of each object to its service models? Second, mobile devices (objects) move around in the network. Both DT locations and model locations determine the DT and model updating costs that include uploading the update data of objects, transmitting the update data to DTs and models, and training the models using the update data. How to place and migrate DT and service models in cloudlets without violating their computing capacities is challenging. Third, due to the uncertainty on the volume of the update data generated by each mobile device since its last uploading, accurately predicting the volume of the update data of the mobile device is challenging. Finally, because of the limited bandwidth of APs, usually only a subset of mobile devices are able to upload their update data at each time slot. To maximize the cumulative fidelity of all service models while minimizing the sum of the DT update and model retraining costs, determining which mobile devices can upload their update data at each time slot must be addressed. In the rest of this paper, we will tackle the aforementioned challenges.

The novelty of this paper lies in the study of fidelity-aware inference service provisioning in a DT-assisted edge computing network, by formulating two novel optimization problems: the DT and model placement problem, and the cumulative fidelity maximization problem. Efficient algorithms are developed for the defined problems, through efficiently scheduling DTs of mobile objects and service models for their placements and migrations in the network. The main contributions of this paper are summarized as follows.

- We investigate fidelity enhancements of inference service models in a DT-assisted edge computing network, through efficiently scheduling DTs of mobile objects and service models for their placements and migrations. We develop novel metrics to measure the service fidelity of models and the monetary cost of various resources consumed on continuous model training, synchronizations between DTs and their mobile devices, and migrations of DTs and models.
- We develop an efficient algorithm for the initial placement problem of DTs and models into cloudlets, by reducing to a series of minimum-cost maximum matching problems in auxiliary bipartite graphs, assuming that the mobility profiles of mobile objects are given.
- We devise an online algorithm with a provable competitive ratio for the cumulative fidelity maximization problem, assuming that both DTs and models have been placed initially. We aim to maximize the cumulative service fidelity of all models while minimizing the total cost of resources consumed for service fidelity enhancement of the models, by scheduling mobile devices for uploading, and DT and model migrations at each time slot over a given time horizon.
- We evaluate the performance of the proposed algorithms through simulations. Simulation results demonstrate that the proposed algorithms are very promising, outperforming the comparison baselines by no less than 28%.

The remainder of the paper is as follows. Section II surveys the related work. Section III introduces the system model, notions and notations, measurement metrics, and problem definitions. Section IV focuses on the development of an algorithm for the initial DT and model placement problem. Section V devises an online algorithm for the cumulative fidelity maximization problem. Section VI provides the simulation results, and Section VII concludes the paper.

II. Related Work

AoI-aware DT placement and updating: DT technology has been envisioned as a promising technique for intelligent services in MEC networks [8], [9], [10], [11], [12], [13], [15], [16], [17], [18], [20], [27], [28], [29], [32], [33], [34], [35], [36], [37], [39]. For example, Li et al. [10] devised approximation and online algorithms for dynamic DT placements with the mobility assumption of objects, while ensuring the low age of information (AoI) of services on DT data. Li et al. [8] studied reliability-aware placements of service function chains (SFCs) in MEC by leveraging DTs to predict the reliability of virtual service function instances. Liang et al. [18] investigated the non-trivial relationships between the freshness of service models and the AoI of training data on service models in DT-empowered edge computing. They devised an efficient algorithm to minimize the cost of various resources consumed to achieve the accumulative freshness of service models. Li et al. [17] addressed the freshness issue of DTs by introducing freshness metrics and explored one potential application of data collection by using Unmanned Aerial Vehicles (UAVs), for which they proposed an efficient approximate solution through DT synchronization with their objects under bandwidth and computing resource constraints. Lin et al. [20] developed a congestion control scheme to ensure the stability of long-term DT services, using Lyapunov optimization. Ren et al. [27] presented a congestion control scheme for DT edge networks and developed a deep reinforcement learning (DRL) method for performance prediction of the physical network. Shu et al. [28] dealt with DT-assisted resource management through energy dispatching and control model training under the constraint of long-term AoI. They proposed a DT-assisted federated learning scheme for the problem, by adopting the Lyapunov optimization technique. Vaezi et al. [29] proposed algorithms for the DT deployment problem to reduce the maximum response delay while meeting user AoI requirements. They developed efficient algorithms for this cost minimization problem.

Mobility-aware DT and model placements and migrations: Zhang et al. [33], [35], [37] dealt with mobility-aware service provisioning in mobile edge computing via DT replica placements and DT migrations, by developing a randomized algorithm with high probability and a DRL algorithm for dynamic service requests. Zhang et al. [34] studied the DT placement and migration problem in an MEC network under the assumption of object and user mobility, with the aim of jointly optimizing the freshness of the DT data and the service cost of users who request DT services. Zhang et al. [36] investigated inference services in DT-assisted MEC networks by sharing limited network resources through digital twin network (DTN) slicing, for which they introduced a framework for DT and model instance placements, and developed efficient algorithms for DTN services while meeting different user service delay requirements. Zhang et al. [32] leveraged DTs for mobile device scheduling to maximize the utility of federated learning (FL) services. They developed efficient approaches for offline multi-FL services and a DRL algorithm under dynamic FL service requests, respectively. Zhao et al. [39] developed a hierarchical routing strategy in a DT-assisted vehicular edge network for service provisioning to vehicle users.

Fidelity-aware, DT-empowered model services: There are several efforts to improve model fidelity of DTs [1], [11], [19], [31]. For example, Chen et al. [1] leveraged DTs to represent the features of a physical system, using different resolution models with differential accuracy levels. They proposed a dynamic programming-based approximation algorithm to select an appropriate model for each DT to maximize the minimum feature accuracy. Li et al. [11] adopted the continual learning technique for model retraining of DTs to achieve timely DT synchronization, and designed efficient algorithms to improve the model accuracy. Liao et al. [19] investigated DT-empowered resource management in 6G networks by formulating an energy management optimization problem for electric vehicles. They developed an AoI-aware DRL algorithm that strives for fine balancing between the service fidelity of models and the differential level consistency of DTs with their objects. Yang et al. [31] devised an end-edge-cloud framework, where human DTs are established to offer human-centric services. They proposed an approach to optimize the accuracy of executing tasks assigned by human DTs, through utilizing the Lyapunov optimization technique.

Unlike the aforementioned studies, in this paper we explore non-trivial trades-off between the fidelity of service models and the monetary cost of resources consumed on model training over a given time horizon. We focus on efficient placements and migrations of DTs and service models, with the aim to maximize the cumulative fidelity of service models while minimizing the total cost of various resources consumed. It must be mentioned that this paper is an extension of the conference paper [15].

III. Preliminaries

In this section, we introduce the system model, notions, notations, and cost modeling. We also define problems and show their NP-hardness.

A. System Model

Consider an MEC network $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ that consists of a set $\mathcal{N}$ of access points (APs) and a set $\mathcal{E}$ of optic links between APs. Each AP has a co-located cloudlet that is interconnected by an optic cable, and their communication delay is negligible. Each AP $j \in \mathcal{N}$ has bandwidth capacity B_j and its co-located cloudlet has computing capacity C_j, respectively. In this paper, we assume that each AP and its co-located cloudlet are interchangeable if no confusion arises.

We consider a finite time horizon $\mathbb{T}$ that is divided into T equal time slots, i.e., $\mathbb{T} = \{1, 2, \ldots T\}$. There is a set V of mobile devices (objects) under the coverage of APs. Associated with each object $v_i \in V$, there is a DT, DT_i, to be placed in a cloudlet, which is a virtual representation of the object. The data generated by object v_i will be uploaded to DT_i at some time slots, and object v_i will be synchronized with DT_i by transmitting the update data from its uploading AP to the cloudlet hosting DT_i. Apparently, a DT location affects the freshness of the state and data of the DT, and thus impacts the accuracy and fidelity of inference services whose source data comes from the DT data.

There is a set M of inference service models (e.g., deep neural network (DNN) models), each model $m_k \in M$ consists of $l_k \geq 1$ attributes (features), where the data of the lth attribute of m_k comes from the DT data of an object, and the object (or its DT data) is referred to as the source data of model m_k if no confusion arises. We further assume that the total amount of time needed for uploading the update data, processing the update data at their DTs, and continual training on service models is no greater than the duration of a one-time slot, i.e., each DT synchronization and model training can be achieved within one-time slot [18].

B. Initial Placement Cost of DTs and Service Models

We deal with the initial placement of DTs and models by making use of top-K frequent visiting locations of each object as approximate representations of its movements in the MEC network. Since objects are movable, it is challenging to place the DT of each object to a location to minimize its subsequent updating (synchronization) cost when the object moves to other locations. However, it is observed that most objects do not move around all locations in a large-scale network [21], [35]. Instead, it usually only visits a very few locations. Let K be the number of the most frequent visiting locations of each object with $K \ll |\mathcal{N}|$. Notice that the top-K visiting locations of each object can be found through mining its historical movement traces stored in its DT. Now, let $\mathcal{N}(v_i)$ be the top-K visiting locations of object $v_i \in V$, and $p_{i,j}$ is the probability of its visit to location $j \in \mathcal{N}(v_i) \subset \mathcal{N}$, i.e., $0 \leq p_{i,j} \leq 1$ and $|\mathcal{N}(v_i)| = K$. We here assume that $\sum_{j \in \mathcal{N}(v_i)} p_{i,j} \approx 1$, that is, the probability sum of object v_i visiting the other locations $j \in \mathcal{N} \setminus \mathcal{N}(v_i)$ is very small that is negligible. With the mobility assumption of each object $v_i \in V$ in the MEC network, if DT_i of object v_i is deployed in cloudlet $h'(DT_i)$, then *the expected DT updating cost* of DT_i is defined as follows.

$$\begin{aligned}&\overline{cost}_{DT}(v_i, h'(DT_i))\\&\quad = cost_{ins}(DT_i) + \sum_{j \in N(v_i)} p_{i,j} \cdot \Bigg(\xi_1 \cdot PX_i \cdot \frac{\widetilde{vol}(v_i, j)}{\widetilde{R}_{i,j}}\\&\qquad + \xi_2 \cdot \widetilde{vol}(v_i, j) \cdot \sum_{e \in P_{j,h'(DT_i)}} length(e)\\&\qquad + \xi_3 \cdot \widetilde{vol}(v_i, j) \cdot f^2_{h'(DT_i)} \Bigg), \qquad (1)\end{aligned}$$

where $cost_{ins}(DT_i)$ is the instantiation cost of DT_i, $\widetilde{vol}(v_i, j)$ is the average volume of the update data by object v_i at location j with the average uploading rate $\widetilde{R}_{i,j}$ whose definition is similar the one of (3), ξ_1 is the cost of a unit power consumption, PX_i is the transmission power of device v_i, ξ_2 is the transferring cost of unit data via a routing path with length 1, $P_{j,h'(DT_i)}$ is the shortest path in the network G between AP j and AP $h'(DT_i)$, and $length(e)$ is the length of link e. ξ_3 is the cost per unit energy consumption, $f_{h'(DT_i)}$ is the CPU frequency of cloudlet $h'(DT_i)$ hosting the DT, DT_i of mobile device v_i, and the term $\xi_3 \cdot \widetilde{vol}(v_i, j) \cdot f^2_{h'(DT_i)})$ is the data processing cost in cloudlet $h'(DT_i)$ in terms of the amount of energy consumed [38].

