Multiple Service Model Refreshments in Digital Twin-Empowered Edge Computing

Xiyuan Liang, Weifa Liang, *Senior Member, IEEE*, Zichuan Xu, *Member, IEEE*, Yuncan Zhang, *Member, IEEE*, and Xiaohua Jia, *Fellow, IEEE*

***Abstract*—Mobile Edge Computing (MEC) has emerged as a promising platform to provide various services for mobile applications at the edge of core networks while meeting stringent service delay requirements of users. Digital twin (DT) that is a mirror of a physical object in cyberspace now becomes a key player in smart cities and the Metaverse, which can be used to simulate or predict the behaviours of the object in future. To enable such a simulation or predication to be more accurate and robust, the state of the digital twin needs to be synchronized (updated) with its object quite often. The quality of inference services in a DT-empowered MEC network usually is determined by the state freshness of service models, while the quality of a service model further is determined by the state freshness of its source DT data. It is vital to refresh the states of service models frequently in order to provide high quality inference services. In this article, we study how to maximize the state freshness of both digital twins and a set of inference service models that are built upon digital twins in an MEC network, while the state freshness of a DT or a service model is achieved through frequent synchronizations between the DT and its physical object. Specifically, we first study a novel cost-aware average model freshness maximization problem with the aim to maximize the average freshness of the states of inference service models while minimizing the cost of achieving the model freshness, and show the NP-hardness of the problem. We then formulate an integer linear programming solution for the offline version of the problem, and devise a performance-guaranteed approximation algorithm for a special case of problem when the monitoring period consists of a single time slot only. Also, we develop an efficient online algorithm for the problem through scheduling objects to upload their update data to their digital twins in the network at each time slot efficiently. We finally evaluate the performance of the proposed algorithms through simulations. Simulation results demonstrate that the proposed algorithms are promising.**

***Index Terms*—Approximation algorithms, minimum-cost generalized assignment problem, mobile edge computing (MEC), digital twin state updating, scheduling algorithms, state refreshment of service models, tradeoffs between the state freshness of service models and the update cost.**

Manuscript received 23 July 2023; revised 18 October 2023; accepted 8 December 2023. Date of publication 12 December 2023; date of current version 9 October 2024. The work of Weifa Liang and Yuncan Zhang was supported by Hong Kong RGC under Grants CityU 9380137, 7005845, and 9043510, respectively. The work of Zichuan Xu was supported in part by the National Natural Science Foundation of China under Grant 61802048, and in part by the "Xinghai Scholar Program" in Dalian University of Technology, China. *(Corresponding author: Weifa Liang.)*

Xiyuan Liang is with the School of Computer and Technology, University of Science and Technology of China, Hefeii, Anhui 230026, China (e-mail: lxy8016@mail.ustc.edu.cn).

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

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

Digital Object Identifier 10.1109/TSC.2023.3341988

I. Introduction

DRIVEN by the explosive growth of the Internet of Things (IoT) and their applications, unprecedented amounts of data generated by IoT devices are invaluable to businesses, governments and organizations, which are proliferating in the physical world [12]. The emerging digital twin technique has attracted more and more attentions in digitizing physical world, through digital representation and analytics of Big Data [21]. Digital twin (DT) has emerged as a breakthrough technology to revolutionize diverse fields, including manufacturing, Internet of Things, autonomous driving, healthcare, education, smart cities and the Metaverse. By leveraging vivid simulations, DTs can provide future insights and perceptual data for users to optimize their decision-making, where the DT of an object can be used to keep the historical data, simulate the behaviours, and provide predictive decisions of the object. With the DT development, there is growing interest in model-driven service provisioning in DT-empowered mobile edge computing (MEC) networks. For example, consider an anomaly detection service model in autonomous driving environments [5], in which there are many different types of anomalies including traffic congestion, traffic violation, collision detection, vehicle breakdown, and driver fatigue or distraction. The detection and timely response to these anomalies can significantly improve driving safety and mitigate fatal vehicle accidents. The adoption of DTs of source objects in such scenario facilitates the historical traces analysis, various anomaly detection, and vehicle trajectory prediction. In this case, if the update data of objects are not uploaded on time (or objects do not synchronize with their DTs on time), the accuracy of the anomaly detection model will drastically deteriorate, and the service quality of the model will be doubtful.

To provide high quality model-driven services in edge computing, it is crucial to maintain service models as fresh as possible. The freshness of each service model is determined by its source DT data, it necessities the DT states of source data of the service model to be as fresh as possible, too. Unfortunately, existing methods and techniques of service provisioning in conventional MEC network are not applicable for model-driven services, due to lack of continuously monitoring the state information of objects, continual training on service models by using the updated source DT data, and efficient prediction

mechanisms on the behaviours of objects in future. It thus urgently needs to develop new schemes, algorithms and techniques for model-driven service provisioning in DT-empowered MEC networks.

Edge intelligence in recent years becomes a new trend in edge services, various machine learning models are trained in the edge of core networks. The trained models later are used for inference services including a simple object recognition, disease prediction and diagnosis, the behaviour simulations of objects, etc. To ensure their service accuracy, service models require to be refreshed quite often. Particularly in a dynamic environment where objects keep moving, the refreshment can be implemented through synchronizations between the data sources of service models - objects, and the DTs of the objects, where an object can upload its update data to its DT, and its DT state then is updated accordingly. The update data finally is forwarded to all service models in which the DT data is their source data for model training. To refresh service models, it is critical to synchronize the DT states of their source DT data with the objects of the DTs, as the DT state freshness of objects heavily impacts the quality of each service model ultimately.

Most existing studies focused on a single inference service model training, and assumed that the update data generated by all objects can be uploaded to the network for the model training in real-time [1], [24]. However, this assumption may not be realistic. In reality, not every object can upload its update data to its DT immediately at each time point due to various resource constraints, e.g., the limited bandwidth capacity on each AP prevents all devices under its coverage to upload their update data at the same time. In practice, each object generates data continuously, and the generated data becomes the source data for model training. Each service model needs to be refreshed through using the update data of its objects to its model retraining. Furthermore, almost all previous studies on DT services focused on services provided by a single DT [2], [9], [23], not sophisticated model-driven services that are built upon different source DT data.

In this paper, we study multiple service model refreshments in a DT-empowered MEC network. It is noticed that such a refreshment of DT states of mobile objects and service models not only consumes various network resources but also is subject to resource capacities (e.g., bandwidth and computation resource capacity constraints). To maximize the average model freshness of all service models while minimizing their refreshment cost, it poses the following challenges.

- Can we be able to develop a metric to accurately measure the freshness of DT states of objects and service models? as this metric is vital to capture the characteristics of DT state freshness and service models and cost modeling of the problem of concern.
- Due to limited bandwidth capacity imposed on each AP, should which objects be chosen to upload their update data for continual training on service models at each time slot to maximize the accumulative freshness of all models?
- Model-driven services consume both computing and bandwidth resources, how to fairly allocate the limited network resources to different service models such that the accumulative freshness of all models is maximized?
- How to strive for nontrivial trade-offs between the service accuracy of all service models and the total cost to achieve the accuracy?

In the rest of this paper, we will address the aforementioned challenges.

The novelty of this paper lies in studying model-driven service provisioning in a DT-empowered MEC network, by maximizing the average freshness of multiple inference service models while minimizing the cost incurred for the model refreshment. An innovative metric of measuring the freshness of DT states and service models is introduced, a novel cost-aware average model freshness maximization problem based on the metric is formulated, and efficient approximation and online algorithms for the problem under different settings are devised and analyzed.

The main contributions of this paper are presented as follows.

- We study the state freshness of multiple inference service models in a DT-empowered MEC network, by introducing a state freshness metric to capture the freshness of both DT states and service models, and formulating a novel cost-aware average model freshness maximization problem built upon the freshness metric.
- We show the NP-hardness of the defined problem and formulate an integer linear program solution to the offline version of the problem.
- We devise an approximation algorithm with guaranteed performance for a special case of the problem if the monitoring period consists of a single time slot only, through a nontrivial reduction to the minimum-cost generalized assignment problem (GAP), where the total cost of the GAP consists of not only the toal cost of packed items but also the total cost of all unpacked items as well.
- We develop an efficient online algorithm for the problem, by exploring nontrivial tradeoffs between the average state freshness of service models and the state updating cost of the models for the entire monitoring period. Only objects whose benefits outweigh their DT state update costs at each time slot are eligible to upload their update data to the MEC network at that time slot.
- We evaluate the performance of the proposed algorithms for the cost-aware average model freshness maximization problem through simulations. Simulation results demonstrate that the proposed algorithms are promising.

The rest of the paper is organized as follows. Section II summarizes the related work on DT state synchronizations in mobile edge computing (MEC). Section III introduces the system model, notions, notations, and the problem definition. Section IV shows the NP-hardness of the defined problem and formulates an integer linear program solution to the problem. Section V devises an approximation algorithm for a special case of the problem with a provable approximation ratio. Section VI develops an online algorithm for the problem. Section VII evaluates the performance of the proposed algorithms, and Section VIII concludes the paper.

II. Related Work

MEC has been emerged as a promising paradigm for delay-sensitive services in proximity of end-users through task offloading. Orthogonal to the MEC technology, digital twin has emerged as an enabling technology that bridges the gap between the physical world and the cyberspace, which has been used for improving the service performance of MEC platforms. Empowered by DTs, MEC platforms can provide various services for delay-sensitive IoT and AI applications such as inference services, emulation services, and decisive prediction services [3], [12], [13], [15], [17]. For example, Dong et al. [3] focused on a deep learning-based model training by mapping an MEC network to its digital twin network, and proposed an efficient algorithm for the DT network training with the aim to minimize the training energy consumption. Lin et al. [12] devised an incentive-based congestion control scheme to meet dynamic service demands of digital twins in an MEC network, by utilizing the Lyapunov optimization technique. Lu et al. [13] developed a federated learning algorithm based on the blockchain technique to enhance data privacy and security in digital twin-assisted MEC networks. Fan et al. [4] designed a DT empowered MEC framework for achieving intelligent vehicular lane-changing. They leveraged the DT technology to construct a corresponding virtual network of the physical MEC network. Yang et al. [23] investigated choosing IoT devices for participating in federated learning under the assumption that each base station has a fixed number of channels for data uploading at each training round. They developed a deep reinforcement learning (DRL) algorithm for choosing devices by leveraging actor-critic networks. All aforementioned studies make use of DTs of objects (IoT devices or cloudlets) for predicting the behaviours and performance, and none of the works considered the refreshment of service models built upon on DT data under computing and communication resource constraints, not to mention how to schedule objects to synchronize with their DTs to enhance the average model freshness of all service models.

