






AoI-Aware Inference Services in Edge Computing via Digital Twin Network Slicing

Yuncan Zhang , *Member, IEEE*, Weifa Liang , *Senior Member, IEEE*, Zichuan Xu , *Member, IEEE*, Wenzheng Xu , *Member, IEEE*, and Min Chen , *Fellow, IEEE*

Abstract—The advance of Digital Twin (DT) technology sheds light on seamless cyber-physical integration with the Industry 4.0 initiative. Through continuous synchronization with their physical objects, DTs can power inference service models for analysis, emulation, optimization, and prediction on physical objects. With the proliferation of DTs, Digital Twin Network (DTN) slicing is emerging as a new paradigm of service providers for differential quality of service provisioning, where each DTN is a virtual network that consists of a set of inference service models with source data from a group of DTs, and the inference service models provide users with differential quality of services. Mobile Edge Computing (MEC) as a new computing paradigm shifts the computing power towards the edge of core networks, which is appropriate for delay-sensitive inference services. In this paper we consider Age of Information (AoI)-aware inference service provisioning in an MEC network through DTN slicing requests, where the accuracy of inference services provided by each DTN slice is determined by the Expected Age of Information (EAoI) of its inference model. Specifically, we first introduce a novel AoI-aware inference service framework of DTN slicing requests. We then formulate the expected cost minimization problem by jointly placing DT and inference service model instances, and develop efficient algorithms for the problem, based on the proposed framework. We also consider dynamic DTN slicing request admissions where requests arrive one by one without the knowledge of future arrivals, for which we devise an online algorithm with a provable competitive ratio for dynamic request admissions, assuming that DTs of all objects have been placed already. Finally, we evaluate the performance of the proposed algorithms through simulations. Simulation results demonstrate that the proposed algorithms are promising, and the

proposed online algorithm improves the number of admitted requests by more than 6% than its counterpart.

Index Terms—Digital twin network slicing, mobile edge computing, the expected age of information (EAoI), inference service models, cost modeling, DT and inference service instance placements.

I. INTRODUCTION

DRIVEN by the advancements in low latency and high-speed B5G/6G networks, the massive number of IoT devices has been connected to the network that realize diverse services such as smart city, healthcare, and autonomous vehicles. The Digital Twin (DT) technology facilitates seamless connection and communication between objects (IoT devices) and their virtual mirrors – digital twins. Through DT modeling, a virtual representation of an object is created to record its historical data, emulate its future behaviors, and provide inference services on objects through data analytics, artificial intelligence and machine learning [5]. DT technology realizes the co-evolution between the virtual space and the physical space. By leveraging the DT technology, the state of each object can be constantly monitored, and its sensory data can be uploaded and merged in its DT, and the data then can be used to facilitate service provisioning such as behaviour predictions of the object in future.

Digital Twin Network (DTN) built upon DTs is a new paradigm of inference service provisioning, where each DTN consists of one or multiple DTs of objects as its components [36]. The information geographically dispersed at different locations by objects can be synchronized with their DTs in a DTN and then aggregated at the service center for inference services (machine learning models based services). To fulfill personalized service demands, network slicing technology has been adopted to accommodate diverse vertical applications and services on a shared infrastructure. Each slice is an independent virtual DTN that provides required service functionalities with guaranteed service quality [3]. By taking advantage of the benefits of DT and network slicing, in this paper we investigate the inference service provisioning via DTN slicing, which utilizes the continuously updated DT data for model retraining and the flexible resource management and isolation for varied user requests.