Now, assume that DT_i has been deployed in cloudlet $h'(DT_i)$ and is one attribute's data source of model m_k. If model m_k is deployed in cloudlet $h'(m_k)$, then *the expected updating cost* of model m_k is defined as follows.

$$\begin{aligned}&\overline{cost}_{model}(m_k, h'(m_k)) = cost_{ins}(m_k)\\&\quad + \sum_{v_i \in V(m_k)} \xi_2 \cdot \sum_{j \in N(v_i)} p_{i,j} \cdot \widetilde{vol'}(v_i, j) \sum_{e \in P_{h'(DT_i), h'(m_k)}} length(e)\\&\quad + \xi_3 \cdot \sum_{j \in N(v_i)} p_{i,j} \cdot \widetilde{vol'}(v_i, j) \cdot f^2_{h'(m_k)}, \qquad (2)\end{aligned}$$

where $cost_{ins}(m_k)$ is the instantiation costs of an instance of model m_k, and $f_{h'(m_k,t)}$ is the CPU frequency of cloudlet $h'(m_k)$ hosting model m_k. Also, $\widetilde{vol'}(v_i, j)$ is the average volume of the processed data of DT_i by object v_i located at AP j, which is sent to the models requiring the DT data of DT_i, and $\widetilde{vol'}(v_i, j)$ is usually smaller than the volume $\widetilde{vol}(v_i, j)$ of the received update data by DT_i.

C. Bandwidth Allocation of Mobile Devices

Each object can be covered by a set of APs, let $AP(v_i, t)$ represent the set of APs that object $v_i \in V$ is under their coverage at time slot t. We here adopt the OFDMA scheme for the bandwidth allocation on each AP. We assume that the total bandwidth B_j on AP $j \in N$ is divided by $L_j \geq 1$ orthogonal sub-channels that supports up to L_j mobile devices to upload their update data via the AP at each time slot [7]. Assuming that device v_i chooses AP j for uploading at time slot t, its data uploading rate $R_{i,j}(t)$ to AP j at time slot t is

$$R_{i,j}(t) = \frac{B_j}{L_j} \cdot \log\left(1 + \frac{PX_i \cdot H_{i,j}}{\sigma^2}\right) \qquad (3)$$

where PX_i is the transmission power of device v_i, $H_{i,j}$ is the channel gain between device v_i and AP j, and σ^2 is the noise power.

If device v_i uploads the volume $vol(v_i, t)$ of update data to AP j at time slot t, then its uploading delay $\tau_{up}(v_i, j, t)$ is

$$\tau_{up}(v_i, j, t) = \frac{vol(v_i, t)}{R_{i,j}(t)}. \qquad (4)$$

Since the number of mobile devices under the coverage of AP j is usually larger than the number L_j of sub-channels of AP j, only some of them can upload their update data to their DTs at each time slot. The updated DT data are then forwarded to the service models that the DTs are their source data for the training of these models.

D. Fidelity Gain of Service Models via Retraining

Given a service model m_k with l_k attributes $attr_{m_k,1}, \ldots, attr_{m_k,l_k}$, the data of each attribute is the DT data of its corresponding object, and each attribute $attr_{m_k,l}$ is assigned to a fixed weight $w_{m_k,l}$ (≥ 0) to capture the importance of the attribute to the accuracy of the inference model. A larger weight implies the attribute heavily impacts on the fidelity of the model, and $\sum_{l=1}^{l_k} w_{k,l} = 1$.

Given an object v_i that is the data source of multiple service models, if its update data is uploaded at time slot t, then it will impact the service fidelity (or service accuracy) of each of the models. In the following, we quantify the fidelity gain on each model by uploading the update data of object v_i at time slot $t \in \mathbb{T}$.

Let $V(m_k)$ be the set of objects contributing to the l_k attributes of model m_k with $|V(m_k)| = l_k$. Let $V(m_k, t) \subseteq V(m_k)$ be a subset of objects of model m_k whose DT data are updated since last retraining of M_m by time slot t. Supposing that object v_i is the lth attribute of model m_k, denote by $\rho_{m_k,l}(t)$ the accumulative volume of the update data of object v_i at DT_i so far as the data of the lth attribute of model m_k. Then,

$$\rho_{k,l}(t) = \begin{cases} \rho_{k,l}(t-1) + vol(v_i, t) & \text{if } v_i \in V(m_k, t) \\ \rho_{k,l}(t-1) & \text{otherwise,} \end{cases}$$

It can be seen that $\rho_{k,l}(t) \geq \rho_{k,l}(t-1)$ and $\rho_{k,l}(0)=0$ initially for all l with $1 \leq l \leq l_k$.

The impact of the update data of object v_i on all service models in which it is their attribute is modeled as follows.

Let $F(m_k,t)$ be the fidelity of model m_k at time slot t with $F(m_k,0)=0$ initially. We make use of a submodular non-decreasing function $g(\cdot,\ldots,\cdot)$ with l_k parameters to measure the fidelity of model $m_k \in M$. Function $g(\cdot,\ldots,\cdot)$ maintains the diminishing fidelity gain property, i.e., if $g(\cdot)$ is a 1-dimensional function, we have $g(x+\delta x)-g(x) \leq g(x'+\delta x)-g(x')$ with $x'<x$ and $\delta x \geq 0$, e.g., $g(x)=\log_2(x+1)$. Assuming that the submodular function $g(\cdot,\ldots,\cdot)$ can be expressed by the weighted sum of 1-dimensional submodular non-decreasing function $f(\cdot)$, then the fidelity gain $\Delta G(m_k,v_i,t)$ of model m_k induced by object $v_i \in V(m_k)$ as its lth attribute at time slot t is defined as follows.

$$\Delta G(m_k,v_i,t) = w_{k,l}\cdot\left(f(\rho_{k,l}(t)) - f(\rho_{k,l}(t-1))\right), \qquad (5)$$

where $w_{k,l}$ is the weight of the lth attribute of model m_k with $\sum_{l=1}^{l_k} w_{k,l}=1$, and $\rho_{k,l}(t)$ is the accumulative volume of update data as the lth attribute of model m_k at time slots t. Notice that the fidelity gain is diminishing with the increase on the accumulative volume of update date of each attribute in model m_k as $f(\cdot)$ is a submodular function.

The fidelity gain $\Delta F(m_k,t)$ of model m_k at time slot t is defined as follows.

$$\Delta F(m_k,t) = \sum_{v_i\in V(m_k)} \Delta G(m_k,v_i,t). \qquad (6)$$

E. Update Cost of DTs and Service Models

The deployment of DTs and service models to cloudlets consumes computing resource. Let $comp(DT_i)$ and $comp(m_k)$ be the amounts of computing resource consumed by DT_i and model m_k, respectively. Let $AP(v_i,t)$ be the set of APs covering object v_i at time slot t. *The uploading cost* of the update data of v_i located at AP $j \in AP(v_i,t)$ at time slot t is

$$cost_{up}(v_i,j,t) = \xi_1 \cdot PX_i \cdot \tau_{up}(v_i,j,t) \qquad (7)$$

where ξ_1 is the cost of a unit energy consumption, PX_i is the transmission power of device v_i, and $\tau_{up}(v_i,j,t)$ is its data uploading duration that is defined in (4).

The data transmission cost of object v_i from AP j to its DT_i located at cloudlet $h(DT_i,t)$ at time slot t is

$$cost_{trans}(v_i,j,h(DT_i,t),t) = \xi_2 \cdot vol(v_i,t)\cdot \sum_{e\in P_{j,h(DT_i,t)}} length(e) \qquad (8)$$

where ξ_2 is the transferring cost of unit data via a routing path with length 1, $P_{j,h(DT_i,t)}$ is the shortest path in the network G between AP j and AP $h(DT_i,t)$, and $length(e)$ is the length of link $e \in \mathcal{E}$.

The DT processing cost of DT_i in cloudlet $h(DT_i,t)$ for the processing of the newly update data is

$$cost_{proc}(v_i,t) = \xi_3 \cdot vol(v_i,t)\cdot f^2_{h(DT_i,t)}, \qquad (9)$$

where $f_{h(DT_i,t)}$ is the CPU frequency of cloudlet hosting digital twin DT_i at time slot t.

To train model m_k using the update data of its attributes, all update data from the DTs of m_k must be forwarded to cloudlet $h(m_k,t)$ that hosts its instance at time slot t. *The DT data transmission cost* of model m_k for aggregation in cloudlet $h(m_k,t)$ is

$$cost_{agg}(m_k,h(m_k,t),t) = \xi_2 \cdot \sum_{v_i\in V(m_k,t)} vol'(v_i,t) \sum_{e\in P_{h(DT_i,t),h(m_k,t)}} length(e), \qquad (10)$$

where $vol'(v_i,t)$ is the volume of the processed data from DT_i at time slot t, which will be sent to the models requiring the DT data of DT_i, and $vol'(v_i,t)$ is usually smaller than the volume $vol(v_i,t)$ of the received update data by DT_i at time slot t.

The training cost of model m_k in cloudlet $h(m_k,t)$ at time slot t due to updating its source data is

$$cost_{train}(m_k,t) = \xi_3 \cdot \sum_{v_i\in V(m_k,t)} vol'(v_i,t)\cdot f^2_{h(m_k,t)}, \qquad (11)$$

where $f_{h(m_k,t)}$ is the CPU frequency of cloudlet hosting model m_k at time slot t.

F. Migration Cost of DTs and Service Models

Since mobile devices are very likely to move to different locations at different time slots, it is highly desirable that their DTs can be migrated from their current locations to new locations not far away from the mobile devices to reduce the transmission cost of their update data. Consequently, service models whose attributes are the DTs of these devices may need to migrate from their current locations to appropriate locations too in order to reduce their data transmission costs.