There are several existing studies that focus on optimizing either service delays or data freshness in DT-enabled MEC networks [2], [14], [19], [20]. For example, Corneo et al. [2] studied the problem of timely dissemination of sensor updates to clouds that provide digital twin service for users with the aim to minimize the age of information (AoI). They devised a novel push-and-pull method of sensor information updating to strive for a non-trivial trade-off between sensory data freshness and the tolerable query delay. Sun et al. [19] utilized digital twins to minimize the offloading latency, through adopting the Lyapunov optimization technique. Lu et al. [14] devised a deep learning-based algorithm to cope with the mobility of mobile devices and migration of digital twins, with the aim to minimize the average service delay. Han et al. [7] introduced a dynamic hierarchical framework in which IoT devices sense and collect physical objects' status information to assist service provider in synchronizing DTs. Li et al. [8], [9] recently considered AoI-aware DT state updating and service issues. Almost of all above studies assumed that DT services are based on a single DT of each object, none of the studies considered machine-learning based service models with multiple source DT data, and the relationship between the service quality of a service model and the frequency (or the update cost) of their source DT updating.

Unlike the aforementioned studies, in this paper we study high quality model-driven service provisioning in a DT-empowered MEC network through state refreshments of DTs and multiple service models. The work in this paper distinguishing from existing works lies in that almost all previous studies focused only on a single service model training and the generated data of all objects can be uploaded for the model training in real-time manner. In contrast, we here deal with multiple service models and not every object can upload its update data at each time slot immediately, due to bandwidth capacity on each AP. Also, we deal with the refreshment of all service models whenever there is any updating on their source data, by formulating a novel service model refreshment problem. To the best of our knowledge, this is the first study on the state freshness of both DTs and service models in a DT-empowered MEC network, and efficient algorithms for the state refreshment of multiple service models are developed.

III. Preliminaries

In this section, we first introduce the system model. We then model the state freshness of DT and service models by introducing a metric. Built upon the freshness metric, we then detail the cost of various resource consumed for the state freshness improvement. We finally define the problem precisely.

A. System Model

We consider a mobile edge computing (MEC) network $G = (N, E)$, where N is the set of Access Points (APs), and E is the set of optical links between APs. With each AP, there is a co-located cloudlet with the AP through a high speed optical fiber cable, and the communication delay between the AP and its co-located cloudlet is neglected. For convenience, an AP and its co-located cloudlet can be used interchangeably if no confusion arises. The digital twins (DTs) of different physical objects are deployed in different cloudlets of G. Let V be the set of objects. Assume that the DT of each object has been deployed in a cloudlet already, and further assume that the computing and storage resources in a cloudlet hosting a DT are sufficient for the DT data processing and the storage demand of the update data of its object for all time slots. The state of each DT can be synchronized (updated) with its physical object when the update data of the object is uploaded to the network at a time slot. Denote by $s(v_i)$ the co-located cloudlet of the AP at which object v_i uploads its data to the MEC network and $h(v_i)$ the cloudlet hosting its DT, DT_i, respectively. We assume that there are M inference service models in the MEC network that need to be retrained often, and each service model M_m has a home cloudlet $h(M_m)$ that provides services of the model for its users, e.g., inference services, where the data sources of each model M_m are the DTs of a subset $V(M_m)$ of objects with $V(M_m) \subseteq V$ and $1 \leq m \leq M$. We further assume that the allocated amount of computing resource in cloudlet $h(M_m)$ for model M_m suffices for the model training whenever there are any data updates on

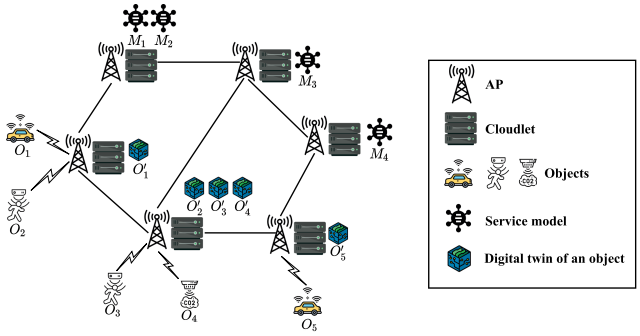


Fig. 1. Illustrative example of an DT-empowered MEC network, where the DTs of five objects $O_1, \ldots, O_5$ are $O'_1, \ldots, O'_5$ that are deployed in three cloudlets. Also, there are four service models $M_1, \ldots, M_4$ deployed in three cloudlets, where data sources of each model come from the DTs of objects, e.g., assume that the data sources of M_2 are O_2, O_3 and O_4, and the data sources of model M_4 are O_1 and O_5, respectively.

its data sources - objects. Users request different model services through issuing queries.

We say that an object is under the coverage of an AP if the AP is within the maximum transmission range of the object. Let $\mathcal{C}(AP_j, t)$ be the set of objects under the coverage of AP_j at time slot t. If object v_i is under the coverage of AP_j at time slot t (i.e., $PX_{ij} \leq PX_i^{max}$), then $v_i \in \mathcal{C}(AP_j, t)$, where PX_i^{max} is the maximum transmission power of object v_i and PX_{ij} is proportional to the square of the euclidean distance $dist(v_i, AP_j)$ between the location of object v_i and the location of AP_j, i.e., $PX_{ij} \propto dist(v_i, AP_j)^2$. Each object $v_i \in V$ uses a transmission power PX_{ij} to upload its update data to AP_j. We further assume that the bandwidth capacity W_j on each AP_j is fixed, and the accumulative bandwidth for all uploading objects under AP_j is upper bounded by its bandwidth capacity W_j with $1 \leq j \leq |N|$. Furthermore, each object can be covered by multiple APs. An illustrative example of an DT-empowered MEC network is given in Fig. 1, where the DTs of objects and service models built upon the DT data of some objects are placed in cloudlets of the network.

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

B. Metrics for State Freshness of Both DTs and Service Models

We start by introducing the freshness of DT states and inference service models. Considering a finite time horizon $\mathbb{T}$ that is divided into T equal time slots, the state freshness $r(DT_i, t)$ of the digital twin DT_i of object $v_i \in V$ at time slot $t \in \mathbb{T}$ is defined as

$$r(DT_i, t) = \begin{cases} a^{t-t_0^{(i)}}, & \text{if} DT_i \text{ is not updated at time slot } t \\ 1, & \text{otherwise} \end{cases} \tag{1}$$

where $a > 1$ is a non-negative constant, representing the decay rate of the DT state freshness of an object with the time length of no update data uploading, $t_0^{(i)}$ is the last state update time slot of DT_i of object v_i and $t_0^{(i)} < t$, and $r(DT_i, 0) = \infty$ initially.

The rationale behind the state freshness metric of DTs is that the freshness decay of a DT exponentially grows with the amount of time elapsed since the last synchronizations of the DT with its object. This metric is similar to the freshness metric of services in [11] and the user satisfaction metric on services in [10], [16].

It can be seen that if the state of DT_i is updated at time slot t, its freshness becomes 1; otherwise, its freshness is larger than 1, and its freshness value becomes larger and larger without the state update over time. Meanwhile, it is noticed that the volume $vol(DT_i, t)$ of the update data generated by object v_i since its last uploading is inversely proportional to the time duration without its DT state update. If a DT is frequently updated, the volume of its update data per update round will be relatively small. Otherwise, the volume of the update data will be large.

The state freshness $r(M_m, t)$ of service model M_m in the beginning of time slot t is defined as

$$r(M_m, t) = \frac{\sum_{v_i \in V(M_m)} r(DT_i, t)}{|M_m|} = \frac{\sum_{v_i \in V(M_m)} a^{t-t_0^{(i)}}}{|M_m|} \tag{2}$$

where $t_0^{(i)}$ is the last state update time of DT_i and $t_0^{(i)} < t$, set $V(M_m)$ $(\subseteq V)$ is the set of objects for M_m, and $|M_m|$ is the number of objects as the data sources of model M_m.

It must be mentioned that, unlike the DT state freshness of an object that is purely determined by its DT state update frequency, the state freshness of a service model is much complicated, which is jointly determined not only by the number of its data sources (objects) but also by the state freshness of each of these data sources. We thus use the average DT freshness of the data sources of each service model as the state freshness of the service model. It can be seen that the smaller the state value $r(M_m, t)$ of model M_m at time slot t, the fresher the model. Particularly, the state value of $r(M_m, t)$ will be 1 if the DT of each object in $V(M_m)$ is updated at time slot t, and model M_m will be the freshest one in the next time slot $t + 1$. Otherwise, a larger state value of $r(M_m, t)$ indicates that most of the data sources of model M_m have not been refreshed by time slot t, assuming that $r(M_m, 0) = \infty$ initially.

C. Costs of State Refreshment of Both DTs and Service Models

We then develop cost modeling for improving the freshness of DT states and models through uploading update data from the objects of the DTs. The state freshness updates of DTs and service models are achieved by consuming computing, storage and bandwidth resources of the MEC network and objects (IoT devices). The costs of computing, storage and bandwidth resources consumed by uploading the update data of each object $v_i \in V$ at time slot t are defined as follows.