Inference service provisioning via DTN slicing is essentially different from conventional service provisioning in MEC networks. Traditional services are implemented by a combination of a set of stateless service functions, any request with the same

Manuscript received 22 March 2024; revised 13 June 2024; accepted 19 July 2024. Date of publication 2 August 2024; date of current version 30 December 2024. The work of Yuncan Zhang and Weifa Liang was supported in part by Hong Kong Research Grants Council (HK RGC) under CityU HK under Grant 7005845, Grant 8730094, Grant 9043510, and Grant 9380137. The work of Zichuan Xu was partially supported in part by the National Natural Science Foundation of China under Grant 62172068, and in part by the Shandong Provincial Natural Science Foundation under Grant ZR2023LZH013, and the work of Wenzheng Xu was supported in part by NSFC under Grant 62272328, and in part by Sichuan Science and Technology Program under Grant 24NSFJQ0152, and the work of Min Chen was supported in part by the National Natural Science Foundation of China under Grant 62276109. (*Corresponding author: Weifa Liang.*)

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

Zichuan Xu is with the School of Software, Dalian University of Technology, Dalian 116620, China (e-mail: z.xu@dlut.edu.cn).

Wenzheng Xu is with the College of Computer Science, Sichuan University, Chengdu 610000, China (e-mail: wenzheng.xu@scu.edu.cn).

Min Chen is with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510640, China, and also with Pazhou Lab, Guangzhou 510330, China (e-mail: minchen@ieee.org).

Digital Object Identifier 10.1109/TSC.2024.3436705

input through a service chain will lead to the same result. In contrast, inference services are machine learning based services such as Deep Neural Networks (DNNs), in which service models need to be trained continuously, using the update data of model parameters, and inference results of an inference model for a service request issued at different times are different. One such an inference model example was given in [5], where an anomaly detection model of vehicles is trained by leveraging the DTs of vehicles, traffic lights, local weathers and sensing data at roadside in vehicular IoT environments. By incorporating the update data from these sources for the model training, the accuracy of the model is significantly enhanced, thereby improving efficiency of autonomous driving and mitigating fatal vehicle accidents.

In this paper, we deal with AoI-aware inference services in Mobile Edge Computing (MEC) networks via DTN slicing requests. As objects are movable in the network, their update data must be uploaded to their DTs, and then used for retraining their service models to meet accuracy requirements of the inference models. However, how to utilize AoIs of these DT data to enhance the service accuracy of inference models poses great challenges. First, the mobility of objects heavily impacts on their DT placements, because it impacts on the expected AoIs of DTs and accuracy of inference services that utilize the DT data for their model training. Second, considering limited computing capacity on each cloudlet, how to determine the DT placement of each mobile object while meeting the AoI requirement of the DT is challenging. Third, where to place the service home center for each DTN slicing request such that the service cost of the DTN slice is minimized is challenging. Finally, how to develop a request admission strategy such that the number of DTN slicing requests admitted is maximized, subject to computing resource capacity on each cloudlet? The rest of this paper will address the mentioned challenges.

The novelty of the work in this paper is proposing a novel AoI-aware inference service framework in an MEC network for DTN slicing requests. Two novel AoI-aware inference service provisioning problems are investigated, and cost modelling and efficient algorithms for the problems are developed.

The main contributions of this paper are as follows.

- We provide an AoI-aware service framework in a DT-empowered MEC network for DTN slicing requests.
- We formulate an Integer Linear Program (ILP) solution for the expected cost minimization problem of DTN slicing when the problem size is small. Otherwise, we develop efficient algorithms for the problem through placing DTs for objects and home service centers for DTN slicing requests.
- We devise an online algorithm with a provable competitive ratio for dynamic DTN request admissions, assuming that DTs of all objects have been placed.
- We validate the effectiveness of the proposed framework and evaluate the performance of proposed algorithms through simulations. Simulation results show that the proposed algorithms are promising, and the proposed online algorithm improves the number of admitted requests by more than 6% than its counterpart.

The remainder of the paper is organized as follows. Section II surveys the related work. Section III introduces the AoI-aware inference service framework of the DTN slicing. Section IV presents problem definitions. Section V provides an ILP solution and devises heuristic algorithms for the expected cost minimization problem. Section VI develops an online algorithm for the dynamic admissions of DTN slicing requests. Section VII provides simulation results, and Section VIII concludes the paper.