Suppose that DT_i in cloudlet $h(DT_i,t-1)$ at time slot $t-1$ will be migrated to cloudlet $h(DT_i,t)$ at time slot t. Because each DT stores all historical data of its object so far, its migration cost $cost_{mig}(DT_i,t)$ is proportional to the accumulative volume $a\_vol(DT_i,t)$ of all update data of object v_i in DT_i so far, and the length of a shortest routing path $P_{h(DT_i,t-1),h(DT_i,t)}$ in $\mathcal{G}$ between cloudlets $h(DT_i,t-1)$ and $h(DT_i,t)$ is $\sum_{e\in P_{h(DT_i,t-1),h(DT_i,t)}} length(e)$. Let $cost_{ins}(DT_i)$ and $cost_{ins}(m_k)$ be *the instantiation costs* of DT_i and an instance of model m_k, respectively.

The migration cost of DT_i at time slot t consists of its instantiation cost $cost_{ins}$ in cloudlet $h(DT_i,t)$ and the cost of migrating its DT data from cloudlet $h(DT_i,t-1)$ to cloudlet $h(DT_i,t)$, which is defined as follows.

$$cost_{mig}(DT_i,t) = \begin{cases} 0, \text{ if no migration for } DT_i \\ \xi_2 \cdot a\_vol(DT_i,t)\sum_{e\in P_{h(DT_i,t-1),h(DT_i,t)}} length(e) \\ \quad + cost_{ins}(DT_i), \text{ otherwise.} \end{cases} \qquad (12)$$

Assume that model m_k is hosted by cloudlet $h(m_k, t-1)$ at time slot $t-1$, and will be migrated to cloudlet $h(m_k, t)$ at time slot t. Let $size(m_k)$ be the parameter size of model m_k. *The migration cost* of model m_k at time slot t is

$$cost_{mig}(m_k, t) = \begin{cases} 0, \text{ if no migration of } m_k \\ \xi_2 \cdot size(m_k) \cdot \sum_{e \in P_{h(m_k,t-1),h(m_k,t)}} length(e) \\ \quad + cost_{ins}(m_k), \text{ otherwise} \end{cases} \tag{13}$$

G. Problem Definitions

In the following, we first define the initial placements of both DTs and service models. We then define the cumulative fidelity maximization problem over a finite time horizon.

Definition 1: Given an MEC network $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ with a set $\mathcal{N}$ of APs (cloudlets) and a set V of mobile objects, a given mobility profile of each object $v_i \in V$, the average volume $\widetilde{vol}(v_i, j)$ of the update data of object v_i per uploading with probability $p_{i,j}$, the average uploading rate $\widetilde{b}_{i,j}$, and the amount $comp(DT_i)$ of computing resource demanded, there is a set M of service models to be placed and retrained continuously, with the amount $comp(m_k)$ of computing resource demanded by each model $m_k \in M$ using the update data of its attributes' objects. *The initial placement problem of DTs and service models* in $\mathcal{G}$ is to minimize the expected cost of various resources consumed on the placement of DTs and service models, i.e., we aim to minimize the optimization objective (14), subject to computing capacity on each cloudlet.

The optimization objective of the initial placement problem is to minimize the total expected cost of various resources consumed, i.e.,

$$\text{Minimize} \sum_{v_i \in V} \overline{cost}_{DT}(v_i, h'(DT_i)) + \sum_{m \in M} \overline{cost}_{model}(m_k, h'(m_k)). \tag{14}$$

Assuming that both DTs and service models have been deployed in cloudlets initially, in the following we deal with user inference request admissions based on deployed service models over a given time horizon. We aim to admit as many user requests as possible by fully utilizing model-driven service provisioning. To keep high service fidelity of each service model, the model needs to be trained often using the update data from its source DTs. Suppose that a mobile device $v_i \in V$ uploads its update data via AP j while its DT_i is located at cloudlet $h(DT_i, t)$ at time slot t, *the DT updating cost* $cost_{DT}(v_i, t)$ of DT_i at time slot t will consist of the DT migration cost (12), the uploading cost (7), the data transmission cost (8), and the DT processing cost (9) respectively, which is defined as follows.

$$cost_{DT}(v_i, t) = cost_{mig}(v_i, t) + cost_{up}(v_i, j, t) + cost_{trans}(v_i, j, h(DT_i, t), t) + cost_{proc}(v_i, t) \tag{15}$$

Recall that $h(m_k, t)$ is the cloudlet hosting model m_k at time slot t. To enhance the service fidelity of model m_k, *the model updating cost* $cost_{model}(m_k, t)$ of model m_k consists of the model migration cost (13), the DT data transmission cost (10), and the training cost (11), which is defined as follows.

$$cost_{model}(m_k, t) = cost_{mig}(m_k, t) + cost_{agg}(m_k, h(m_k, t), t) + cost_{train}(m_k, t). \tag{16}$$

The cumulative fidelity maximization problem over time horizon $\mathbb{T}$ is to maximize the cumulative fidelity of all service models while minimizing the cost of various resources consumed on fidelity improvements of all service models over time horizon $\mathbb{T}$, i.e.,

$$\text{Maximize} \sum_{t=1}^{|\mathbb{T}|} \sum_{m_k \in M} \Delta F(m_k, t) - \omega \cdot \sum_{t=1}^{|\mathbb{T}|} \left(\sum_{v_i \in V} cost_{DT}(v_i, t) + \sum_{m_k \in M} cost_{model}(m_k, t) \right), \tag{17}$$

where $\omega > 0$ is a scaling coefficient to balance two different metrics: the cumulative fidelity and the total cost, $\Delta F(m_k, t)$ is the fidelity gain of model m_k at time slot t by (6), $cost_{DT}(v_i, t)$ and $cost_{model}(m_k, t)$ are the updating costs of DT_i and model m_k at time slot t by (15) and (16), respectively.

Assuming that both DTs and models have been placed, we now define the cumulative fidelity maximization problem over a time horizon $\mathbb{T}$ as follows.

Definition 2: Given an MEC network $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ with a set $|\mathcal{N}|$ of APs (cloudlets), a given time horizon $\mathbb{T}$, a set V of mobile devices with their placed DTs, and a set M of service models with their placed model instances, *the cumulative fidelity maximization problem* in $\mathcal{G}$ is to maximize the cumulative fidelity of service models while minimizing the total cost of resources consumed for model fidelity enhancement, i.e., we aim to maximize the optimization objective (17), through DT data updating, DT migrations, and service model training by using the updated DT data over time horizon $\mathbb{T}$, subject to computing capacity on each cloudlet and bandwidth capacity on each AP.

H. NP-Hardness

Theorem 1: The initial DT and model placement problem in an MEC network $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ is NP-hard.

Proof: The NP-hardness of the initial DT and model placement problem is proven by a reduction from the minimum-cost Generalized Assignment Problem (GAP), which is NP-hard [24]. Considering a set of bins with capacity for each bin and a set of items with a weight and cost for each item, the minimum-cost GAP is to minimize the total cost of packed items into bins, subject to the capacity of each bin.

We consider a special case of the initial DT and model placement problem, where the amount of consumed resources and the incurred cost of model training are negligible. Therefore, we only consider the placements of DTs in cloudlets. We treat each cloudlet j as a bin b_j with capacity C_j, i.e., the computing capacity on cloudlet j. We also treat each DT_i as an item v_i with weight $comp(DT_i)$. Assigning an item into bin b_j causes a cost $\overline{cost}_{DT}(v_i, j)$. We observe that this special problem is equivalent to the minimum-cost GAP. Thus, the initial DT and model placement problem is NP-hard. □

IV. Algorithm for the Initial DT and Model Placement Problem

In this section, we deal with the initial DT and model placement problem, by utilizing the mobility profiles of objects through analyzing their historical update data traces.

A. Algorithm

The initial placement problem of DTs and service models is a joint optimization problem that places both DTs and models to cloudlets. Since it is very intriguing to tackle this joint optimization problem, we instead decompose the problem into two sub-optimization problems: the DT placement problem, followed by the model placement problem. We deal with each of them by finding a series of minimum-cost maximum matching in corresponding auxiliary bipartite graphs.

To place DTs to cloudlets, we construct a series of auxiliary bipartite graphs iteratively. Within each iteration r with $r \geq 1$, the DTs of some objects will be placed. Specifically, the bipartite graph $B_{DT}(r) = (V(r), N(r), E(r))$ at round r is constructed as follows, where $V(r)$ is the set of objects whose DTs have not yet been placed with $V(1) = V$ initially, and $N(r)$ is the set of cloudlets with residual computing resource. $E(r)$ is the edge set in $B_{DT}(r)$, and there is an edge $(v_i, j) \in E(r)$ between object $v_i \in V(r)$ and cloudlet $j \in N(r)$ with weight $\overline{cost}_{DT}(v_i, j)$ by (1), if cloudlet j has adequate residual computing resource to host DT_i.

Let $MT_{DT}(r)$ be a minimum-cost maximum matching in bipartite graph $B_{DT}(r)$, which can be found by applying the Hungarian algorithm. Then, for each edge $(v_i, j) \in MT_{DT}(r)$, DT_i of object v_i is placed to cloudlet j with the amount $comp(DT_i)$ of computing resource demanded. This procedure continues until the DTs of all objects are deployed in cloudlets.

Having placed the DTs of all objects, we then place service models in M by adopting the same placement approach as we did for DT placements. That is, service model placements are implemented iteratively too. Within each iteration $r \geq 1$, we construct a bipartite graph $B_M(r) = (M(r), N'(r), E'(r))$, where $M(r)$ is the set of service models that have not been placed and $M(1) = M$ initially. $N'(r)$ is the set of cloudlets with residual computing resource. $E'(r)$ is the edge set of $B_M(r)$, and there is an edge $(m_k, j) \in E'(r)$ between a service model $m_k \in M(r)$ and a cloudlet $j \in N'(r)$ if cloudlet j has adequate residual computing resource to accommodate service model m_k, and the weight of edge (m_k, j) is $\overline{cost}_{model}(m_k, j)$ by (2).

Let $MT_{model}(r)$ be a minimum-cost maximum matching in $B_M(r)$. Then, for each edge $e(m_k, j) \in MT_{model}(r)$, model m_k is placed to cloudlet j with the amount $comp(m_k)$ of computing resource consumed. The amount of residual computing resource in cloudlet j then is reduced by $comp(m_k)$. This procedure continues until all models are deployed.