The data uploading cost $c_{up\_load}(v_i, t)$ of each object v_i to an AP_j at time slot t is proportional to the energy consumption on the object, which is defined as follows.

$$c_{up\_load}(v_i, AP_j, t) = \eta_1 \cdot PX_{ij} \cdot \frac{vol(DT_i, t)}{R_{ij}(t)}, \tag{3}$$

where η_1 is the cost of per time unit energy consumption, PX_{ij} is the transmission power of object v_i to AP_j, which is

TABLE I
TABLE OF SYMBOLS

Notations	Descriptions
$G = (N, E)$	An MEC network as an undirected graph $G = (N, E)$, where N is a set of Access Points (APs) or colocated cloudlets, and E is a set of links connecting APs
V, v_i, DT_i	The set of objects, and $v_i \in V$ is an object and its DT, DT_i is placed into the MEC already
$dist(v_i, AP_j)$	The Euclidean distance between the location of object v_i and the location of AP_j
$vol(DT_i, t)$	The volume of update data of object v_i at time slot t
$r(DT_i, t)$	The freshness value of the DT, DT_i, of object v_i at time slot t
$s(v_i)$ and $h(v_i)$	The co-located cloudlet $s(v_i)$ of the AP at which object v_i uploads its data and $h(v_i)$ the cloudlet hosting DT_i
$\mathcal{C}(AP_j, t)$	The set of objects under the coverage of AP_j at time slot t
$h(M_m)$	The home cloudlet $h(M_m)$ of model M_m that provides inference services
$r(M_m, t)$	The freshness value of model M_m at time slot t
$V(M_m)$, $\|V(M_m)\|$	The number of DT data sources of model M_m and its cardinality
$V_{m,t}$	The set of objects that upload their update data to model M_m in cloudlet $h(M_m)$ and $V_{m,t} \subseteq V(M_m)$ at time slot t
$c(DT_i, t)$	The routing cost $c(DT_i, t)$ of the update date of object v_i from AP (cloudlet) $s(v_i)$ to $h(v_i)$ at time slot t
$P_{s(v_i),h(v_i)}$	is The shortest path in G between cloudlets $s(v_i)$ and $h(v_i)$
$w(e)$	$w(e)$ is the routing cost of a unit data along link $e \in E$
$c_{route}(M_m, t)$	The routing cost $c_{route}(M_m, t)$ of the DT update data of objects in $V_{m,t}$ routing from their DT homes to the model home $h(M_m)$ at time slot t
$c_{up\_load}(v_i, AP_j, t)$	The data uploading cost of object v_i to an AP_j at time slot t
PX_{ij}, PX_i^{max}	The transmission power of object v_i to AP_j, and PX_i^{max} is the maximum transmission power of object v_i
W_j	The bandwidth capacity of AP j
$R_{ij}(t)$	The data transmission rate of object v_i under the coverage of AP_j at time slot t
$c_{comp}(M_m, t)$	The local processing cost of model M_m at cloudlet $h(M_m)$ at time slot t
$C_{update}(t)$	The total cost of various resources consumed for the state refreshment of all models at time slot t
η_1, η_2	η_1 is the cost of per time unit energy consumption, and η_2 is the unit time usage cost of the CPU
$f_{h(M_m)}$	$f_{h(M_m)}$ is the CPU frequency of cloudlet $h(M_m)$
$\triangle_{cost}(v_i, AP_j, t)$	The cost of various resources consumed by uploading the update data of object v_i to AP_j at time slot t
$\nabla_{cost}(v_i, t)$	The amount of cost reduction of the objective function due to the update data uploading of object v_i at time slot t
$\delta(v_i, AP_j, t)$	$\delta(v_i, AP_j, t) = \nabla_{cost}(v_i, t) - \triangle_{cost}(v_i, AP_j, t)$ at time slot t
$c_{ul}(v_i, AP_j, t)$	The cost contribution of object v_i to the objective function if its update data is uploaded to AP_j at time slot t
$c_{nl}(v_i, t)$	The cost contribution of object v_i to the objective function if its update data does not be uploaded at time slot t
$x_{i,j,t}$	$x_{i,j,t}$ is a binary decision variable, if $x_{i,j,t} = 1$ and $v_i \in \mathcal{C}(AP_j, t)$, then object v_i will upload its update data to its DT_i via AP_j; otherwise $x_{i,j,t} = 0$ at time slot t
α, β	The constant coefficients
a	$a > 1$ is a non-negative constant, representing the decay rate of the DT state freshness of an object with the time length of no update data uploading
$\mathbb{T}$, t, and T	The entire time horizon $\mathbb{T}$, $t \in \mathbb{T}$ is a time slot, and $T = \|\mathbb{T}\|$

proportional to the square of the distance between object v_i and AP_j, $vol(DT_i, t)$ is the volume of the update data of v_i at time slot t, $R_{ij}(t)$ is the data transmission rate of v_i at time slot t, assuming that object v_i is under the coverage of AP_j (co-located with cloudlet $s(v_i)$), and W_j is the bandwidth capacity on AP_j with $1 \leq j \leq |N|$. Note that an object v_i is under the coverage of AP_j if AP_j is within the maximum transmission range of object v_i. Thus, an object may be covered by multiple APs.

$$R_{ij}(t) = W_j \log\left(1 + \frac{PX_{ij} \cdot H_{i,j}}{\rho}\right), \tag{4}$$

where PX_{ij} the transmission power of v_i to AP_j, $H_{i,j}$ is the channel power gain of object v_i, and ρ is the noise power spectral density.

The data routing cost $c(DT_i, t)$ of DT_i of object v_i at time slot t is the cost of communication resource consumption by routing its update data from cloudlet $s(v_i)$ to the cloudlet $h(v_i)$ hosting its DT for processing and prediction, which is defined as follows.

$$c(DT_i, t) = \sum_{e \in P_{s(v_i),h(v_i)}} w(e) \cdot vol(DT_i, t), \tag{5}$$

where $P_{s(v_i),h(v_i)}$ is the shortest path in the MEC network G between cloudlets $s(v_i)$ and $h(v_i)$, and $w(e)$ is the routing cost of a unit data along link $e \in E$.

Given an inference service model M_m, its model updating cost is the sum of two component costs: one is to route the update data of its DTs to its home cloudlet $h(M_m)$ for processing; and another is to aggregate the collected update data from its DTs at cloudlet $h(M_m)$ to form an aggregate result by local computing, where $h(M_m)$ is the cloudlet allocated to model M_m for data aggregation. The update data of an object $v_i \in V(M_m)$ for model M_m needs to be routed from its DT home $h(v_i)$ to the model home $h(M_m)$, and the routing path is obtained by finding a single source shortest path tree (SPT) rooted at $h(M_m)$ and spanning all nodes hosting the updated DTs of objects in

$V_{m,t} \subseteq V(M_m)$, where $V_{m,t}$ is the set of objects uploading their update data at time slot t. Let $T_{m,t}$ be the SPT. Notice that different SPTs are built at different time slots, as different objects update their DTs at different time slots.

The routing cost $c_{route}(M_m, t)$ of the update data of DTs of all objects in $V_{m,t}$ routing from their DT homes to the model home of M_m at time slot t is defined as

$$c_{route}(M_m, t) = \sum_{v_i \in V_{m,t}} \sum_{e \in P_{h(v_i),h(M_m)}} w(e) \cdot vol(DT_i, t), \tag{6}$$

where $V_{m,t}$ is a subset of $V(M_m)$ in which the update data of objects will be uploaded at time slot t, $P_{h(v_i),h(M_m)}$ is the routing path in tree $T_{m,t}$ between cloudlets $h(v_i)$ and $h(M_m)$.

The local data processing cost $c_{comp}(M_m, t)$ of model M_m at time slot t is the cost of computing resource consumption of cloudlet $h(M_m)$ for updating the state of model M_m due to the state updating of some of its DTs at time slot t, i.e.,

$$c_{comp}(M_m, t) = \eta_2 \cdot \frac{\sum_{v_i \in V_{m,t}} vol(DT_i, t)}{f_{h(M_m)}}, \tag{7}$$

where η_2 is the unit time usage cost of the CPU of cloudlet $h(M_m)$, and $f_{h(M_m)}$ is the CPU frequency of cloudlet $h(M_m)$.

The total cost $C_{update}(t)$ of various resources consumed for the state refreshment of all models at time slot t is written as follows.

$$C_{update}(t) = \sum_{v_i \in V} \left(\frac{\sum_{m=1}^{M} \{c_{up\_load}(v_i, t) + c(DT_i, t) \mid v_i \in V_{m,t}\}}{\sum_{m=1}^{M} \{1 \mid v_i \in V_{m,t}\}} \right) + \sum_{m=1}^{M} \left(c_{route}(M_m, t) + c_{comp}(M_m, t) \right), \tag{8}$$

where $V_{m,t}$ is a subset of $V(M_m)$ in which the update data of objects will be uploaded at time slot t, $c_{up\_load}(v_i, t)$ and $c(DT_i, t)$ are defined in (3) and (5), respectively. Notice that the first term in the right hand side of (8) is a result when the update data of each object v_i is uploaded and sent to the cloudlet hosting DT_i at time slot t once, while the second term is the cost sum of routing their update data of DTs of the objects to their model home $h(M_m)$ for processing, and the model updating processing takes place at the model home.

D. Problem Definition

Given an MEC network $G = (N, E)$ and a finite time horizon $\mathbb{T}$ that is divided into T equal time slots, there is a set V of mobile objects with each having a DT placed in a cloudlet, and there are M inference service models $M_1, \ldots, M_M$ deployed at different cloudlets, where model M_m is trained by data samples from a set $V(M_m)$ ($\subseteq V$) of objects. Let $V_{m,t}$ be the set of objects uploading their update data to the MEC network at time slot $t \in \mathbb{T}$, then $V_{m,t} \subseteq V(M_m)$. The accuracy of each service model M_m is determined by the data quality and volume of its training data, or the freshness of the DT states of objects in $V(M_m)$ with $1 \leq m \leq M$. However, not the update data of each object can be uploaded to its DT in a cloudlet in real-time due to various resource constraints on the MEC network, e.g., the limited bandwidth capacity on each AP. *The cost-aware average model freshness maximization problem* in $G(N, E)$ is to schedule DT updating of mobile objects in V at each time slot $t \in \{1, 2, \ldots, T\}$ such that the average state of all service models as fresh as possible while keeping the state updating cost of the models as small as possible, i.e., the problem optimization objective is to

$$\text{minimize} \quad \sum_{t=1}^{T} \left(\alpha \cdot \sum_{m=1}^{M} r(M_m, t) + \beta \cdot C_{update}(t) \right), \tag{9}$$

subject to the bandwidth capacity on each AP, where α and β are non-negative coefficients used for balancing between the average freshness of all models and the total state updating cost of achieving the model freshness, and $C_{update}(t)$ is the total state updating cost of various resources consumed by uploading the update data of objects in $\bigcup_{m=1}^{M} V_{m,t}$ at each time slot t with $1 \leq t \leq T$.

It can be seen from the objective function in (9) that at each time slot t, a smaller value of $\sum_{m=1}^{M} r(M_m, t)$ indicates the average state of all models is fresher. Meanwhile, a smaller value of $C_{update}(t)$ implies a less updating cost for refreshing the average state of all models.

IV. NP-Hardness and an ILP Solution

In this section, we first show the NP-hardness of the cost-aware average model freshness maximization problem. We then formulate an integer linear program (ILP) solution for the problem in offline setting, assuming that both the volume of the update data and the location of each object at each time slot are given.

A. NP-Hardness of the Problem

Theorem 1. The cost-aware average model freshness maximization problem in an MEC network $G(N, E)$ is NP-hard.

Proof. We consider a special case of the problem when the time horizon $\mathbb{T}$ consists of one time slot only, the MEC network consists of one AP and its co-located cloudlet only, the update costs of DT states and models are neglected, and the average freshness of all models is considered as the optimization objective of the problem.