II. RELATED WORK

Network slicing technology has been intensively studied for diverse networks to separate limited resources for different services. For example, Esmat et al. [3] presented a resilient network slicing design to cope with failures and guarantee service continuity in satellite-terrestrial edge computing networks. Hossain and Ansari [6] studied the impact of the numerology schemes on a sliced radio access network, and devised algorithms for maximizing the system throughput of delay-sensitive services. Jošilo et al. [8] studied 6G resource allocation by jointly considering dynamic radio and computing resource management within and across different slices. Li et al. [14] considered the home service placement in MEC networks through network slicing for different IoT applications. Prasad et al. [27] presented a slice admission control and resource allocation framework in 5G networks with the aim to maximize the revenue. Shen et al. [29] considered bandwidth resource slicing-based task offloading in space-air-ground integrated vehicular networks. Zheng et al. [45] developed an approach to slice based resource allocation that relies on a constrained resource allocation game. These mentioned network slicing approaches however cannot handle continuous data synchronizations between objects and DTs in the network.

There are investigations of adopting the DT technology to enhance network performance. DT is an emerging concept that is gaining attention in various industries [23]. Bellavista et al. [1] designed the application-driven digital twin networking middleware to support the twofold objective of simplifying the interaction with heterogeneous distributed industrial devices and of dynamically managing network resources in distributed industrial environments. Hui et al. [7] designed the DT of each vehicle and developed a DT-enabled collaborative and distributed autonomous driving scheme. Li et al. [12] studied the reliability-aware SFC placement in MEC with leveraging DTs to predict the reliability of service function instances. Lin et al. [19] developed a congestion control scheme to ensure the stability of long-term DT services, by using the Lyapunov optimization with the aim to maximize the profit. Ren et al. [28] presented a congestion control scheme for DT edge networks and developed Deep Reinforcement Learning (DRL) method for performance prediction of the physical network. Vaezi et al. [34] proposed algorithms for the DT placement problem to minimize the maximum response delay of requests while meeting AoI requirements of requests. Zhao et al. [44] introduced a hierarchical routing strategy in a DT-assisted vehicle network to enable various services for vehicle users.

There are several recent efforts on AoI-aware DT-assisted service provisioning in MEC networks. For instance, Shu et al. [31] addressed the DT-assisted resource management for energy dispatching and control model training under the constraint of a long-term AoI, and proposed a DT-assisted federated learning scheme for the problem. Yi et al. [40] developed an online DT-empowered content resale mechanism with the aim to maximize the utility in AoI-aware edge caching networks. Tong et al. [33] studied the semantic-aware efficient sampling policy for remote state estimation in a DT-empowered smart factory with multiple wireless sensing devices and an edge server. They formulated an optimization problem to minimize the long-term age of incorrect information of remote state estimation. Li et al. [10], [16] devised approximation and online algorithms for dynamic DT placements in an MEC network to cope with the mobility of objects while ensuring low AoI services. They also developed algorithms for IoT application queries in DT-assisted MEC networks by developing performance-guaranteed approximation and online algorithms [11]. Liang et al. [18] investigated the relationship between the freshness of a service model and AoIs of DT data of its training source data. They devised an efficient algorithm for minimizing the total cost that consists of DT updating by their objects and the service cost of users requesting for service models. Zhang et al. [41] leveraged DTs to improve 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 the dynamic setting. Zhang et al. [42] studied the DT placement and migration problem under the mobility of objects and users. They developed an efficient algorithm for the problem that jointly optimizes the freshness of DT data and the service cost of users requesting DT services. Zhang et al. [43] considered mobility-aware service provisioning via DT replica placements. They adopted multiple DT replica placements for DT services, developed a randomized algorithm with high probability for a set of service requests, and an online algorithm for dynamic service requests. The aforementioned works focused on DT-assisted service provisioning without considering network slicing for different service providers.