Algorithm 1: Algorithm for the Initial DT and Model Placement Problem.

```
Input: An MEC network $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, a set $V$ of mobile
  objects with each having its top-$K$ visiting location profile
  and demanding the amount $comp(DT_i)$ of computing
  resource for its DT, a set $M$ of service models that are
  retrained by the update data from mobile objects in $V$.
Output: Find a DT and model placement schedule for each
  object in $V$ and each model in $M$ such that the
  objective (14) is minimized.
1: /* DT deployments */
2: $r \leftarrow 1$; $V(r) \leftarrow V$; $N(r) \leftarrow \mathcal{N}$;
3: Construct the edge set $E(r)$, where an edge
     $e(v_i, j) \in E(r)$ has a weight $\overline{cost}_{DT}(v_i, j)$ by (1);
4: while $E(r) \neq \emptyset$ do
5:    Construct graph $B_{DT}(r) = (V(r), N(r), E(r))$;
6:    Find a minimum-cost maximum matching $MT_{DT}(r)$
        in $B_{DT}(r)$ by the Hungarian algorithm;
7:    for each edge $e(v_i, j) \in MT_{DT}(r)$ do
8:       Place $DT_i$ to cloudlet $j$;
9:    end for ;
10:   $r \leftarrow r + 1$;
11:   Update $V(r)$, $N(r)$, and $E(r)$ by removing matched
        objects;
12: end while ;
13: /* service model deployments */;
14: $r \leftarrow 1$; $M(r) \leftarrow M$;
15: Identify $N'(r)$ and $E'(r)$, where each edge
     $e(m_k, j) \in E'(r)$ has a weight $\overline{cost}_{model}(m_k, j)$;
16: while $E'(r) \neq \emptyset$ do
17:   Construct graph $B_M(r) = (M(r), N'(r), E'(r))$;
18:   Find a minimum-cost maximum matching
        $MT_{model}(r)$ in $B_M(r)$ by the Hungarian algorithm;
19:   for each edge $e(m_k, j)$ in $MT_{model}(r)$ do
20:      Place model $m_k$ to cloudlet $j$;
21:   end for ;
22:   $r \leftarrow r + 1$;
23:   Update $M(r)$, $N'(r)$ and $E'(r)$) by removing matched
        service models;
24: end while.
```

The proposed algorithm for the initial placement problem of DTs and service models is detailed in `Algorithm 1`.

B. Algorithm Analysis

Theorem 2: Given an MEC network $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, a set V of mobile objects, a set $\mathcal{N}$ of cloudlets, and a set M of service models that are retrained by the update data of mobile objects in V, there is an algorithm, `Algorithm 1`, for the initial placement problem of DTs and service models. The time complexity of `Algorithm 1` is $O(|V| \cdot (|V| + |\mathcal{N}|)^3 + |M| \cdot (|M| + |\mathcal{N}|)^3)$.

Proof: We first show that the solution delivered by `Algorithm 1` is feasible. Recall that $MT_{DT}(r)$ is a minimum-cost maximum matching of graph $B_{DT}(r)$ in iteration r. It can be seen that if there is one edge $(v_i, j) \in MT_{DT}(r)$ between an object $v_i \in V(r)$ and a cloudlet $j \in N(r)$, it means that cloudlet j has sufficient computing resource to host DT_i, and DT_i will be placed to it. The deployments of DTs thus are feasible. Similarly, the placements of service models to cloudlets by `Algorithm` 1 are feasible too.

We then analyze the time complexity of `Algorithm` 1. The dominant time complexity in DT placements is to construct a bipartite graph and find a minimum-cost maximum matching in the graph at each iteration r. There are at most $|V|$ iterations, the placements of all DTs thus takes $O(|V| \cdot (|V| + |\mathcal{N}|)^3)$ time. Similarly, it takes $O(|M| \cdot (|M| + |\mathcal{N}|)^3)$ time to place all service models to cloudlets. `Algorithm` 1 thus takes $O(|V| \cdot (|V| + |\mathcal{N}|)^3 + |M| \cdot (|M| + |\mathcal{N}|)^3)$ time for the initial placement problem of DTs and models. □

V. Online Algorithms for the Cumulative Fidelity Maximization Problem

In this section, assuming that both DTs and service models have already been placed into cloudlets by invoking `Algorithm` 1, we deal with the cumulative fidelity maximization problem over a finite time horizon $|\mathbb{T}|$ as follows.

The basic idea behind the proposed online algorithm proceeds as follows. We start by predicting the average volume of the update data of each object since its last uploading. We then choose some if not all mobile devices to upload their update data to their DTs due to the limited bandwidth on each AP. Furthermore, we choose some service models to retrain at the current time slot. We also determine whether some DTs and service models need to be migrated to new locations at the current time slot, by a migration control condition. That is, if the condition is met, the DT and service model migrations will take place at that time slot.

A. *Predicting the Average Volume of the Update Data of Each Mobile Device at Each Time Slot*

So far we assumed that the volume $vol(v_i, t)$ of the update data of each object $v_i \in V$ since its last uploading at time slot t is given, which in fact is unknown. Accurately predicting the volume of the update data of each object at each time slot is crucial, as the data volume determines not only the cumulative fidelity gain of all service models in which the object is their attribute but also the total cost of various resources consumed on uploading, transferring, and processing of the update data.

We here utilize the DT of each object to predict the average volume of its update data since its last uploading. We then choose some objects for uploading based on the predicted values due to bandwidth constraint on each AP, with the aim to maximize the optimization objective (17). Specifically, given a time slot $t \in \mathbb{T}$, the volume $vol(v_i, t)$ of the update data of object v_i at time slot t is predicted as follows.

Since the volume of the update data of each object v_i fluctuates dynamically over time, there are many prediction methods for the volume prediction, including the Long-Short-Term-Memory (LSTM) method, auto-regression method, etc. We here adopt the auto-regression method to predict the average volume $\widetilde{vol}(v_i, r)$ of the update data of object v_i since its last uploading (its $(r-1)$th uploading) at its rth uploading (assuming at time slot t), using its historical update data in the past $q \geq 1$ rounds uploading and the uploaded data are stored in DT_i [30].

$$\widetilde{vol}(v_i, r) = \lambda_1 \cdot vol(v_i, r-1) + \lambda_2 \cdot vol(v_i, r-2) + \dots \lambda_q \cdot vol(v_i, r-q) \tag{18}$$

where $\lambda_1, \dots, \lambda_q$ are non-negative constants with $\sum_{i=1}^{q} \lambda_i = 1$ and $\lambda_q \geq \lambda'_q$ when $q < q'$, as the update data generated recently is more important than the earlier one.

B. *Scheduling Mobile Devices for Uploading at Each Time Slot*

Having placed DTs and service model instances, and the given predicted volume $\widetilde{vol}(v_i, t)$ of the update data of object v_i at time slot t, we now determine which objects to upload their update data at time slot t so that the cumulative fidelity gain of all service models is maximized.

Recall that each object (mobile device) v_i is covered by a set $AP(v_i, t)$ of APs at time slot t. Denote by $\mathcal{M}(v_i)$ the set of service models whose source DT data comes from object v_i. Having deployed DT_i of object v_i to cloudlet $h(DT_i, t)$ and service model $m_k \in M$ to cloudlet $h(m_k, t)$, if object v_i uploads the volume $vol(v_i, t)$ of the update data via AP $j \in AP(v_i, t)$, then its contribution $obj\_contri(v_i, j, t)$ to the objective (17) is

$$\begin{aligned}
&obj\_contri(v_i, j, t) = \\
&\sum_{m_k \in \mathcal{M}(v_i)} \Delta G(m_k, v_i, t) - \omega \cdot (\xi_1 \cdot PX_i \cdot \tau_{up}(v_i, j, t) \\
&+ \sum_{e \in P_{j, h(DT_i, t)}} \xi_2 \cdot vol(v_i, t) \cdot length(e) \\
&+ \xi_3 \cdot vol(v_i, t) \cdot f^2_{h(DT_i, t)} \\
&-\omega \cdot \left(\sum_{m_k \in \mathcal{M}(v_i)} \left(\sum_{e \in P_{h(DT_i, t), h(m_k, t)}} \xi_2 \cdot vol'(v_i, t) \cdot length(e) \right) \right. \\
&\left. + \xi_3 \cdot vol'(v_i, t) \cdot f^2_{h(m_k, t)} \right),
\end{aligned} \tag{19}$$

where the first term in the RHS of (19) is a positive contribution to the cumulative fidelity gain $\Delta G(m_k, v_i, t)$ defined by (5) of all models by data uploading of object v_i. The rest terms are the uploading cost, the transmission cost of the update data from the uploading location of v_i to its DT location $h(DT_i, t)$, the processing cost of the update data at $h(DT_i, t)$, the transmission cost of the update data from its DT location to the locations of service models in $\mathcal{M}(v_i)$, and the training cost of each model m_k at cloudlet $h(m_k, t)$ with CPU frequency $f_{h(m_k, t)}$, respectively.

Due to the bandwidth constraint on each AP, we choose some objects to upload their update data to their DTs so that the sum $\sum_{v_i \in V} obj\_contri(v_i, j, t)$ is maximized at time slot t. We reduce this choosing objects to the maximum-profit Generalized Assignment Problem (GAP) as follows.

Recall each AP $j \in \mathcal{N}$ has L_j (>1) orthogonal sub-channels, there is a bin b_j with capacity L_j. For each object $v_i \in V$, if

Algorithm 2: Algorithm for Scheduling Objects to Upload Their Update Data via APs to Their DTs at Time Slot $t \in \mathbb{T}$.

```
Input: An MEC network $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, a set $\mathcal{N}$ of cloudlets
  (APs), a set $V$ of mobile devices (objects) with their DT
  placements, and a set $M$ of placed service models at time
  slot $t$.
Output: Schedule some objects via APs to upload their
  update data to their DTs at time slot $t$ such that the
  objective (17) is maximized.
1: $S_t \leftarrow \emptyset$; /* the solution of object scheduling at time slot
     $t$ */
2: for each $v_i \in V$ do
3:   for each AP $j \in \mathcal{N}$ do
4:     if AP $j \in AP(v_i, t)$ then
5:       Calculate $obj\_contri(v_i, j, t)$ by (19);
6:     else
7:       $obj\_contri(v_i, j, t) \leftarrow 0$;
8:     end if
9:   end for
10: end for;
11: Construct an instance of the maximum-profit GAP,
     where each AP $j \in \mathcal{N}$ corresponds to a bin $b_j$ with
     capacity $L_j$. Each object $v_i \in V$ is an item $item_i$ with
     weight 1, i.e., if object $v_i$ is assigned to one sub-channel
     of AP $j \in AP(v_i, t)$ and $obj\_contri(v_i, j, t) > 0$, then
     the profit brought is $obj\_contri(v_i, j, t)$; otherwise the
     profit is 0;
12: Find an approximate solution $S_t$ to the maximum-profit
     GAP, by applying the approximation algorithm in [2];
13: return An approximate solution $S_t$ is obtained.
```

it is covered by AP j at time slot t (i.e., $j \in AP(v_i, t)$) and $obj\_contri(v_i, j, t) > 0$, there is an item $item_i$ with weight 1, i.e., object v_i occupies one of the L_j sub-channels of AP j if it uploads its update data via AP j, and the profit obtained is $obj\_contri(v_i, j, t)$ by (19). Otherwise, the profit of assigning $item_i$ to bin b_j is 0, i.e., object v_i cannot upload its update data via AP $j \in \mathcal{N} \backslash AP(v_i, t)$ as either it is not under the coverage of AP j or its contribution to the optimization objective will be zero. There is a constant approximation algorithm for the maximum-profit GAP [2]. The proposed algorithm for choosing mobile devices to upload their update data via APs at time slot t is then given in Algorithm 2.