Since the bandwidth capacity W on each AP is fixed, not all objects under the coverage of the AP can upload their update data to the AP at the same time slot, the problem then is to identify that which objects can upload their update data to their DTs in cloudlets and the update data then are used for training their service models, such that the average freshness of all models is maximized, subject to the bandwidth capacity on each AP. We reduce the Knapsack problem to this special cost-aware average model freshness maximization problem as follows.

There are n items and a bin with capacity W, and each item a_i has a cost $r(a_i)$ and size $s(a_i)$ with $1 \leq i \leq n$. The Knapsack problem is to pack as many items as possible into the bin such that the total cost of packed items is minimized, subject to the

bin capacity. We reduce the knapsack problem to this special problem as follows.

Each item a_i corresponds to an object v_i, the cost $r(a_i)$ of item a_i corresponds to the DT state freshness $r(v_i)$ of object v_i at that time slot, the size $s(a_i)$ of item a_i corresponds to the amount R_i of allocated bandwidth to object v_i to upload its update data, and the bin capacity W corresponds to the bandwidth capacity on the AP. Thus, if there is an optimal solution to this special cost-aware average model freshness problem, there is an optimal solution to the Knapsack problem. As the latter is NP-hard, the problem of concern is NP-hard, too. ■

B. ILP Formulation

Given an object $v_i \in V$, let $\triangle_{cost}(v_i, AP_j, t)$ be the cost of various resources consumed by uploading its update data to AP_j at time slot t, then

$$\begin{aligned}
&\triangle_{cost}(v_i, AP_j, t)\\
&= \beta \cdot \frac{\sum_{m=1}^{M}\{c_{up_load}(v_i, AP_j, t) \mid v_i \in V_{m,t}\}}{\sum_{m=1}^{M}\{1 \mid v_i \in V_{m,t}\}}\\
&\quad + \beta \cdot \frac{\sum_{e \in P_{s(v_i),h(v_i)}} w(e) \cdot vol(DT_i, t) \mid v_i \in V_{m,t}\}}{\sum_{m=1}^{M}\{1 \mid v_i \in V_{m,t}\}}\\
&\quad + \beta \cdot \sum_{m=1}^{M} \sum_{v_i \in V_{m,t}\ \&\ e \in P_{h(v_i),h(M_m)}} w(e) \cdot vol(DT_i, t),\\
&\quad + \beta \cdot \sum_{m=1}^{M} \sum_{v_i \in V_{m,t}\ \&\ e \in P_{h(v_i),h(M_m)}} \eta_1 \cdot \frac{vol(DT_i, t)}{f_{h(M_m)}}, \qquad (10)
\end{aligned}$$

where $c_{up\_load}(v_i, AP_j, t)$ is defined in (3), $V_{m,t}$ is the set of objects that upload their update data to model M_m in cloudlet $h(M_m)$ at time slot t with $V_{m,t} \subseteq V(M_m)$, $f_{h(M_m)}$ is the CPU frequency of cloudlet $h(M_m)$, and $\triangle_{cost}(v_i, 0) = 0$ initially.

If object v_i uploads its update data to AP_j at time slot t, its cost contribution $c_{ul}(v_i, AP_j, t)$ toward the objective function is given as follows.

$$\begin{aligned}
c_{ul}(v_i, AP_j, t) &= \alpha \cdot \sum_{m=1}^{M} \frac{\{r(DT_i, t+1) \mid v_i \in V_{m,t}\}}{|M_m|}\\
&\quad + \triangle_{cost}(v_i, AP_j, t)\\
&= \alpha \cdot \sum_{m=1}^{M} \frac{\{1 \mid v_i \in V_{m,t}\}}{|M_m|} + \triangle_{cost}(v_i, AP_j, t),
\end{aligned} \qquad (11)$$

where the first term in the right hand side of (11) is the accumulative freshness to all models contributed by object v_i at time slot t, and the second term $\triangle_{cost}(v_i, AP_j, t)$ defined in (11) is the cost of all resources consumed for the DT state updating of v_i and the freshness improvement of all models. If DT_i of object v_i is updated in the end of time slot t, its state freshness value at time slot $t+1$ will become 1, i.e., $r(DT_i, t+1) = 1$. We also assume that $c_{ul}(v_i, 0) = 0$ for each $v_i \in V$ initially. Otherwise (there is no update data uploading from object v_i at time slot t), the cost contribution $c_{nl}(v_i, t)$ of object v_i toward the objective function is given as follows.

$$\begin{aligned}
&c_{nl}(v_i, t)\\
&= \alpha \cdot \sum_{m=1}^{M} \frac{\{r(DT_i, t+1) \mid v_i \in V(M_m)\ \&\ v_i \notin V_{m,t}\}}{|M_m|}\\
&= \alpha \cdot \sum_{m=1}^{M} \frac{\{a \cdot r(DT_i, t) \mid v_i \in V(M_m)\ \&\ v_i \notin V_{m,t}\}}{|M_m|},
\end{aligned} \qquad (12)$$

assuming that $c_{nl}(v_i, 0) = 0$ for each $v_i \in V$ initially.

Let $x_{i,j,t}$ be a binary decision variable, where $x_{i,j,t} = 1$ means that object v_i will upload its update data to its DT_i via AP_j at time slot t if $v_i \in \mathcal{C}(AP_j, t)$; otherwise, $x_{i,j,t} = 0$, where $\mathcal{C}(AP_j, t)$ is the set of objects under the coverage of AP_j at time slot t.

The integer linear program (ILP) for the cost-aware average model freshness maximization problem is presented as follows.

$$\text{Minimize} \sum_{t=1}^{T} \left(\alpha \cdot \sum_{m=1}^{M} r(M_m, t) + \beta \cdot C_{update}(t) \right). \qquad (13)$$

Formula (13) can be equivalently rewritten as

$$\begin{aligned}
\text{Minimize} \quad & \sum_{t=1}^{T} \sum_{v_i \in V} \sum_{j=1}^{|N|} \left(c_{ul}(v_i, AP_j, t) \cdot x_{i,j,t} \right.\\
& \left. + c_{nl}(v_i, t) \cdot (1 - x_{i,j,t}) \right), \qquad (14)
\end{aligned}$$

subject to: (1)–(12),

$$R_{ij}(t) = W_j \log\left(1 + \frac{PX_{ij} \cdot H_{i,j}}{\rho}\right),$$
$$\forall v_i \in V,\ v_i \in \mathcal{C}(AP_j, t),\ \exists\, j \in [1, |N|] \qquad (15)$$

$$\sum_{v_i \in \mathcal{C}(AP_j, t)} R_{ij}(t) \cdot x_{i,j,t} \le W_j,$$
$$\forall v_i \in V,\ \forall j \in [1, |N|],\ \forall t \in [1, T] \qquad (16)$$

$$\sum_{j=1}^{|\mathcal{N}|} x_{i,j,t} \le 1,\ \ \forall v_i \in V, \forall t \in [1, T] \qquad (17)$$

$$x_{i,j,t} \in \{0, 1\},\ \forall v_i \in V,\ \forall t \in [1, T],\ \forall j \in [1, |N|] \qquad (18)$$

$$x_{i,j,t} = 0 \text{ if } v_i \notin \mathcal{C}(AP_j, t),$$
$$\forall v_i \in V,\ \forall t \in [1, T],\ \forall j \in [1, |N|] \qquad (19)$$

where the optimization objective (14) is the accumulative cost of refreshing all model states and the DT state updating of all objects for a given time horizon $\mathbb{T}$. Constraint (15) is the data transmission rate of object v_i at time slot t. Constraint (16) ensures that no bandwidth resource capacity W_j on AP_j is violated. Constraint (17) ensures that each object can at most upload to one AP at each time slot t. Note that each object v_i can be at a different location at a different time slot if the

object is movable, i.e., it is possible that $v_i \in \mathcal{C}(AP_j, t)$ and $v_i \in \mathcal{C}(AP_{j'}, t')$ if $j \neq j'$ and $t \neq t'$.

V. Approximation Algorithm for the Special Cost-Aware Average Model Freshness Maximization Problem

The ILP solution in the previous section is not scalable when the problem size is large, and its main purpose serves as a benchmark against other algorithms when the problem size is small. In this section, we consider a special case of the problem, where the monitoring period consists of a single time slot only. Even for this special case, the problem is still NP-hard, we devise an approximation algorithm for it and show the correctness of the approximate solution.

A. Approximation Algorithm

We devise an approximation algorithm for the special cost-aware average model freshness maximization problem at a given time slot $t \in \mathbb{T}$ with the aim to minimize the following objective function.

$$fresh(M,t) = \alpha \cdot \sum_{m=1}^{M} r(M_m, t) + \beta \cdot C_{update}(t), \quad (20)$$

where $r(M_m, t)$ and $C_{update}(t)$ are defined in (2) and (8), respectively.

The idea behind the proposed approximation algorithm is a non-trivial application of an approximation algorithm for the minimum cost general assignment problem (GAP) [18].

Traditionally, the GAP is to pack as many items as possible into bins such that the total cost of packed items is minimized, subject to the capacity on each bin. In contrast, the problem we consider is to minimize the sum of the total cost of items packed to the bins and the total cost of items that have not been packed to any bin at all. We reduce the special cost-aware average model freshness maximization problem to the GAP as follows.

We create a virtual bin with sufficient capacity to accommodate all items, i.e., each unpacked item can be packed to this virtual bin. We ensure that if an item can be packed to a bin with less cost than it is packed to the virtual bin, then it will never be packed to the virtual bin. In other words, if an item is placed in the virtual bin, then the cost of placing it to any bin is strictly larger than the cost of placing it in the virtual bin. Specifically, there are $|N|$ bins with each corresponding to an AP in the MEC network, the capacity on bin B_j is its bandwidth capacity W_j with $1 \leq j \leq |N|$, and the size of item (object) v_i is its transmission rate $R_{ij}(t)$ if it is packed to AP_j (i.e., it is under the coverage of AP_j), and $R_{ij}(t)$ can be calculated by (4). In addition to the original $|N|$ bins corresponding to the $|N|$ APs, there is also a *virtual bin* $B_{|N|+1}$ with capacity $W_{|N|+1} = 2|N| \cdot \max\{W_j \mid 1 \leq j \leq |N|\}$, its capacity setting ensures that every item can be packed into it if the item cannot be packed to one of the original $|N|$ bins.