Considering the advantages of network slicing and DT technologies in network resource management and optimization, a few studies combined both the techniques to improve network efficiency. For example, Karunaratna et al. [9] discussed the approach of realizing the Metaverse by emphasizing the importance of network slicing and edge computing. Li et al. [17] devised a learning-based algorithm for dynamic cooperative network slicing in a DTN to optimize the resource allocation and the long-term utility of operators. Naeem et al. [25] developed a DT-enabled deep distributional Q-network framework that constructs a digital mirror of the physical slicing-enabled network to simulate its complex environment and predict its dynamic characteristics. Tang et al. [32] designed a DTN-assisted network slicing framework by adopting a DRL approach. Wang et al. [35] proposed a DT of network slicing to investigate the composite traffic generated by slices, and utilized a graph neural network to predict the end-to-end metrics of network slices. These studies did not take into account the AoI of DT

data that plays an important role in the quality of DT-assisted services.

Unlike the aforementioned studies, in this paper we consider the AoI-aware inference services in an MEC network by different service providers with differential quality of services via DTN slicing. We introduce a novel AoI-aware inference service framework of DTN slicing, and propose efficient heuristic and online algorithms for DT and inference model placements to respond to static and dynamic DTN slicing requests, respectively.

III. AOI-AWARE INFERENCE SERVICE FRAMEWORK VIA DTN SLICING

In this section, we first introduce the system model, and then propose a novel AoI-aware inference service framework for differential quality of services in an MEC network via DTN slicing requests.

A. System Model

Consider an MEC network $G = (N, E)$ that consists of a set N of Access Points (APs) and a set E of links between APs. There is a co-located cloudlet with each AP and the communication delay between the AP and its co-located cloudlet is negligible. In this paper we use AP and its co-located cloudlet interchangeably if no confusion arises.

There is a set V of mobile objects under the coverage of APs in the network. Each object $v_i \in V$ has a DT, DT_i , which will be placed in a cloudlet of the MEC network. The data generated by object v_i is continuously uploaded from its current location, referred to as its gateway cloudlet, and synchronized with DT_i by transmitting the generated data from its gateway to the cloudlet hosting DT_i . Apparently, the DT location affects the freshness of the data transmitted from its object, and thus impacts the QoS of inference services whose source data come from the DT data of the object.

Given a set \mathbb{S} of DTN slicing requests, for each DTN slicing request $s_k \in \mathbb{S}$, it requests to establish a virtual digital twin network for its service, where there is a set $V_k \subseteq V$ of objects to provide the source data for request s_k , by updating their sensing data and states of their DTs continuously. Each s_k has a *home service center* located at a cloudlet to accommodate its inference services, and the inference accuracy of the trained model is determined by the freshness of its DT data. The freshness of the DT data of an object usually is measured by the AoI of its DT data, which is the *duration* of the update data from its generation to its first usage [39].

B. AoI-Aware Inference Service Framework

An AoI-aware inference service framework is proposed for an MEC network via DTN slicing requests, which consists of three layers, where Layer 0 is the physical layer that consists of the MEC network and a set V of mobile objects. Layer 1 is the DT network layer that is composed of DTs of mobile objects, and Layer 2 is the slice layer for inference service provisioning by different service providers. Under this framework, each

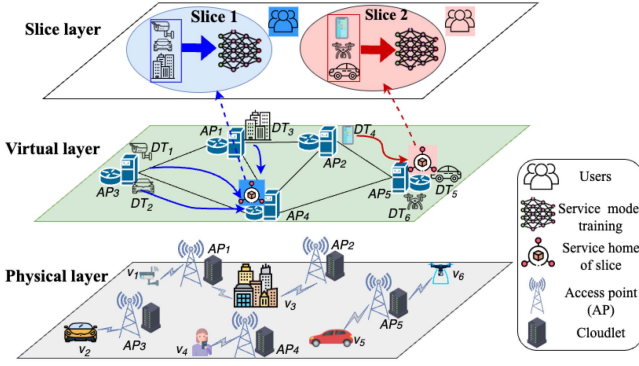


Fig. 1. An AoI-aware inference service framework through DTN slicing.

mobile object can upload its update data at its current location, the uploaded update data is then forwarded to its DT for storage and processing. The processed data then is transferred to the home service centers of the object if needed. When receiving all update data from its source DTs, a home service center of a DTN then makes use of the DT data to retrain its inference model for improved service accuracy.