C. Online Algorithm

Given a finite time horizon, to maximize the optimization objective value (17) at each time slot, some DTs and/or models may be triggered to migrate. However, each such migration will incur an overhead (or a service cost). Especially if DT_i of object v_i is migrated to a new location at time slot t, then its migration cost $cost_{mig}(DT_i, t)$ will be determined by the accumulative size of its update data stored at DT_i so far and the length of its migration path. It can be seen that frequent migrations of a DT are likely to result in the high accumulative migration cost of the DT. Thus, it needs to develop an effective migration strategy to determine whether DT and/or model migrations take place at each time slot. In the following, we discuss the conditions under which DT and model migrations will occur.

Let $mig\_cost(t)$ be the total cost of some DT and model migrations at time slot t, which is defined as follows.

$$mig\_cost(t) = \sum_{v_i \in V} cost_{mig}(v_i, t) + \sum_{m_k \in M} cost_{mig}(m_k, t),$$

where the migration costs of a DT and a model at time slot t are defined in (12) and (13), respectively.

Let $nomig\_cost(t)$ be the total cost excluding the migration costs at time slot t, which is defined as follows.

$$nomig\_cost(t) = \sum_{v_i \in V} cost_{DT}(v_i, t) + \sum_{m_k \in M} cost_{model}(m_k, t) - mig\_cost(t), \quad (20)$$

where $cost_{DT}(v_i, t)$ is the updating cost of DT_i by (15), and $cost_{model}(m_k, t)$ is the model updating cost of m_k by (16). The value of the optimization objective function of the problem at time slot t then is $\sum_{m_k \in M} \Delta F(m_k, t) - \omega \cdot (mig\_cost(t) + nomig\_cost(t))$, by (17).

Given a time slot t, we determine whether some DT and model migrations take place or not. To this end, we first find a potential alternative solution of migrations, that is, which DTs and which service models need to migrate and to where they migrate. This alternative solution is obtained through a simple modification on Algorithm 1 as follows.

Potential DT migrations: Recall that Algorithm 1 first places DTs of objects by constructing an auxiliary bipartite graph, where the weight of an edge between a DT_i and a cloudlet is the expected updating cost of placing DT_i in the cloudlet by (1). The expected DT updating cost is calculated based on the probabilities of object v_i visiting top-K locations, while the exact location j of object v_i at the current time slot is given. With the expected DT updating cost by (15), the weight $\widetilde{cost}_{DT}(v_i, j', t)$ of an edge between DT_i and a cloudlet j' in the auxiliary bipartite graph at time slot t is defined as follows.

$$\widetilde{cost}_{DT}(v_i, j', t) = cost_{mig}(v_i, t) + cost_{up}(v_i, j, t) + cost_{trans}(v_i, j, j', t) + cost_{proc}(v_i, t), \quad (21)$$

where $cost_{mig}(v_i, t)$ is the migration cost of DT_i by (12), and $cost_{mig}(v_i, t)$ is 0 when there is no migration of DT_i, i.e., $cost_{mig}(DT_i) = 0$ if $h(DT_i, t-1) = j'$. $cost_{up}(v_i, j, t)$, $cost_{trans}(v_i, j, j', t)$, and $cost_{proc}(v_i, t)$ are the costs defined by (7), (8) and (9), respectively.

Potential model migrations: Algorithm 1 places service models under the assumption that all DTs have been placed. We refer to the model updating cost in (16), and define $\widetilde{cost}_{model}(m_k, j', t)$ as the new weight of an edge between model m_k and cloudlet j' (the migrated location of m_k) in the auxiliary bipartite graph as follows.

$$\widetilde{cost}_{model}(m_k, j', t) = cost_{mig}(m_k, t) + cost_{agg}(m_k, j', t) + cost_{train}(m_k, t), \quad (22)$$

where the migration cost $cost_{mig}(m_k, t)$ in (13) is determined by whether model migrations take place at time slot t or not, and $cost_{agg}(m_k, j', t)$ and $cost_{train}(m_k, t)$ are the costs defined by (10) and (11), respectively.

As a result, a potential alternative solution $\mathcal{S}_{mig}(t)$ of DT and model migrations is obtained, by invoking the modified Algorithm 1 with the modified edge weights $\widetilde{cost}_{DT}(v_i, j', t)$ and $\widetilde{cost}_{model}(m_k, j', t)$ in their corresponding auxiliary bipartite graphs at time slot t. Based on the potential alternative solution $\mathcal{S}_{mig}(t)$, which mobile devices can upload their update data at time slot t is determined by invoking Algorithm 2. An alternative solution $\mathbb{S}_1$ for the cumulative fidelity maximization problem at time slot t is then obtained.

Let $\hat{t}$ be the last time slot of migrations of DTs and models with $\hat{t} = 0$ initially. Given the current time slot t with $t > \hat{t}$, let β be a tunable parameter with $\beta > 1$, to determine whether to adopt the alternative partial solution $\mathcal{S}_{mig}(t)$ at time slot t, we apply the following migration control condition.

$$\omega \cdot mig\_cost(t) \le \frac{1}{\beta} \cdot \sum_{t'=\hat{t}+1}^{|\mathbb{T}|} \left(\sum_{m_k \in M} \Delta F(m_k, t') - \omega \cdot nomig\_cost(t') \right), \tag{23}$$

where the tuning parameter β is used to bound the total migration cost at time slot t with $t > \hat{t} \ge 0$. Equation (23) says if the total migration cost at time slot t is no greater than $\frac{1}{\beta}$ of the sum of the objective function values without DT and model migrations between time slots $\hat{t}+1$ and t, the migration operation will proceed at time slot t, and the solution $\mathbb{S}_1$ for the cumulative fidelity maximization problem at time slot t is obtained; otherwise (no migrations occur at time slot t), all DTs and models are kept at their locations at time slot $t-1$, and a solution $\mathbb{S}_2$ to the cumulative fidelity maximization problem at time slot t is obtained, by invoking Algorithm 2.

The online algorithm for the cumulative fidelity maximization problem is given in Algorithm 3.

Algorithm 3: Algorithm for the Cumulative Fidelity Maximization Problem.

```
Input: An MEC network $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, a set $\mathcal{N}$ of cloudlets
  (APs), a given time horizon $\mathbb{T}$, a set $V$ of mobile objects
  with their DT placements, and a set $M$ of placed service
  models.
Output: Schedule objects via APs to upload their update
  data and schedule models for retraining using the update
  data of objects, and migrate some DTs and models to other
  locations at each time slot $t \in \mathbb{T}$ if needed such that the
  problem optimization objective (17) over time horizon $\mathbb{T}$ is
  maximized.
 1: $\mathbb{S} \leftarrow \emptyset$; /* the problem solution */
 2: for each time slot $t \in \mathbb{T}$ do
 3:    Predict the volume $vol(v_i, t)$ of the update data
       generated by object $v_i$ at time slot $t$, by applying the
       auto-regression prediction mechanism;
 4:    Find a potential migration solution $\mathcal{S}_{mig}(t)$ for DT and
       model migrations, by invoking an algorithm similar to
       Algorithm 1 with the modified edge weights
       $\widetilde{cost}_{DT}(v_i, j', t)$ and $\widetilde{cost}_{model}(m_k, j', t)$ in the
       corresponding auxiliary bipartite graphs;
 5:    Calculate the utility gain $\Delta F(m_k, t)$ of each model
       $m_k$, the migration cost $mig\_cost(t)$, and the
       non-migration cost $nomig\_cost(t)$ based on the
       potential migration solution at time slot $t$;
 6:    if Ineq (23) holds then
 7:       $\hat{t} \leftarrow t$; /* $\hat{t}$ is the last time slot in which DT and
          model migrations took place */
 8:       Perform DT and model migrations by adopting the
          potential migration solution;
 9:       Find a potential alternative solution $\mathbb{S}_1$ for the
          problem at time slot $t$, by invoking Algorithm 2;
10:       $\mathbb{S} \leftarrow \mathbb{S} \cup \{\langle \mathbb{S}_1, t \rangle\}$;
11:    else
12:       Find a solution $\mathbb{S}_2$ for the problem at time slot $t$, by
          invoking Algorithm 2, assuming that neither DTs
          nor models change their locations at time slot $t$;
13:       $\mathbb{S} \leftarrow \mathbb{S} \cup \{\langle \mathbb{S}_2, t \rangle\}$
14:    end if
15: end for;
16: return Solution $\mathbb{S}$.
```

D. Algorithm Analysis

Lemma 1: Given an MEC network $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, a set $\mathcal{N}$ of cloudlets (APs), a set of $\mathcal{E}$ of links between APs, a set V of mobile objects with their placed DTs, a set M of placed service models that are continuously retrained using the update data of objects in V, and a given time slot $t \in \mathbb{T}$, there is an approximation algorithm, Algorithm 2 for scheduling objects to upload their update data at time slot t, delivering a $\frac{1}{2+\epsilon}$-approximate solution $\mathbb{S}_t$. The algorithm takes $O(|\mathcal{N}| \cdot |V| \cdot \log \frac{1}{\epsilon} + \frac{|\mathcal{N}|}{\epsilon^4})$ time, where ϵ is constant with $0 < \epsilon < 1$.

Proof: Given the solution by Algorithm 2 at each time slot t, some objects in it are allocated with sub-channels via APs to upload their update data, and there is not any bandwidth capacity violation on any AP in terms of sub-channel allocation at time slot t. Thus, assigning objects under the coverage of an AP with its different sub-channels for their update data uploading is feasible. Meanwhile, assigning a subset of service requests in $U(t)$ at time slot t to service model instances in different cloudlets for processing is reduced to the maximum-profit GAP, and there is an approximation algorithm for the maximum-profit GAP, delivering an approximate solution with no less than $\frac{1}{2+\epsilon}$ times the optimal one [2]. This request assignment is feasible too, because there is no computing capacity violation on any cloudlet. Thus, the solution delivered by Algorithm 3 is feasible.