Now, if object v_i is assigned to bin B_j (the AP) under its coverage with $1 \leq j \leq |N|$, then it incurs a cost $cost(i,j) = c_{ul}(v_i, AP_j, t)$ that is defined in (11) with size $size(i,j) = R_{ij}(t)$, which implies that the update data of object v_i will be uploaded at time slot t; otherwise, the cost of packing v_i into bin B_j is $cost(i,j) = \infty$ with size $size(i,j) = 2 \cdot \max\{W_j \mid 1 \leq j \leq |N|\}$ to enforce that it will not be packed to bin B_j. That is, if object v_i is not in the coverage of AP_j, its packing cost is infinity and its size is no less than twice the capacity of the bin. Notice that all unpacked items will be packed into the virtual bin $B_{|N|+1}$, where the cost $cost(i, |N|+1)$ of packing item v_i into bin $B_{|N|+1}$ is $c_{nl}(v_i, t)$, which is defined in (12), and the size is $size(i, |N|+1) = \max\{W_j \mid 1 \leq j \leq |N|\}$, assuming that $r(DT_i, 0) = \infty$ initially.

In summary, if an item is packed into one of the $|N|$ bins, say bin j, its update data will be uploaded at time slot t, and its contribution to the objective function is $c_{ul}(v_i, AP_j, t)$, which the sum of the cost of all DT state updating and the refreshment cost of all models at time slot t; otherwise, the item must be packed to the virtual bin, and its cost is the sum of the freshness values of all model freshness of its contribution in the end of time slot t, and the value of its DT state freshness is $c_{nl}(v_i, t)$ that is its value at time slot $t-1$ multiplied by the constant a in (1). It can be seen that all objects are packed into these $|N|$ bins and the virtual bin. Thus, this minimum cost generalized assignment problem is to minimize the total cost of all packed items while meeting the capacity constraint on each bin, for which there is an approximation algorithm that delivers an optimal solution, at the expense of twice the bandwidth capacity violations on each AP [18].

The detailed approximation algorithm for the special cost-aware average model freshness maximization problem is given in Algorithm 1.

B. Algorithm Analysis

In the following, we show the correctness of the approximate solution and analyze the approximation ratio of the proposed approximation algorithm.

Lemma 1: For the solution delivered by the approximation algorithm Algorithm 1 at each time slot t, (i) each item $v_i \in V$ must be packed to a bin B_j with $1 \leq j \leq |N|$ or the virtual bin $B_{|N|+1}$. (ii) If an item is packed to bin B_j with $1 \leq j \leq |N|$, its update data will be uploaded at time slot t; otherwise, there is no updating on its DT state at time slot t. (iii) If there is an item v_i with $c_{ul}(v_i, AP_j, t) - c_{nl}(v_i, t) \geq 0$, it is impossible to pack it to any bin B_j with $1 \leq j \leq |N|$. In other words, it must be packed into bin $B_{|N|+1}$, where $c_{ul}(v_i, AP_j, t)$ and $c_{nl}(v_i, t)$ are defined by (11) and (12), respectively.

Proof: We first show claim (i). For each $v_i \in V$, if it is not packed into a bin B_j with $1 \leq j \leq |N|$, it must be packed to bin $B_{|N|+1}$ because bin $B_{|N|+1}$ has sufficient capacity to accommodate all items.

We then prove claim (ii). If an item v_i is packed into bin B_j with $1 \leq j \leq |N|$, following the proposed algorithm, its update data will be uploaded at time slot t, and its DT, DT_i, will be updated at that time slot, too.

We finally show claim (iii). For a given object $v_i \in V$, if $c_{ul}(v_i, AP_j, t) - c_{nl}(v_i, t) \geq 0$, this implies that the net cost contribution of object v_i towards the objective function increases,

Algorithm 1: Approximation Algorithm for the Cost-Aware Average Model Freshness Maximization Problem at Time Slot $t \in \mathbb{T}$.

Input: assume that the volume $vol(DT_i, t)$ of the update data of each object $v_i \in V$ at time slot t is given, its AP $s(v_i)$, its DT cloudlet $h(v_i)$ and its model M_m cloudlet $h(M_m)$ with $1 \leq m \leq M$ are given, too.

Output: choose a set S_t of objects to upload their update data to their DTs, and their service models will be trained using the updated DT dataa at each time slot t with $1 \leq t \leq T$.

$S_t \leftarrow \emptyset$; /* the solution for uploading update data of objects at time slot t */
for each object $v_i \in V$ **do**
  for each $AP_j \in N$ **do**
    Compute $\triangle_{cost}(v_i, AP_j, t)$ by (10);
    Compute $c_{ul}(v_i, AP_j, t)$ by (11);
  end for ;
  Compute $c_{nl}(v_i, t)$ by (12);
end for ;
for $j \leftarrow$ to $|N|+1$ **do**
  if $j \leq |N|$ **then**
    Create bin B_j with capacity W_j;
    $\mathcal{C}(AP_j, t) \leftarrow \{v_i \mid$ if v_i is the coverage of $AP_j\}$;
  else
    Create bin $B_{|N|+1}$ with capacity of $2|N| \cdot \max\{W_j \mid 1 \leq j \leq |N|\}$;
    $\mathcal{C}(AP_{|N|+1}, t) \leftarrow \{v_i \mid \forall v_i \in V\}$;
  end if ;
end for ;
for each object $v_i \in V$ **do**
  for each bin B_j **do**
    if $v_i \in \mathcal{C}(AP_j, t)$ & $j \neq |N|+1$ **then**
      Calculate the transmission rate $R_{ij}(t)$ of object v_i by (4);
      $cost(i,j) \leftarrow c_{ul}(v_i, AP_j, t)$; $size(i,j) \leftarrow R_{ij}(t)$;
    else
      if $j \leq |N|$ **then**
        $cost(i,j) \leftarrow \infty$;
        $size(i,j) \leftarrow 2 \cdot \max\{W_j \mid 1 \leq j \leq |N|\}$;
      else
        $cost(i, |N|+1) \leftarrow c_{nl}(v_i, t)$;
        $size(i, |N|+1) \leftarrow \max\{W_j \mid 1 \leq j \leq |N|\}$;
      end if ;
    end if ;
  end for ;
end for ;
Find an approximate solution $\mathbb{S}$ for the minimum-cost GAP, by applying the approximation algorithm due to Shmoys and Tardos [18];
for each $v_i \in V$ **do**
  if v_i is packed to bin B_j in $\mathbb{S}$ with $j \leq |N|$ **then**
    $S_t \leftarrow S_t \cup \{\langle v_i, AP_j \rangle\}$;
  end if
end for
return S_t.

compared with the case where object v_i does not upload its update data at time slot t, i.e., $c_{ul}(v_i, AP_j, t) \geq c_{nl}(v_i, t)$. In other words, if object v_i is allocated to bin $B_{|N|+1}$, its cost contribution to the objective function is less than its cost contribution to the objective function when it uploads its update data at time slot t. Otherwise, assume that v_i has been allocated to a bin B_j with $j \leq |N|$ by the proposed scheduling algorithm and there is another scheduler, in which object v_i is allocated to bin $B_{|N|+1}$. This will result in a less cost solution, which contradicts the assumption that the cost of the solution delivered by the proposed algorithm is the minimum one, the lemma thus follows. ■

Theorem 2: Given an MEC network $G = (N, E)$ and a set V of mobile objects that their DTs have been placed into cloudlets already, a set of M inference service models that are deployed to different cloudlets for services, assume that the source data of these service models come from the DTs of objects in V, and the models need to be refreshed often to maintain their service accuracy, due to the updatings on their source DT data. There is an approximation algorithm, `Algorithm 1`, for the special cost-aware average model freshness maximization problem at a single time slot t, which delivers an optimal solution, at the expense of twice the bandwidth capacity violations on any AP in G.

Proof: As shown in Theorem 1, even for a single time slot, the problem is still NP-hard.

Following the proposed approximation algorithm Algorithm 1, all items in V will be packed to either $|N|$ bins or the virtual bin. If an object is packed to one of the $|N|$ bins (cloudlets at their co-located APs), the update data of the object will be uploaded at time slot t; otherwise, the object will not upload its update data at time slot t, and its corresponding item must be packed to the virtual bin. Thus, the total cost of refreshing all models due to DT state updating of some objects at time slot t is $fresh(M, t)$, which is defined in (20). The GAP is to minimize the value of $fresh(M, t)$. Following Shomys and Tardos [18], there is an approximation algorithm for the GAP, which delivers an optimal soluiton, at the expense of twice the bin capacity violations. ■

Remarks: The proposed approximation algorithm will serve as a subroutine of an algorithm for the cost-aware average model freshness maximization problem, by invoking the approximation algorithm at each time slot, and the solution obtained will be a lower bound on the optimal solution, at the expense of twice the bandwidth capacity violations on each AP in the worst case.

VI. Online Algorithm for the Cost-Aware, Average Model Freshness Maximization Problem

In this section, we first develop an efficient online algorithm for the cost-aware, average model freshness maximization problem, through exploring nontrivial trade-offs between the freshness improvement on all service models and the total updating cost spent to achieve the freshness. We then prove the correctness of the solution delivered by the proposed algorithm. Notice that for any object in a given monitoring period, there is no prior knowledge about its where about and the volume of the update data it generated since its uploading until that particular time slot.

A. Algorithm Overview

One solution for the cost-aware, average model freshness maximization problem is to identify a subset of objects to upload their update data at each time slot t, by applying the proposed approximation algorithm, `Algorithm 1`. However, this solution may violate the bandwidth capacity on each AP at that time slot. In the following, we develop an online algorithm, which delivers a feasible solution without any bandwidth capacity violation.

The key idea behind the proposed algorithm is an important observation. That is, at each time slot, each object uploading its update data to its DT will not necessarily reduce the value of the objective function $fresh(M,t)$. On the contrary, uploading the update data of some objects to their DTs will increase the value of the objective function, and such uploading should be avoided. The rest thus is to how identify objects that can upload their update data at time slot t to minimize the value of the objective function.

B. Online Algorithm

Given an object $v_i \in V$, we determine whether its update data will be uploaded at time slot t. If uploading its update data can bring the maximum reduction on the value of the objective function, the update data uploading will proceed; otherwise, no uploading from the object will take place at time slot t.

Let $\nabla_{cost}(v_i,t)$ be the amount of cost reduction on the staleness of all service models (or the freshness improvement on all service models) in the objective function, due to the update data uploading of object v_i at time slot t. Then,

$$\nabla_{cost}(v_i,t) = \alpha \cdot \sum_{m=1}^{M} \{r(DT_i,t) \mid v_i \in V_{m,t}\} = \alpha \cdot \left(\sum_{m=1}^{M} \frac{\{a^{t-t_0^{(i)}} - 1 \mid v_i \in V_{m,t}\}}{|M_m|} \right), \quad (21)$$

where the value of the DT state freshness of object v_i changes from $a^{t-t_0^{(i)}}$ at time slot $t-1$ to 1 when it is updated in the end of time slot t, while the total cost $\triangle_{cost}(v_i, AP_j, t)$ of various resources consumed by uploading the update data of object v_i to AP_j at time slot t is defined in (10).