Fig. 1 is an illustrative example. There are two DTN slicing requests for inference service provisioning, where DTN slice 1 provides an inference services for the Internet of Vehicles (IoVs). In the IoV case, each autonomous vehicle needs to decide whether its next move is safe, which is guided by the inference service model that needs updating data input/retraining from multiple sources in the vehicle surroundings, including traffic lights, roadside cameras, and nearby vehicles [5]. The requested DT data is aggregated at the home service center AP_4 of DTN slice 1, and then the data is used for its service model retraining.

IV. PROBLEM FORMULATIONS

In this section, we first formulate AoI-aware inference services in a DT-empowered MEC network as a joint optimization problem of DT placements and home service center placements for DTN slicing requests. We then define two optimization problems under static and dynamic DTN slicing request arrival settings.

A. Cost Modeling of Inference Models

We deal with inference service provisioning in a DT-assisted MEC network through DTN slicing requests, under the mobility assumption of objects and users. We use indices i and k to represent the indices of objects and DTN slices, and indices l , j , and m to represent the current location of an object, its DT placement location, and a home service center location of a service model, respectively. We assume that object $v_i \in V$ moves to a location $c_l \in N$ with probability $p_{i,l}$ and the volume of the update data $vol(v_i)$, where both $p_{i,l}$ and $vol(v_i)$ are assumed to be given, and $\sum_{l=1}^{|N|} p_{i,l} = 1, \forall v_i \in V$, by leveraging data mining and machine learning techniques [20], [24], [37]. Notice that the volume of the update data $vol(v_i)$ of object v_i can be estimated based on the historical traces of the DT data of v_i , by adopting different prediction methods such as the auto-regression method, or neural network models [20].

AoI of the update data generated by each object: The AoI of the update data of an object is the duration that consists of the uploading delay of the generated data from the object to its gateway, the transmission delay from the gateway to the DT location of the object, and the data processing delay at the DT of the object. Let $upload(v_i, c_l)$ denote the data uploading delay of object v_i at location c_l , and we have $upload(v_i, c_l) = \frac{vol(v_i)}{B_{i,l}}$, where $B_{i,l}$ is the uploading rate of object v_i to AP_l . If DT_i of object v_i is placed in cloudlet c_j , then the AoI of the DT data of object v_i is

$$AoI(v_i, c_j, c_l) = upload(v_i, c_l) + vol(v_i) \cdot d(path_{l,j}) + \frac{vol(v_i)}{f(DT_i) \cdot comp_i^{obj}}, \quad (1)$$

where $path_{l,j}$ is the shortest path in G between cloudlets c_l and c_j , $d(path_{l,j})$ is the transmission delay of unit data along path $path_{l,j}$, $f(DT_i)$ is the data processing rate per unit computing resource of DT_i , and $comp_i^{obj}$ is the amount of computing resource demanded to place DT_i of v_i . The value of $d(path_{l,j})$ is calculated by $d(path_{l,j}) = \sum_{e \in path_{l,j}} d_e$, where d_e is the transmission delay of unit data on link $e \in E$.