The time complexity of Algorithm 2 is analyzed as follows. The approximation algorithm in [2] takes $(|\mathcal{N}| \cdot |V| \cdot \log \frac{1}{\epsilon} + \frac{|\mathcal{N}|}{\epsilon^4})$ time per time slot. Therefore, the time complexity of Algorithm 2 is $O(|\mathcal{N}| \cdot |V| \cdot \log \frac{1}{\epsilon} + \frac{|\mathcal{N}|}{\epsilon^4})$ per time slot. □

Lemma 2: The total migration cost in the solution delivered by `Algorithm` 3 is no greater than $\frac{1}{\beta}$ times the sum of the objective function values over time horizon $\mathbb{T}$ without DT and service model migrations, i.e., $\sum_{t=1}^{|\mathbb{T}|} mig\_cost(t) \leq \frac{1}{\beta} \cdot \sum_{t=1}^{|\mathbb{T}|}(\sum_{m_k \in M} \Delta F(m_k, t) - nomig\_cost(t))$, where $\beta > 1$.

Proof: Let $\hat{t}_r$ be the time slot of the rth round of migrations of DTs and models with $\hat{t}_0 = 0$ and $mig\_cost(\hat{t}_0) = 0$ initially. Let $\hat{t}_R \leq |\mathbb{T}|$ be the last time slot in which DT and/or service model migrations occur. By the given migration control condition (23), we have

$$\begin{aligned}\omega \cdot \sum_{t=1}^{|\mathbb{T}|} mig\_cost(t) &= \omega \cdot \sum_{r=1}^{R} mig\_cost(\hat{t}_r)\\ &\leq \frac{1}{\beta} \cdot \sum_{t=1}^{\hat{t}_R} \left(\sum_{m_k \in M} \Delta F(m_k, t) - \omega \cdot nomig\_cost(t) \right)\\ &\leq \frac{1}{\beta} \cdot \sum_{t=1}^{|\mathbb{T}|} \left(\sum_{m_k \in M} \Delta F(m_k, t) - \omega \cdot nomig\_cost(t) \right) \end{aligned} \tag{24}$$

Notice that (24) holds only if the assumption that $\sum_{m_k \in M} \Delta F(m_k, t) - \omega \cdot nomig\_cost(t) \geq 0$ holds for all $t \in \mathbb{T}$. □

Lemma 3: Denote by OPT the optimal solution of the cumulative fidelity maximization problem. Then, the value of the OPT is no greater than σ times the accumulative objective value over time horizon $\mathbb{T}$ without migrations of DTs and models, i.e.,

$$OPT \leq \sigma \cdot \sum_{t=1}^{|\mathbb{T}|} \left(\sum_{m_k \in M} \Delta F(m_k, t) - \omega \cdot nomig\_cost(t) \right),$$

where $\sigma = \frac{\max\{\sum_{m_k \in M} \Delta F(m_k,t) - \omega \cdot nomig\_cost(t) \mid t \in \mathbb{T}\}}{\min\{\sum_{m_k \in M} \Delta F(m_k,t) - \omega \cdot nomig\_cost(t) \mid t \in \mathbb{T}\}}$.

Proof: Let $\Delta F^*(m_k, t)$, $mig\_cost^*(t)$ and $nomig\_cost^*(t)$ be the fidelity gain of model m_k, the migration cost and non-migration cost in the optimal solution OPT at time slot t, respectively. Following the definition of σ, we have

$$\begin{aligned}&\sum_{m_k \in M} \Delta F^*(m_k, t) - \omega \cdot nomig\_cost^*(t)\\ &\quad \leq \sigma \cdot \left(\sum_{m_k \in M} \Delta F(m_k, t) - \omega \cdot nomig\_cost(t) \right)\end{aligned} \tag{25}$$

Then,

$$\begin{aligned}&\sum_{t=1}^{|\mathbb{T}|} \left(\sum_{m_k \in M} \Delta F^*(m_k, t) - \omega \cdot nomig\_cost^*(t) \right)\\ &\quad \leq \sigma \cdot \sum_{t=1}^{|\mathbb{T}|} \left(\sum_{m_k \in M} \Delta F(m_k, t) - \omega \cdot nomig\_cost(t) \right). \end{aligned} \tag{26}$$

We have

$$\begin{aligned}OPT &= \sum_{t=1}^{|\mathbb{T}|} \sum_{m_k \in M} \Delta F^*(m_k, t)\\ &\quad - \omega \cdot \sum_{t=1}^{|\mathbb{T}|} (nomig\_cost^*(t) + mig\_cost^*(t))\\ &\leq \sum_{t=1}^{|\mathbb{T}|} \left(\sum_{m_k \in M} \Delta F^*(m_k, t) - \omega \cdot nomig\_cost^*(t) \right)\\ &\leq \sigma \cdot \sum_{t=1}^{|\mathbb{T}|} \left(\sum_{m_k \in M} \Delta F(m_k, t) - \omega \cdot nomig\_cost(t) \right) \end{aligned} \tag{27}$$

Equation (27) holds by (26). □

Theorem 3: Given an MEC network $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, a given time horizon $\mathbb{T}$, a set V of mobile objects with their DT placements, and a set M of placed service models, there is an online algorithm, `Algorithm` 3, with a competitive ratio of $\frac{1}{\sigma}(1 - \frac{1}{\beta})$, for the cumulative fidelity maximization problem in $\mathcal{G}$, which takes $O(|V| \cdot (|V| + |\mathcal{N}|)^3 + |M| \cdot (|M| + |\mathcal{N}|)^3 + |\mathcal{N}| \cdot |V| \cdot \log \frac{1}{\epsilon} + \frac{|\mathcal{N}|}{\epsilon^4})$ time per time slot, where $\sigma > 1$ is defined in Lemma 3, $\beta > 1$ is a control parameter, and ϵ is constant with $0 < \epsilon \leq 1$.

Proof: Let OPT and S be the values of the optimal solution and the solution delivered by `Algorithm` 3, respectively. Then

$$\begin{aligned}S &= \sum_{t=1}^{|\mathbb{T}|} \left(\sum_{m_k \in M} \Delta F(m_k, t) - \omega \cdot (nomig\_cost(t) + mig\_cost(t)) \right)\\ &\geq \sum_{t=1}^{|\mathbb{T}|} \sum_{m_k \in M} \Delta F(m_k, t) - \omega \cdot \sum_{t=1}^{|\mathbb{T}|} nomig\_cost(t)\\ &\quad - \frac{1}{\beta} \cdot \sum_{t=1}^{|\mathbb{T}|} \left(\sum_{m_k \in M} \Delta F(m_k, t) - \omega \cdot nomig\_cost(t) \right) \end{aligned} \tag{28}$$

$$\begin{aligned}&\geq \left(1 - \frac{1}{\beta}\right) \cdot \sum_{t=1}^{|\mathbb{T}|} \left(\sum_{m_k \in M} \Delta F(m_k, t) - \omega \cdot nomig\_cost(t) \right)\\ &\geq \frac{1}{\sigma} \cdot \left(1 - \frac{1}{\beta}\right) \cdot OPT \end{aligned} \tag{29}$$

Equation (28) holds by Lemma 2, and (29) holds by Lemma 3.

The time complexity of `Algorithm` 3 is analyzed as follows. The running times of `Algorithm` 1 and the approximation algorithm in [2] are the dominant ones, where `Algorithm` 1 takes $O(|V| \cdot (|V| + |\mathcal{N}|)^3 + |M| \cdot (|M| + |\mathcal{N}|)^3)$ time per time slot by Theorem 2, and the approximation algorithm in [2] takes $(|\mathcal{N}| \cdot |V| \cdot \log \frac{1}{\epsilon} + \frac{|\mathcal{N}|}{\epsilon^4})$ time per time slot. Thus, `Algorithm` 3 takes $O(|V| \cdot (|V| + |\mathcal{N}|)^3 + |M| \cdot (|M| + |\mathcal{N}|)^3 + |\mathcal{N}| \cdot |V| \cdot \log \frac{1}{\epsilon} + \frac{|\mathcal{N}|}{\epsilon^4})$ time per time slot. □

VI. Performance Evaluation

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

A. Experiment Settings

We considered MEC network instances generated by the GT-ITM tool [3], where the number of APs (and their co-located cloudlets) is drawn from 50 to 250. The computing capacity on each cloudlet is drawn from 4,000 MHz to 8,000 MHz randomly [14]. The amount of computing resource demanded by a DT or a service model is within $[20, 100]$ MHz [10]. The bandwidth capacity on an AP ranges from 5 MHz to 20 MHz, and the number of orthogonal sub-channels of an AP ranges from 3 to 6 by adopting the OFDMA scheme [6]. There are 2,000 mobile devices (objects), and the transmission power of each mobile device is in the range of $[0.1, 0.5]$ Watt [14]. The cost ξ_1 of a unit power is within $[0.01, 0.1]$, and the noise power is set as 1×10^{-10} Watt [14]. Following [23], the channel gain between a mobile device v_i and an AP j is set as $dist_{i,j}^{-\alpha}$, where $dist_{i,j}$ is the euclidean distance between their locations, and α is the path loss coefficient (we set $\alpha = 4$). The volume of the update data of each mobile device is within $[1, 5]$ MB [29], and the volume of the processed update data is half of that of its raw data. There are 500 service models, and the number of attributes in each model ranges from 5 to 15, respectively. The cost ξ_2 of transferring a unit data along a link is set within $[0.01, 0.1]$ [22]. The coefficient ξ_3 is set within $[0.001, 0.01]$ per unit energy consumption. The instantiation cost of a DT instance or a model instance in a cloudlet is proportional to the amount of computing resource consumed, i.e., it costs 0.01 per MHz. The cost of processing unit data by a DT or a model in its hosting cloudlet is set within $[0.01, 0.1]$. The volume of a service model is set within $[2, 10]$ MB. Referring to the definition of fidelity gain of service models by (5) and (6), we adopted a submodular function $f(x) = \log_{1.1}(\frac{x}{10} + 1)$ in (5), and assigned the lth attribute of model m_k with weight $w_{k,l} = \frac{1}{l_k}$, where l_k is the number of attributes in model m_k.

The time horizon consists of 20 time slots. The maximum number of potential movement locations of each mobile device is 10% of the number of APs in the network [21]. We adopted the auto-regression method [30] to predict the volume $vol(v_i, t)$ of the update data of mobile device v_i at time slot t, with $vol(v_i, t) = 0.4 \cdot vol(v_i, t-1) + 0.3 \cdot vol(v_i, t-2) + 0.2 \cdot vol(v_i, t-3) + 0.1 \cdot vol(v_i, t-4)$. We set ω at 0.1, ϵ at 0.5, and β is set at 4, respectively.