Let $\delta(v_i, AP_j, t)$ is the *net cost reduction* toward the objective function by uploading the update data of object v_i to AP_j at time slot t. Then,

$$\delta(v_i, AP_j, t) = \nabla_{cost}(v_i,t) - \triangle_{cost}(v_i, AP_j, t). \quad (22)$$

It can be seen that if $\delta(v_i, AP_j, t) > 0$, the value of the objective function at the end of time slot t will be reduced; otherwise, the update data of object v_i will result in the increase on the value of the objective function. This observation leads to an online algorithm for the cost-aware average model freshness maximization problem, which proceeds iteratively.

Within each time slot t, the algorithm chooses objects to upload their update data greedily. It always chooses an object with the maximum net cost reduction at the moment, where object $v_i \in V$ is chosen if $\delta(v_i, AP_j, t)$ (> 0) is the maximum one among those not yet chosen objects. This choice procedure continues until either all objects are chosen or the bandwidth capacity on any AP runs out. The detailed online algorithm is given in `Algorithm 2`.

Algorithm 2: Online Algorithm for the Cost-Aware Average Model Freshness Maximization Problem.

Input: assume that the update data $vol(DT_i,t)$ of each object $v_i \in V$ is given, its AP $s(v_i)$, its DT cloudlet $h(v_i)$, and its model M_m cloudlet $h(M_m)$ with $1 \le m \le M$ at time slot t are given.

Output: choose a set $S_t \in V$ of objects to upload their update data to minimize the objective function at each time slot t with $1 \le t \le T$.

$S \leftarrow \emptyset$; /* the solution */
for $t \leftarrow 1$ to T **do**
  $S_t \leftarrow \emptyset$ /* the solution for objects to upload their update data at time slot t */;
  for each $v_i \in V$ **do**
    for each $AP_j \in N$ **do**
      $\delta(v_i, AP_j, t) \leftarrow \nabla_{cost}(v_i,t) - \triangle_{cost}(v_i, AP_j, t)$ by (21) and (10);
    end for
  end for ;
  Sort $\delta(\cdot,\cdot)$ in non-increasing order;
  Let $\mathbb{S}$ be the sorted sequence: $\delta_{i_1}, \delta_{i_2}, \ldots, \delta_{i_t}$ with $\delta_{i_k} > 0$ and $1 \le k \le i_t \le |V| \cdot |N|$;
  $k \leftarrow 1$;
  while $k \ne i_t + 1$ **do**
    Examine the first element $\delta_{i_k} = (v_i, AP_j, t)$ in sequence $\mathbb{S}$;
    if object $v_i \in \mathcal{C}(AP_j, t)$ and AP_j has sufficient residual bandwidth for object v_i **then**
      Allocate a fractional bandwidth $R_{ij}(t)$ of AP_j to object v_i;
      Set the transmission power PX_{ij} of object v_i and upload its update data $vol(DT_i,t)$ to cloudlet $s(v_i)$;
      Route the update data $vol(DT_i,t)$ of object v_i from cloudlet $s(v_i)$ to its DT hosting cloudlet $h(v_i)$;
      for each model M_m with object $v_i \in V(M_m)$ **do**
        Route the update data of object v_i from its DT cloudlet $h(v_i)$ to the home cloudlet $h(M_m)$ of model M_m;
      end for ;
      $\mathbb{S} \leftarrow \mathbb{S} \setminus \{\text{elements in } \mathbb{S} \text{ containing object } v_i\}$;
      $S_t \leftarrow S_t \cup \{\langle v_i, AP_j, t\rangle\}$;
    end if ;
    Remove the first element in $\mathbb{S}$;
    $k \leftarrow k+1$;
  end while
  $S \leftarrow S \cup S_t$;
end for
return S.

C. Algorithm Analysis

The rest is to show important properties of the proposed algorithm and analyze its time complexity. Intuitively, at each time slot t, if there is still bandwidth available for an object under an AP to upload its update data, the object should upload the data to refresh its DT state. However, this intuition is not always correct, as the DT state refreshment is achieved at the expense of various resource consumptions. Only if the refreshment on the DT state of an object outweighs the cost of resources consumed, then the data update of the object can proceed. We thus have the following lemma.

Lemma 2: Given a time slot $t \in \mathbb{T}$, assume that there is available bandwidth for an object v_i to upload its update data to AP_j, the update data uploading of object v_i can reduce the value of the optimization objective function if and only if $\delta(v_i, AP_j, t) \geq 0$.

Proof: As discussed, if $\delta(v_i, AP_j, t) < 0$, the cost of resource consumption for the update data uploading of v_i is larger than the gain on the state freshness of DT_i, which leads to the increase on the cost of the objective function. Otherwise, the state update of DT_i at time slot t will reduce the cost of the objective function, the update data uploading of v_i will take place by `Algorithm 2`. ■

For a given object v_i at time slot t, if $\delta(v_i, AP_j, t) < 0$, it will not upload its update data to the MEC network. To ensure its update data uploading to reduce the cost of the objective function in a future time slot, the nearest future time slot t' is given by the following lemma.

Lemma 3: Given an object $v_i \in V$ with $\delta(v_i, AP_j, t) < 0$ at time slot t for all $AP_j \in N$, it can upload its update data to $AP_{j'}$ at time slot t' with a smallest t' ($> t$) such that the cost of the objective function will be reduced, where t' is defined as follows.

$$t' = \operatorname*{argmin}_{t} \sum_{m=1}^{M} \frac{\{\alpha \cdot \left(a^{t-t_0^{(i)}} - 1\right) \mid v_i \in V(M_{m,t})\}}{|M_m|} - \Delta\left(v_i, AP_{j'}, t\right)) \geq 0. \tag{23}$$

Proof: If $c_{nl}(v_i, t) - c_{ul}(v_i, AP_j, t) < 0$, then object v_i uploads its update data at time slot t will increase the value of the objective function. It thus is not allowed to upload its update data at that time slot.

To ensure that the update data uploading of object v_i at time slot t' will lead to the reduction on the value of the objective function, it must have

$$c_{nl}(v_i, t') - c_{ul}(v_i, AP_{j'}, t') \geq 0, \quad \text{if } t' > t \geq t_0^{(i)}. \tag{24}$$

Ineq. (24) can be equivalently written as follows.

$$\sum_{m=1}^{M} \frac{\{\alpha \cdot a^{t'-t_0^{(i)}} \mid v_i \in V(M_{m,t})\}}{|M_m|} - \left(\sum_{m=1}^{M} \frac{\{\alpha \cdot 1 \mid v_i \in V(M_{m,t})\}}{|M_m|} + \Delta\left(v_i, AP_{j'}, t'\right)\right) \geq 0, \tag{25}$$

where the first term in the left hand side of Ineq. (25) is the value of the freshness of object v_i contributed to all models if its DT is not updated at time slot t' since its last update time slot $t_0^{(i)}$, while the second term is the total cost contribution of object v_i to the objective function when it uploads its update data at time slot t'.

It can be seen that if there is a minimum integer t' ($> t_0^{(i)}$) to ensure that Ineq. (25) holds, it must ensure that Ineq.(23) holds, too. The lemma then follows. ■

Theorem 3: Given an MEC network $G(N, E)$, a finite time horizon $\mathbb{T}$, and a set V of mobile objects with each having a DT placed in a cloudlet, there are M inference service models deployed in cloudlets, there is an online algorithm, `Algorithm 2`, for choosing objects to upload their update data to their DTs at each time slot $t \in \mathbb{T}$ such that the optimization objective defined in (14) is minimized, subject to the bandwidth capacity on each AP. The algorithm takes $O(|V| \cdot |N| \log(|V| \cdot |N|) + M \cdot |N| \log |E|)$ time at each time slot $t \in \mathbb{T}$.

Proof: We first show that the solution delivered by Algorithm 2 is feasible. For any given time slot t, the bandwidth capacity at each AP is not violated when allocating a fractional amount of bandwidth to objects by the proposed online algorithm. So, an object is able to upload its update data to an AP if and only if it is under the coverage of the AP and its uploading will reduce the value of the objective function. Thus, the solution for the given time horizon is feasible.

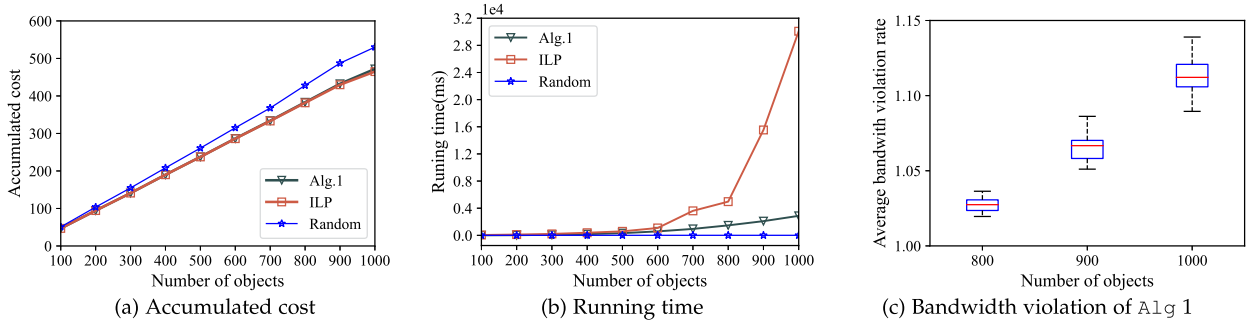
We then analyze the time complexity of algorithm, Algorithm 2, at each time slot $t \in \mathbb{T}$. The construction of an SPT tree for each service model M_m takes $O(|N| \log |E|)$ time as there are M such models. It thus takes $O(M \cdot |N| \log |E|)$ time in total. Also, it takes $O(|V| \cdot |N| \log(|V| \cdot |N|))$ time to sort objects in non-increasing order of their value $\delta(\cdot, \cdot)$, while it takes $O(|\mathcal{C}(AP_j, t)|)$ time to determine which objects to upload their update data at each time slot t with $1 \leq j \leq |N|$, i.e., $O(|V|)$ time. In total, it takes $O(|V| \log |V|)$ time for scheduling objects to upload their update data at each time slot. The algorithm thus takes $O(|V| \cdot |N| \log(|V| \cdot |N|) + M \cdot |N| \log |E|)$ time. ■

VII. Performance Evaluation

In this section, we evaluated the proposed algorithms for the cost-aware average model freshness maximization problem through simulations. We first evaluated the performance of the approximation algorithm for the special case of the problem. We then studied the performance of the online algorithm. We finally investigated the impacts of important parameters on the performance of the proposed algorithms.