As each object is movable within the network, the AoI of the DT data of object v_i varies when object v_i sojourns at different locations. We here consider the *expected AoI* of the DT data of each object. When DT_i of object v_i is placed in cloudlet c_j , the expected AoI, $E AoI_{i,j}^{obj}$, of DT_i is

$$E AoI_{i,j}^{obj} = \sum_{l=1}^{|N|} p_{i,l} \cdot AoI(v_i, c_j, c_l). \quad (2)$$

AoI of the service model of each DTN slice: For DTN slicing request $s_k \in S$, its inference model updating (retraining) time at its home service center is

$$t_k^{comp} = \frac{\sum_{i=1}^{|V_k|} vol(DT_i)}{\rho_k \cdot comp_k^{ns}}, \quad (3)$$

where $vol(DT_i)$ is the data volume after processing at DT_i , ρ_k is the data processing rate per unit computing resource at the home service center of s_k , and $comp_k^{ns}$ is the required amount of computing resource of slice s_k for aggregating the update data from all its source DTs and retraining the inference model.

Let $x_{i,j}$ be a binary variable indicating whether the DT of object v_i is placed in cloudlet c_j ($x_{i,j} = 1$) or not ($x_{i,j} = 0$). If the home service center of s_k is located at cloudlet c_m , the EAoI of its inference service model is

$$E AoI_{k,m}^{ns} = \max_{v_i \in V_k} \left\{ \sum_{j=1}^{|N|} (E AoI_{i,j}^{obj} + vol(DT_i) d(path_{j,m})) \cdot x_{i,j} \right\} + t_k^{comp}. \quad (4)$$

Equation (4) indicates that the expected AoI of the inference model of s_k is the maximum one among the sum of the EAoI of each DT of its source data, the delay of the updated source data transferred from the DT to the service home center c_m , and the model updating delay, provided that the DTs of all objects in V_k

have been placed, i.e., for each v_i , only at most one $c_j \in N$ with $x_{i,j} = 1$ and the rest of cloudlets $c_{j'} \in N \setminus \{c_j\}$ with $x_{i,j'} = 0$.

DT synchronization cost of each object: The DT synchronization cost of each object consists of the uploading cost, transmission cost, and update data processing cost. Let ζ denote the cost of power consumption per unit time. Let PX_i be the transmission power of object v_i . The uploading cost of object v_i at location $c_l \in N$ then is $upload_cost(v_i, c_l) = \zeta \cdot PX_i \cdot \frac{vol(v_i)}{B_{i,l}}$. When DT_i is placed in cloudlet c_j , the DT synchronization cost of object v_i at location c_l is

$$\begin{aligned} cost(v_i, c_j, c_l) = & upload_cost(v_i, c_l) \\ & + vol(v_i) \cdot comm(path_{l,j}) + comp_i^{obj} \mu_j, \end{aligned} \quad (5)$$

where $comm(path_{l,j})$ is the transmission cost per unit data along path $path_{l,j}$ in G and μ_j is the cost of unit computing resource of cloudlet c_j . The value of $comm(path_{l,j})$ is calculated by $comm(path_{l,j}) = \sum_{e \in path_{l,j}} \xi_e$, where ξ_e is the transmission cost per unit data on link $e \in E$.

The expected service cost $Ecost_{i,j}^{obj}$ of v_i when its DT_i is placed at cloudlet c_j is

$$Ecost_{i,j}^{obj} = \sum_{l=1}^{|N|} p_{i,l} \cdot cost(v_i, c_j, c_l). \quad (6)$$

Service cost of each DTN slicing request: For each DTN slicing request $s_k \in \mathbb{S}$, the requested DT data needs to be transmitted from the DT location of each object in V_k to the home service center of s_k , and then the aggregated DT data is processed at the home service center. If the home service center of s_k is at location c_m , then the service cost of the inference model of s_k is

$$\begin{aligned} cost_{k,m}^{ns} = & \sum_{v_i \in V_k} \sum_{j=1}^{|N|} vol(DT_i) \cdot comm(path_{j,m}) \cdot x_{i,j} \\ & + comp_k^{ns} \cdot \mu_m. \end{aligned} \quad (7)$$

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

B. Problem Definitions

Considering the mobility of objects, we assume that for each object $v_i \in V$, the expected AoI of its DT, DT_i , is no greater than a given threshold θ_i^{obj} . The rationale behind this assumption is that the update frequency of each object is critical, which reflects the object's current state in a certain degree of accuracy. Furthermore, the location of a home service center of a DTN slice should not be far away from the locations of its requested DTs, in order to shorten the DT data transfer delay and reduce the service cost on data transmission. Thus, we impose a threshold θ_k^{ns} on the expected AoI of each service model of DTN slice $s_k \in \mathbb{S}$. The precise problem definitions are given as follows.