To evaluate the performance of the proposed `Algorithm 1` (i.e., `Alg.1`) for the initial DT and model placement problem, we introduced the following two benchmarks: `Heu.1_initial` first deploys each DT in a cloudlet with the least expected DT updating cost by (1) greedily. It then deploys each service model in a cloudlet with the least expected model updating cost by (2) greedily too. `Heu.2_initial` first considers cloudlets one by one for deploying DTs, i.e., each cloudlet is iteratively assigned with a DT, which can achieve the least expected DT updating cost, this process continues until the cloudlet cannot accommodate any more DTs. It then considers cloudlets one by one for model deployments. Each cloudlet is iteratively assigned with a model, which can achieve the least expected model updating cost.

To study the performance of the proposed `Algorithm 3`, referred to as `Alg.3`, for the cumulative fidelity maximization problem, we proposed the following two benchmarks: `Heu.1_on`: It first invokes algorithm `Heu.1_initial` for the initial DT and model placements. It then determines whether to perform migrations of DTs and models at each time slot $t \in \mathbb{T}$. Specifically, at each time slot t, it has a potential scheduling that deploys DT_i of each object $v_i \in V$ to a cloudlet j' with the least $\widetilde{cost}_{DT}(v_i, j', t)$ by (21) and each service model m_k to a cloudlet j' with the least $\widetilde{cost}_{model}(m_k, j', t)$ by (22), respectively. Within each time slot, `Heu.1_on` considers mobile devices one by one, and predicts the update data size at the current time slot, through adopting that in the previous time slot (`Heu.1_on` predicts the update data size at the 1st time slot, through adopting the expected size of the update data). `Heu.1_on` assigns each mobile device to an AP with the maximum $obj\_contri(v_i, j, t)$ by (17). Another benchmark `Heu.2_on` proceeds as follows. It invokes `Heu.2_initial` for the initial DT and model placements. It then has a potential scheduling for DT and model migrations at each time slot t. That is, it considers cloudlets one by one for DT deployments, i.e., each cloudlet is assigned with a DT with the least $\widetilde{cost}_{DT}(v_i, j', t), t)$. This procedure continues until the cloudlet cannot host any more DTs. It then deploys models to each cloudlet one by one similarly. Each cloudlet is assigned a model with the least $\widetilde{cost}_{model}(m_k, j', t)$, and this procedure continues until the cloudlet cannot host any models further. `Heu.2_on` adopts the same prediction method as the one for `Heu.1_on`. For scheduling the data uploading of mobile devices at each time slot, it considers APs one by one at each time slot, i.e., each AP is iteratively assigned with a mobile device which can achieve the largest $obj\_contri(v_i, j, t)$ until the AP cannot accommodate any more mobile devices. This procedure continues until all APs have been examined.

The value in each figure is the mean of 30 different network instances of the same size. The running time of each algorithm is obtained by a desktop with an Octa-Core Intel(t) Xeon(t) CPU @ 2.20 GHz, 32 G RAM. Unless otherwise specified, we adopt the above parameters in the default setting.

B. Performance of Different Algorithms for the Initial DT and Model Placement Problem

We first evaluated the performance of `Alg.1` against `Heu.1_initial` and `Heu.2_initial` for the initial DT and model placement problem, by varying the network size from 50 to 250. Fig. 2 depicts the performance and running time of different algorithms. Fig. 3(a) shows that when the network size is 250, the total expected cost defined in (14) delivered by `Alg.1` is the lowest among comparison algorithms, which is 77.1% and 69.8% of those by `Heu.1_initial` and `Heu.2_initial`, respectively. Fig. 2(b) shows that the running time of `Alg.1` is the longest one due to the constructions of bipartite auxiliary graphs.

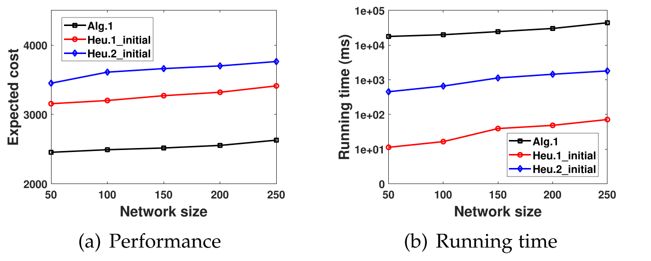


Fig. 2. Performance of different algorithms for the initial DT and model placement problem.

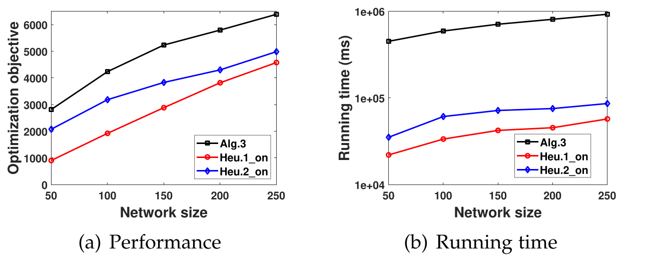


Fig. 3. Performance of different algorithms for the cumulative fidelity maximization problem.

C. Performance of Different Algorithms for the Cumulative Fidelity Maximization Problem

We then studied the performance of Alg.3 against Heu.1_*on* and Heu.2_*on* for the cumulative fidelity maximization problem, by varying the network size from 50 to 250. Fig. 3 plots the performance and running time of the three comparison algorithms. It can be observed from Fig. 3(a) that Alg.3 achieves the best performance among the algorithms, outperforming Heu.1_on and Heu.2_on by 39.5 % and 28.1% respectively, when the network size is 250. The rationale behind this is that Alg.3 adopts an effective mechanism to predict the update data size of each IoT device and then determines which APs to upload their update data at each time slot. Meanwhile, it is observed from Fig. 3(b) that the running time of Alg.3 is the longest.

D. Impacts of Parameters on the Performance of Algorithm 3

We finally evaluated the impacts of important parameters $|V|$, $|M|$, $|\mathcal{N}|$, T, L and β on the performance of Alg.3 for the cumulative fidelity maximization problem as follows.

We studied the impact of the number $|V|$ of mobile devices on the performance of Alg.3, by varying its range from 1,000 to 5,000. It can be seen from Fig. 4(a) that the performance of Alg.3 when $|V| = 3,000$ is 64.8% of itself when $|V| = 1,000$, assuming that the network size is set at 250. This indicates that DTs of mobile devices can be updated more frequently when less numbers of mobile devices are deployed in the MEC network. Fig. 4(b) shows Alg.3 takes the longest running time when $|V| = 3,000$.

We dealt with the impact of the number $|M|$ of the service models on the performance of Alg.3 when $|M| = 300, 400, 500, 600,$ and 700. As evidenced by Fig. 5(a), the performance of Alg.3 with 300 models is 39.7% of that by itself

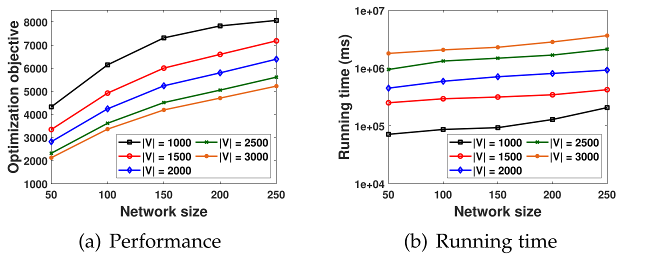


Fig. 4. Impact of the number $|V|$ of objects on the performance of Alg.3.

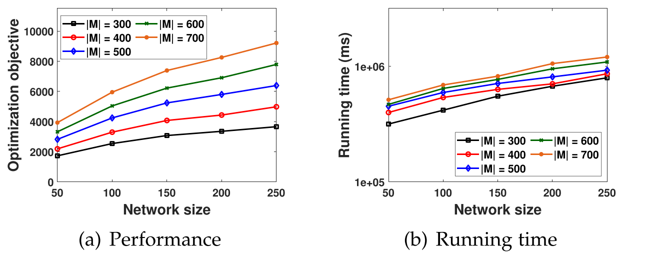


Fig. 5. Impact of the number $|M|$ of service models on the performance of Alg.3.

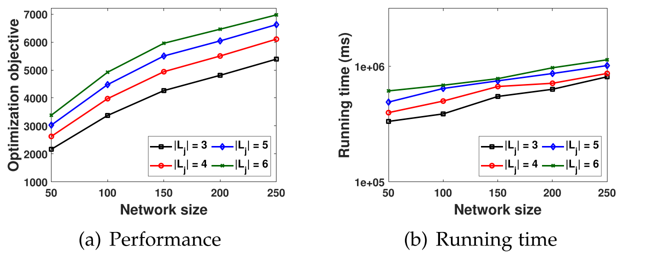


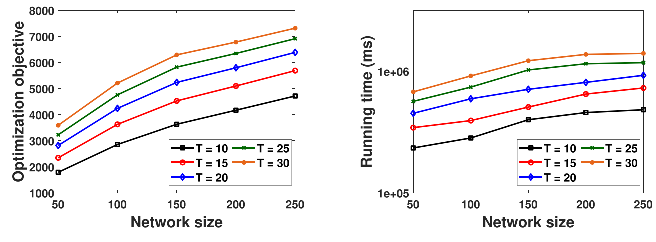
Fig. 6. Impact of the number L_j of sub-channels of APs on the performance of Alg.3.

with 700 models when the network size is 250. This is due to that larger fidelity gain of service models can be obtained with more service models available, and Alg.3 takes a longer time as well, as shown in Fig. 5(b).

We evaluated the impact of the number L_j of sub-channels of each AP $j \in \mathcal{N}$ on the performance of Alg.3, by varying the value of L_j from 3 to 6. Fig. 6(a) shows that the performance of Alg.3 with $L_j = 3$ is 78.3% of itself with $L_j = 6$ when the network size is set at 250. This is because the data of more devices can be uploaded to enhance the model fidelity with the increase on more sub-channels of APs.

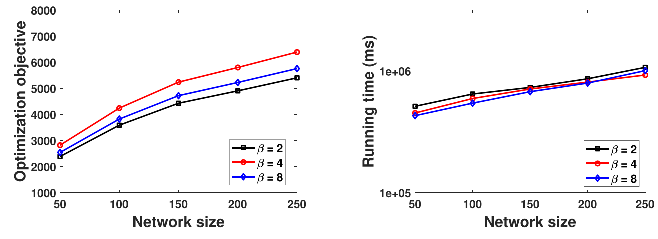
We investigated the impact of the number T of time slots on the performance of Alg.3, by varying the number of time slots from 10 to 30. Fig. 7(a) depicts that the performance of Alg.3 with 30 time slots is 55.1% higher than that of itself with 10 time slots when the network size is set at 250. This is due to the fact that the fidelity of a service model is measured by a submodular non-decreasing function, the net performance improvement of Alg. 3 at a time slot decreases with the increase on the number of time slots.