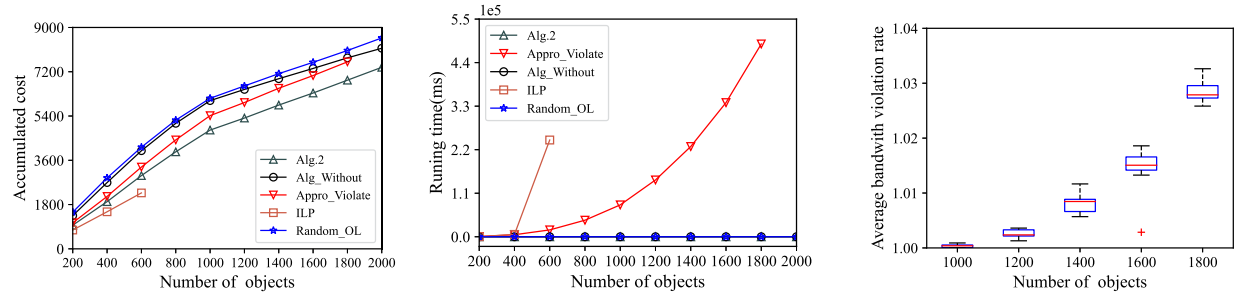
A. Experimental Environment Settings

Each MEC network instance is generated by adopting the tool GT-ITM [6], which consists of 100 APs, and there is a co-located cloudlet with each AP. Each cloudlet has computing capacity drawn randomly from 4,000 MHz to 14,000 MHz [22], and the cost η_2 per unit computing resource ranges from \$0.01 to \$0.03 per MHz [22]. Assume that there are 2,000 objects, and each object $v_i \in V$ has a digital twin DT_i deployed in one of the cloudlets in the MEC network. Note that the DT of an object



(a) Accumulated cost (b) Running time (c) Bandwidth violation of Alg 1

Fig. 2. Performance evaluation of the proposed approximation algorithm for the special cost-aware average model freshness maximization problem at time slot $t \in \mathbb{T}$, by varying the number $|V|$ of objects.



(a) Accumulated cost (b) Running time (c) Bandwidth violation of Appro_violate

Fig. 3. Performance of different algorithms for the cost-aware average model freshness maximization problem, by varying the number of objects $|V|$.

$v_i \in V$ is not necessarily deployed in the co-located cloudlet $s(v_i)$ of its AP_j when $v_i \in \mathcal{C}(AP_j, t)$ at time slot t, and could be deployed in another cloudlet. The location of DT_i of object v_i is fixed, despite that object v_i can move from one place to another place at different time slots. We assume that the bandwidth of each AP is 1,000Mpbs and set the bandwidth resource demand of each object update is between 100 Mbps and 200Mpbs randomly, and the cost η_1 of per time unit energy consumption on data uploading of each device is set from \$0.01 to \$0.03 randomly. The maximum and minimum transmit powers of an object are set at 23 dBm and 12dBm [3], respectively. We further assume that there are M (=400) service models deployed. For each model M_m with $1 \leq m \leq M$, recall that set $V(M_m)$ is the set of its data sources, which is a subset of V. The members of $V(M_m)$ are randomly drawn from V, and the candinality $|M_m|$ of M_m is drawn from 5 to 10 randomly. The default settings of parameters α and β are both 1 s.

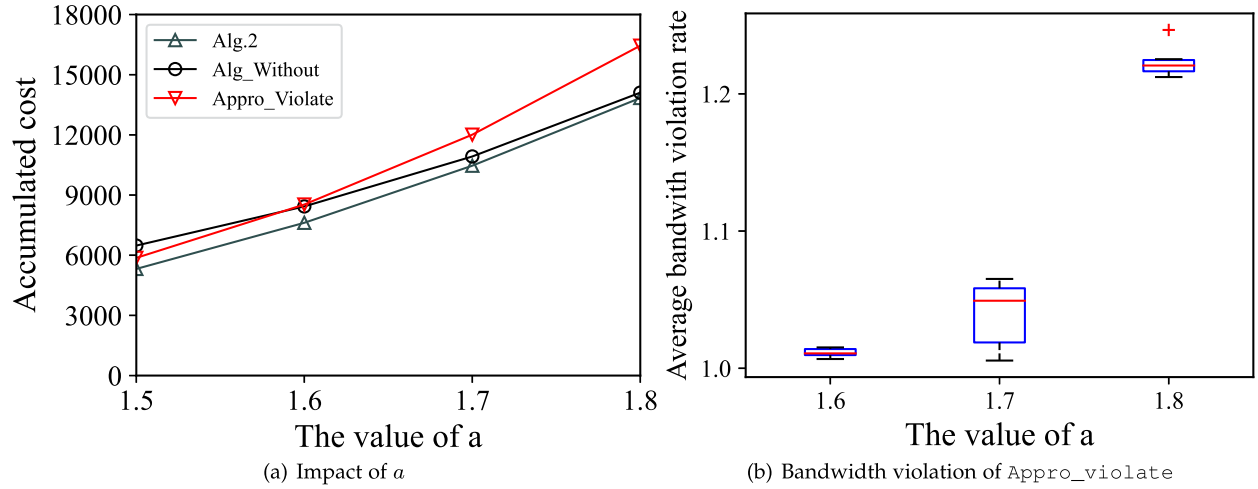
To evaluate the proposed approximation algorithm `Algorithm 1`, referred to as `Alg.1`, for the special cost-aware average model freshness maximization problem, we evaluated it against a lower bound on the optimal solution - the ILP solution of the problem at time slot t. We also proposed a heuristic, `Random`, for the problem, which randomly chooses objects to upload their update data at each AP until there is not enough bandwidth for any remaining objects at the AP for uploading.

To evaluate the online algorithm Algorithm 2 for the cost-aware average model freshness maximization problem, referred to as `Alg.2`, we considered three comparison algorithms: one is similar to the proposed algorithm but allows objects with $\delta(\cdot,\cdot) < 0$ to upload their update data if there is still the residual bandwidth on their APs, which is referred to as algorithm `Alg_Without`. Another is to apply the approximation algorithm `Appro_violate` at each time slot $t \in \mathbb{T}$, for which the bandwidth capacity on each AP may be violated. And the last one is algorithm `Random_OL`, which applies algorithm `Random` at each time slot $t \in \mathbb{T}$. Notice that the lower bound on the optimal solution of the problem is the ILP solution (14) for the offline version of the problem, referred to as `ILP`.

The value in each figure is the result by averaging the results based on 30 different MEC instances with the same network size. Simulations are performed by a desktop with a 3.60 GHz Intel i7 octa-core CPU and 16 GB RAM. Unless otherwise specified, the above-mentioned parameters are adopted by default.

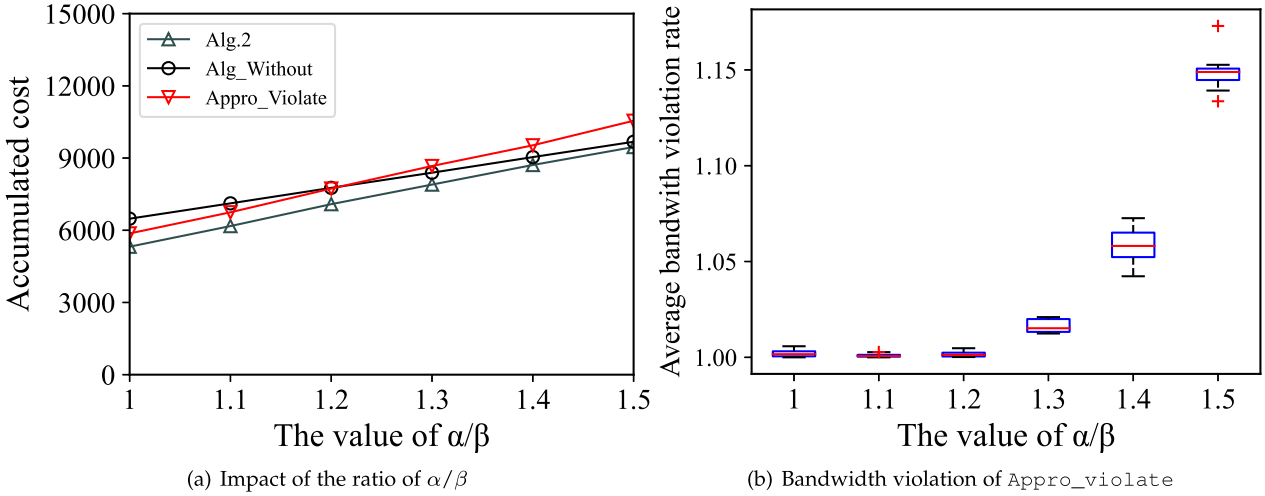
B. Performance Evaluation of the Approximation Algorithm

We first evaluated the proposed approximation algorithm Algorithm 1 against the optimal solution delivered by the ILP of the problem at time slot t. Assume that the freshness value of each object at time slot t is randomly drawn from 1 to a^4 and the value of a is set at 1.5. It can be seen from Fig. 2(a) that the accumulated cost in the solution delivered by `Alg 1` is around 1.3% times larger than that of the lower bound and is 18.5% times lower than `Random`, when there are 1,000 objects. Fig. 2(b) is the running time of the proposed algorithm and the ILP solution, and Fig. 2(c) is the bandwidth capacity violation on each AP. It can be seen that the average bandwidth violation rate is 11.3% when there are 1,000 objects.



(a) Impact of a (b) Bandwidth violation of `Appro_violate`

Fig. 4. Impact of parameter a on the performance of Alg.2 for the cost-aware average model freshness maximization problem.



(a) Impact of the ratio of α/β (b) Bandwidth violation of `Appro_violate`

Fig. 5. Impacts of the ratio of α to β on the performance of Alg.2 for the cost-aware average model freshness maximization problem.

C. Performance Evaluation of the Online Algorithm

We then evaluated the proposed online algorithm Algorithm 2 against `Alg_Without`, `Appro_violate`, and the lower bound `ILP` on the optimal solution of the problem. It can be seen from Fig. 3(a) when there are 400 objects, the accumulated cost by Algorithm 2 is larger than 27.2% of that by the ILP. When the number of objects is between 1,000 and 2,000, the accumulated cost by `Alg_Without` is 29.2% larger than that by Algorithm 2. Among all comparison algorithms, the performance of algorithm `Random` is the worst one. With the growth on the number of objects, the performance gap among different algorithms becomes smaller due to the fact that the bandwidth capacity of each AP will be exhausted at each time slot. When there are 1,800 objects, the accumulated cost by Algorithm 2 is less than 10.1% that of the solution by `Appro_violate`. The accumulated cost by `Alg_Without` is 13.2% larger than that by Algorithm 2. Fig. 3(b) plots the running time curves of the comparison algorithms. Fig. 3(c) depicts the curves of bandwidth capacity violation ratios of APs by algorithm `Appro_violate`, it can be seen that the ratio of the bandwidth capacity violation on each AP is no greater than 3.26% of its capacity.