Definition 1: Given an MEC network $G = (N, E)$, a set V of objects with each object $v_i \in V$ having an EAoI requirement θ_i^{obj} on its DT, and a set \mathbb{S} of network slicing requests with each

$s_k \in \mathbb{S}$ having an EAoI requirement θ_k^{ns} . Each network slicing request s_k is represented by a tuple (V_k, θ_k^{ns}) where V_k is the set of requested objects. Assuming that the mobility profile of each object is given, the expected cost minimization problem of AoI-aware inference service provisioning in G is to identify the DT placement of each object and the home service center placement for the service model of each DTN slicing request while meeting their EAoI requirements, such that the total placement cost is minimized, subject to computing capacity on each cloudlet.

Since an MEC platform can provide inference services for different service providers and users, both service providers and users may change their interests over time. To accommodate new services with limited resources in the MEC network, for a given series of arriving DTN slicing requests, we need to decide which to be accepted or rejected immediately when it arrives. We thus consider dynamic DTN slicing request admissions as follows.

Definition 2: Given an MEC network $G = (N, E)$, a finite time horizon \mathbb{T} , and a set V of mobile objects, assuming that the DT of each object in V has been deployed in a cloudlet while meeting its EAoI requirement, assume that there is a sequence \mathbb{S} of DTN slicing requests arriving one by one without the knowledge of future arrivals, and each arrived DTN slicing request is represented by a tuple (V_k, θ_k^{ns}) . The dynamic DTN slicing request admission problem is to maximize the number of requests admitted for the given monitoring period \mathbb{T} through placing the home service centers of admitted requests to proper cloudlets, subject to computing capacity on each cloudlet.

Theorem 1: The expected cost minimization problem in an MEC network $G = (N, E)$ is NP-hard.

Proof: We show the NP-hardness of the expected cost minimization problem in an MEC network by a reduction from the well-known NP-hard problem - the minimum-cost generalized assignment problem (GAP). Given $|N|$ bins with computing capacity C'_m for each bin m , there are a set \mathbb{S} of items to be assigned to the bins, where if item k is assigned to bin m , it consumes the amount $comp_k^{ns}$ of computing resource of bin j and incurs cost $cost_{k,m}$. The minimum-cost GAP problem is to assign items to the bins such that the total cost of assigned items is minimized, subject to the computing capacity on each bin [30].

We consider a special case of the expected cost minimization problem where the DT placement of each object is given, and each DTN slicing request does not have any AoI requirement. We assume that each cloudlet (bin) n_m has computing capacity C'_m , which is the residual computing capacity of cloudlet n_m after DT placement. If the home service center of slicing request $s_k \in \mathbb{S}$ (item) is placed in cloudlet n_m , it consumes the amount $comp_k^{ns}$ of computing resource of bin m and incurs cost $cost_{k,m}^{ns} (= \sum_{v_i \in V_k} \sum_{j=1}^{|N|} vol(DT_i) \cdot comm(path_{j,m}) \cdot x_{i,j} + comp_k^{ns} \mu_m)$, where the first item of calculating $cost_{k,m}^{ns}$ is obtained based on the DT placement. We aim to find an optimal home service center placement for all slicing requests to minimize the total cost, subject to the computing capacity on each cloudlet. It can be seen that this special case of the expected cost minimization problem is equivalent to