We considered the impact of parameters β on the performance of Alg.3, by varying its value from 2 to 8. It can be seen from Fig. 8(a) that when the network size is 250, the performance of Alg.3 with $\beta = 4$ is 10.9% and 18.3% higher than that by itself



(a) Performance (b) Running time

Fig. 7. Impact of the number T of time slots on the performance of `Alg.3`.



(a) Performance (b) Running time

Fig. 8. Impact of parameter β on the performance of `Alg.3`.

with $\beta = 2$ and $\beta = 8$, respectively. The rationale is that a larger β implies that `Alg.3` intends to avoid frequent migrations of DTs and models. Fig. 8(b) shows that the impact of β on the running time of `Alg.3` is negligible.

VII. Conclusion

In this paper, we investigated fidelity-aware inference service provisioning in a DT-assisted edge computing network through jointly choosing mobile objects for uploading and service models for retraining using the update data of mobile devices to enhance service fidelity of models. We formulated two closely related optimization problems: the initial placement problem of DTs and service models, assuming that mobility profiles of mobile devices (objects) are given; and the cumulative fidelity maximization problem over a given time horizon. We developed an efficient algorithm for the former by reducing to the minimum-cost maximum matching problem in a series of auxiliary bipartite graphs. Meanwhile, we devised an online algorithm with a provable competitive ratio for the latter through scheduling which mobile devices for uploading, and which DTs and service models for migrations at each time slot. We finally evaluated the performance of the proposed algorithms by simulations. Simulation results show that the proposed algorithms are promising, outperforming their comparison counterparts by no less than 28%.

Acknowledgment

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

References

[1] H. Chen, T. D. Todd, D. Zhao, and G. Karakostas, “Digital twin model selection for feature accuracy,” *IEEE Internet Things J.*, vol. 11, no. 7, pp. 11415–11426, Apr. 2024.
[2] R. Cohen, L. Katzir, and D. Raz, “An efficient approximation for the generalized assignment problem,” *Inf. Process. Lett.*, vol. 100, pp. 162–166, 2006.
[3] GT-ITM, 2019. [Online]. Available: http://www.cc.gatech.edu/projects/gtitm/
[4] Q. Guo, F. Tang, and N. Kato, “Resource allocation for aerial assisted digital twin edge mobile network,” *IEEE J. Sel. Areas Commun.*, vol. 41, no. 10, pp. 3070–3079, Oct. 2023.
[5] Y. Hui et al., “Collaboration as a service: Digital-twin-enabled collaborative and distributed autonomous driving,” *IEEE Internet Things J.*, vol. 9, no. 19, pp. 18607–18619, Oct. 2022.
[6] J. Kim, T. Kim, M. Hashemi, C. G. Brinton, and D. J. Love, “Joint optimization of signal design and resource allocation in wireless D2D edge computing,” in *Proc. IEEE Conf. Comput. Commun.*, 2020, pp. 2086–2095.
[7] Z. Kuang, Y. Shi, S. Guo, J. Dan, and B. Xiao, “Multi-user offloading game strategy in OFDMA mobile cloud computing system,” *IEEE Trans. Veh. Technol*, vol. 68, no. 12, pp. 12190–12201, Dec. 2019.
[8] J. Li et al., “Digital twin-assisted, SFC-enabled service provisioning in mobile edge computing,” *IEEE Trans. Mobile Comput.*, vol. 23, no. 1, pp. 393–408, Jan. 2024.
[9] J. Li et al., “Mobility-aware utility maximization in digital twin-enabled serverless edge computing,” *IEEE Trans. Comput.*, vol. 73, no. 7, pp. 1837–1851, Jul. 2024.
[10] 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.
[11] 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.
[12] J. Li et al., “AoI-aware service provisioning in edge computing for digital twin network slicing requests,” *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 14607–14621, Dec. 2024.
[13] J. Li et al., “AoI-aware, digital twin-empowered IoT query services in mobile edge computing,” *IEEE/ACM Trans. Netw.*, vol. 32, no. 4, pp. 3636–3650, Aug. 2024.
[14] 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.
[15] J. Li, W. Liang, J. Wang, and X. Jia, “Accumulative fidelity maximization of inference services in DT-assisted edge computing,” in *Proc. 1st IEEE Int. Conf. Meta Comput.*, 2024, pp. 64–73.
[16] J. Li, J. Wang, Q. Chen, Y. Li, and A. Y. Zomaya, “Digital twin-enabled service satisfaction enhancement in edge computing,” in *Proc. IEEE Conf. Comput. Commun.*, 2023, pp. 1–10.
[17] 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.
[18] X. Liang, W. Liang, Z. Xu, Y. Zhang, and X. Jia, “Multiple service model refreshments in digital twin-empowered edge computing,” *IEEE Trans. Serv. Comput.*, vol. 17, no. 5, pp. 2672–2686, Sep./Oct. 2024.
[19] H. Liao, Y. Shu, J. Lu, Z. Zhou, M. Tariq, and S. Mumtaz, “Integration of 6G signal processing, communication, and computing based on information timeliness-aware digital twin,” *IEEE J. Sel. Topics Signal Process.*, vol. 18, no. 1, pp. 98–108, Jan. 2024.
[20] X. Lin, J. Wu, J. Li, W. Yang, and M. Guizani, “Stochastic digital-twin service demand with edge response: An incentive-based congestion control approach,” *IEEE Trans. Mobile Comput.*, vol. 22, no. 4, pp. 2402–2416, Apr. 2023.
[21] Y. Ma, W. Liang, J. Li, X. Jia, and S. Guo, “Mobility-aware and delay-sensitive service provisioning in mobile edge-cloud networks,” *IEEE Trans. Mobile Comput.*, vol. 21, no, 1, pp. 196–210, Jan. 2022.
[22] Y. Ma, W. Liang, J. Wu, and Z. Xu, “Throughput maximization of NFV-enabled multicasting in mobile edge cloud networks,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 2, pp. 393–407, Feb. 2020.
[23] S. Nath, Y. Li, J. Wu, and P. Fan, “Multi-user multi-channel computation offloading and resource allocation for mobile edge computing,” in *Proc. IEEE Int. Conf. Commun.*, 2020, pp. 1–6.

[24] R. M. Nauss, “Solving the generalized assignment problem: An optimizing and heuristic approach,” *Informs J. Comput.*, vol. 15, no. 3, pp. 249–266, 2003.

[25] S. D. Okegbile, J. Cai, H. Zheng, J. Chen, and C. Yi, “Differentially private federated multi-task learning framework for enhancing human-to-virtual connectivity in human digital twin,” *IEEE J. Sel. Areas Commun.*, vol. 41, no. 11, pp. 3533–3547, Nov. 2023.

[26] Y. Qiu et al., “Reliable or green? Continual individualized inference provisioning in fabric metaverse via multi-exit acceleration,” *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 11449–11465, Dec. 2024.

[27] Y. Ren, S. Guo, B. Cao, and X. Qiu, “End-to-end network SLA quality assurance for C-RAN: A closed-loop management method based on digital twin network,” *IEEE Trans. Mobile Comput.*, vol. 23, no. 5, pp. 4405–4422, May 2024.

[28] Y. Shu, Z. Wang, H. Liao, Z. Zhou, N. Nasser, and M. Imran, “Age-of-information-aware digital twin assisted resource management for distributed energy scheduling,” in *Proc. IEEE Glob. Commun. Conf.*, 2022, pp. 5705–5710.

[29] M. Vaezi, K. Noroozi, T. D. Todd, D. Zhao, and G. Karakostas, “Digital twin placement for minimum application request delay with data age targets,” *IEEE Internet Things J.*, vol. 10, no. 13, pp. 11547–11557, Jul. 2023.

[30] Z. Xu, W. Liang, M. Jia, M. Huang, and G. Mao, “Task offloading with network function requirements in a mobile edge-cloud network,” *IEEE Trans. Mobile Comput.*, vol. 18, no. 11, pp. 2672–2685, Nov. 2019.

[31] Y. Yang et al., “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.

[32] R. Zhang, Z. Xie, D. Yu, W. Liang, and X. Chang, “Digital twin-assisted federated learning service provisioning over mobile edge networks,” *IEEE Trans. Comput.*, vol. 73, no. 2, pp. 586–598, Feb. 2024.

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

[34] 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.

[35] 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.

[36] 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.

[37] Y. Zhang, L. Wang, and W. Liang, “Deep reinforcement learning for mobility-aware digital twin migrations in edge computing,” *IEEE Trans. Serv. Comput.*, vol. 18, no. 2, pp. 704–717, Mar./Apr. 2025.

[38] F. Zhao, Y. Chen, Y. Zhang, Z. Liu, and X. Chen, “Dynamic offloading and resource scheduling for mobile-edge computing with energy harvesting devices,” *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 2, pp. 2154–2165, Jun. 2021.

[39] L. Zhao, Z. Bi, A. Hawbani, K. Yu, Y. Zhang, and M. Guizani, “ELITE: An intelligent digital twin-based hierarchical routing scheme for softwarized vehicular networks,” *IEEE Trans. Mobile Comput.*, vol. 22, no. 9, pp. 5231–5247, Sep. 2023.

Jing Li received the BSc (first class honours) and PhD degrees from Australian National University, in 2018 and 2022, respectively. He is currently a postdoctoral fellow with the City University of Hong Kong. His research interests include edge computing, Internet of Things, digital twin, network function virtualization, and combinatorial optimization.

Jianping Wang (Fellow, IEEE) received the BS and MS degrees in computer science from Nankai University, China, in 1996 and 1999, respectively, and the PhD degree in computer science from the University of Texas at Dallas, in 2003. She is currently a professor with the Department of Computer Science, City University of Hong Kong. Her research interests include cloud computing, service oriented networking, edge computing, and network performance analysis.

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 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. 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 and online algorithms, combinatorial optimization, and graph theory. He currently serves as an editor of the *IEEE Transactions on Communications*.

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, 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 the *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, Panel co-chair of IEEE INFOCOM 2011, and the general co-chair of IEEE ICDCS 2023.

Albert Y. Zomaya (Fellow, IEEE) is the Peter Nicol Russell chair professor of computer science with the School of Computer Science, Sydney University, and serves as the director with the Centre for Distributed and High-Performance Computing. He has published more than 800 scientific papers and articles and is the author, co-author, or editor of more than 30 books. He is the past editor in chief of the *IEEE Transactions on Computers*, *IEEE Transactions on Sustainable Computing*, and *ACM Computing Surveys*. He is a fellow of the Australian Academy of Science, Royal Society of New South Wales, a foreign member of Academia Europaea, and a member of the European Academy of Sciences and Arts. His research interests include the areas of parallel and distributed computing, networking, and complex systems.