D. Impacts of Parameters on Algorithm Performance

We finally investigated the impacts of important parameters on performance of different algorithms for the cost-aware average model freshness maximization problem as follows.

Impact of parameter of a: We studied the value of a on the performance of different algorithms by setting it at 1.5, 1.6, 1.7, and 1.8, respectively, considering that there are 1,200 objects and T is set at 7. With the increase on the value of a, it can be seen from Fig. 4(a) that the performance gap between Algorithm 2 and `Appro_Without` becomes smaller, due to the fact that the bandwidth capacity on each AP will be exhausted. Also, their performance gap becomes narrower with the increase on the number of objects, and the total cost by `Appro_violate`

is larger than that by Alg 2. With the growth of a, it can also be seen from Fig. 4(b) that the ratio of bandwidth capacity violation becomes large, the performance gap between Algorithm 2 and Appro_violate becomes larger, and the accumulated cost by Appro_violate is 19.5% Algorithm 2 larger than that by when $a = 1.8$.

Impact of the ratio of α to β: We evaluated the impact of the ratio of α to β on the performance of different algorithms by setting its value at 1.0, 1.1, 1.2, 1.3, 1.4 and 1.5, respectively, assuming that there are 1,200 objects and T is set at 12.

With the growth on the value of the ratio α/β, it can be seen from Fig. 5(a) that the performance gap between Algorithm 2 and Alg_Without becomes smaller, since the DT freshness of each object will be out-of-the-date very quickly, and the DT state of the object needs to update as soon as possible. Alg_Without always updates the DT states of objects in the very first few time slots so that the accumulated cost by the algorithm almost linearly grows, and the performance gap between Algorithm 2 and Appro_violate is smaller first and then larger, and finally the accumulated cost by Appro_violate is larger than that by Algorithm 2. It can also be seen from Fig. 5(b) that the ratio of bandwidth capacity violation becomes larger, and the accumulated cost by Appro_violate is 12.6% larger than that by Algorithm 2 when $\alpha/\beta = 1.5$.

VIII. Conclusion

In this article, we studied a novel cost-aware, average model freshness maximization problem of model-driven services in a DT-empowered edge computing network, through exploring non-trivial trade-offs between the accumulative freshness of all service models and the total cost to achieve the freshness, which is a fundamental question that balances between service accuracy of service models and the associated cost to maintain the accuracy. We first introduced the state freshness concept for both DTs of objects and service models that are built upon DTs, and showed the NP-hardness of the problem. We then formulated an integer linear program solution to the offline version of the problem, and devised an approximate solution to a special case of the problem when the monitoring period consists only of a single time slot. We third developed an online algorithm for the problem through DT state updating scheduling, and thereby service model updating at each time slot. We finally evaluated the performance of the proposed algorithms through experimental simulations. Simulation results demonstrate that the proposed algorithms are promising.

The potential research built upon this study is to accelerate model training by shortening the model training time, while maintaining their service accuracy at the acceptable level, thereby prolonging their user service time. To this end, we will explore various options of utilizing different computing resources including CPUs, GPUs and other AI accelerators for model training. We will also consider a joint optimization of service model refreshments and user service satisfaction on services provided by the service models while meeting their service delay requirements, for which we will focus on DT and service model placements, by incorporating the mobility of both objects and users.

Acknowledgments

The authors appreciate 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] J. Chen, H. Xing, Z. Xiao, L. Xu, and T. Tao, "A DRL agent for jointly optimizing computation offloading and resource allocation in MEC," *IEEE Internet Things J.*, vol. 8, no. 24, pp. 17508–17524, Dec. 2021.

[2] L. Corneo, C. Rohner, and P. Gunningberg, "Age of information-aware scheduling for timely and scalable Internet of Things applications," in *Proc. IEEE Conf. Comput. Commun.*, 2019, pp. 2476–2484.

[3] R. Dong, C. She, W. Hardjawana, Y. Li, and B. Vucetic, "Deep learning for hybrid 5G services in mobile edge computing systems: Learn from a digital twin," *IEEE Trans. Wireless Commun.*, vol. 18, no. 10, pp. 4692–4707, Oct. 2019.

[4] B. Fan, Y. Wu, Z. He, Y. Chen, T. Q. S. Quek, and C.-Z. Xu, "Digital twin empowered mobile edge computing for intelligent vehicular lane-changing," *IEEE Netw.*, vol. 35, no. 6, pp. 194–201, Nov./Dec. 2021.

[5] D. Gupta, S. S. Moni, and A. S. Tosun, "Integration of digital twin and federated learning for securing vehicular Internet of Things," in *Proc. Int. Conf. Res. Adaptive Convergent Syst.*, 2023, pp. 1–8.

[6] GT- ITM, 2019. [Online]. Available: http://www.cc.gatech.edu/projects/gtitm/

[7] Y. Han et al., "A dynamic hierarchical framework for IoT-assisted digital twin synchronization in the metaverse," *IEEE Internet Things J.*, vol. 10, no. 1, pp. 268–284, Jan. 2023.

[8] J. Li et al., "Wait for fresh data? digital twin empowered IoT services in edge computing," in *Proc. 20th Int. Conf. Mobile Ad Hoc Smart Syst.*, 2023, pp. 397–405.

[9] J. Li et al., "AoI-aware user service satisfaction enhancement in digital twin-empowered edge computing," *IEEE/ACM Trans. Netw.*, to be published, doi: 10.1109/TNET.2023.3324704.

[10] J. Li et al., "Maximizing user service satisfaction for delay-sensitive IoT applications in edge computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 5, pp. 1199–1212, May 2022.

[11] M. Li, J. Gao, C. Zhou, X. Shen, and W. Zhuang, "Adaptive mobile VR content delivery for industrial 5.0," in *Proc. 1st Workshop Digit. Twin Edge AI Ind. IoT*, 2022, pp. 1–6.

[12] 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.

[13] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Communication-efficient federated learning and permissioned blockchain for digital twin edge networks," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2276–2288, 2021.

[14] Y. Lu, S. Maharjan, and Y. Zhang, "Adaptive edge association for wireless digital twin networks in 6G," *IEEE Internet Things J.*, vol. 8, no. 22, pp. 16219–16230, Nov. 2021.

[15] L. U. Khan, W. Saad, D. Niyato, Z. Han, and C. S. Hong, "Digital-twin-enabled 6G: Vision, architectural trends, and future directions," *IEEE Commun. Mag.*, vol. 60, no. 1, pp. 74–80, Jan. 2022.

[16] 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.

[17] R. Minerva, G. M. Lee, and N. Crespi, "Digital twin in the IoT context: A survey on technical features, scenarios, and architectural models," in *Proc. IEEE*, vol. 108, no. 10, pp. 1785–1824, Oct. 2020.

[18] D. Shomys and E. Tardos, "An approximation algorithm for the generalized assignment problem," *Math. Program.*, vol. 62, pp. 461–474, 1993.

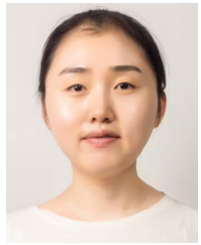
[19] W. Sun, H. Zhang, R. Wang, and Y. Zhang, "Reducing offloading latency for digital twin edge networks in 6G," *IEEEE Trans. Veh. Technol.*, vol. 69, no. 10, pp. 12240–12251, Oct. 2020.

[20] C. Wang, Z. Cai, and Y. Li, "Sustainable blockchain-based digital twin management architecture for IoT devices," *IEEE Internet Things J.*, vol. 10, no. 8, pp. 6535–6548, Apr. 2023.
[21] Z. Wang et al., "Mobility digital twin: Concept, architecture, case study, and future challenges," *IEEE Internet Things J.*, vol. 9, no. 18, pp. 17452–17467, Sep. 2022.
[22] Z. Xu, W. Liang, M. Huang, M. Jia, S. Guo, and A. Galis, "Efficient NFV-enabled multicasting in SDNs," *IEEE Trans. Commun.*, vol. 67, no. 3, pp. 2052–2070, Mar. 2019.
[23] W. Yang, W. Xiang, Y. Yang, and P. Cheng, "Optimizing federated learning with deep reinforcement learning for digital twin empowered industrial IoT," *IEEE Trans. Ind. Inform.*, vol. 19, no. 2, pp. 1884–1893, Feb. 2023.
[24] B. Zhu, K. Chi, J. Liu, K. Yu, and S. Mumtaz, "Efficient offloading for minimizing task computation delay of NOMA-based multiaccess edge computing," *IEEE Trans. Commun.*, vol. 70, no. 5, pp. 3186–3203, May 2022.

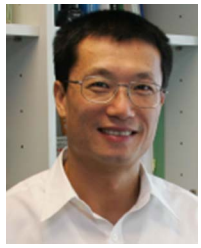
Xiyuan Liang receiced the BSc degree in computer science from the Hefei University of Technology, China, in 2020, and the MSc degree in computer science from the University of Science and Technology of China, in 2023. His research interests include cloud and edge computing, severless computing, distributed systems, and performance evaluation of distributed resource allocation and optimization.

Weifa Liang (Senior Member, IEEE) received the BSc degree in computer science from Wuhan University, China, in 1984, the ME degree in computer science from the University of Science and Technology of China, in 1989, and the PhD degree in computer science from Australian National University, in 1998. He is a full professor with the Department of Computer Science at 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 of *IEEE Transactions on Communications*.

Zichuan Xu (Member, IEEE) received the BSc and ME degrees in computer science from the Dalian University of Technology in China, in 2008 and 2011, and the PhD degree from the Australian National University, in 2016. 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.



Yuncan Zhang (Member, IEEE) received the BE degree from the Dalian University of Technology, Dalian, China, in 2013, the ME degree from the University of Science and Technology of China, Hefei, China, in 2016, and the PhD degree from Kyoto University, Kyoto, Japan, in 2021. She now is a postdoc with the Department of Computer Science, 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 network survivability.



Xiaohua Jia (Fellow, IEEE) received the BSc and MEng 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 *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 general co-chair IEEE ICDCS 2023.