TABLE I
TABLE OF NOTATIONS

Notation	Descriptions
$G = (N, E)$	An MEC network with a set N of APs and a set E of links.
$path_{l,j}$ and $d(path_{l,j}), comm(path_{l,j})$	The shortest path in G between cloudlets c_l and c_j . The transmission delay and the transmission cost of unit data along routing path $path_{l,j}$ in G , respectively.
V and \mathbb{S}	A set of objects and a set of DTN slicing requests.
\mathbb{T}	A finite time horizon.
V_k	A set of objects that provide the source data for request s_k , where $V_k \subseteq V$ and $s_k \in \mathbb{S}$.
$comp_i^{obj}$	The required amount of computing resource of placing DT_i of v_i .
$comp_k^{ns}$	The required amount of computing resource of slice s_k for aggregating the update data from all its source DTs and retraining the inference model.
$p_{i,l}$	The probability of object $v_i \in V$ moves to a location $c_l \in N$, where $\sum_{l=1}^{ N } p_{i,l} = 1, \forall v_i \in V$.
$vol(v_i)$	The volume of the update data of object $v_i \in V$.
$B_{i,l}$	The uploading rate of object v_i to AP_l .
$upload(v_i, c_l), upload_cost(v_i, c_l)$	The data uploading delay and uploading cost of object v_i to AP_l , respectively.
$AoI(v_i, c_j, c_l)$	The AoI of the DT data of object v_i when DT_i of object v_i is placed in cloudlet c_j .
$E AoI_{i,j}^{obj}$	The expected AoI of DT_i when DT_i of object v_i is placed in cloudlet c_j .
t_k^{comp}	The inference model updating time at the home service center of slicing request s_k .
$E AoI_{k,m}^{ns}$	The expected AoI of the inference service model of slicing request s_k when the home service center of s_k is located at cloudlet c_m .
$cost(v_i, c_j, c_l)$	The DT synchronization cost of object v_i at location c_l .
$E cost_{i,j}^{obj}$	The expected service cost of v_i when its DT_i is placed at cloudlet c_j .
$cost_{k,m}^{ns}$	The service cost of the inference model of s_k when the home service center of s_k is at location c_m .
$x_{i,j}$	A binary variable indicating whether the DT of object v_i is placed in cloudlet c_j ($x_{i,j} = 1$) or not ($x_{i,j} = 0$).
$y_{k,m}$	A binary variable, indicating whether DTN slicing request s_k chooses cloudlet c_m as its home service center ($y_{k,m} = 1$) or not ($y_{k,m} = 0$).
$\theta_i^{obj}, \theta_k^{ns}$	A given threshold for the EAoI of object v_i and service model of DTN slice s_k , respectively.
$\mathcal{N}(k)$	The feasible set of cloudlets for DTN slicing request s_k , where its EAoI requirement is fulfilled when its service home center is placed at any cloudlet in $\mathcal{N}(k)$.
$R_{\max}, \phi_{\max}, \phi_{\min}, a$	The parameters to guide resource reservation for future slicing requests.
m^*	The index m^* of the cloudlet that is chosen to host the home service center of incoming request s_k .
α_m, γ_k	The dual real variables of Constraint (30) and (31), respectively.

the minimum-cost GAP. As the minimum-cost GAP is NP-hard, the expected cost minimization problem is NP-hard, too.

V. ALGORITHMS FOR THE EXPECTED COST MINIMIZATION PROBLEM

In this section, we first formulate an ILP solution to the expected cost minimization problem. Since it takes prohibitively high time to find an exact solution of the ILP when the problem size is large, we then develop efficient algorithms for the problem.

A. ILP Formulation

Let $x_{i,j}$ be a binary variable indicating whether the DT of object v_i is placed in cloudlet c_j ($x_{i,j} = 1$) or not ($x_{i,j} = 0$). Let $y_{k,m}$ be a binary variable, indicating whether DTN slicing request s_k chooses cloudlet c_m as its home service center

($y_{k,m} = 1$) or not ($y_{k,m} = 0$). The expected cost minimization problem can be formulated as the following integer non-linear programming (INP).

$$\text{Minimize } \sum_{v_i \in V} \sum_{j=1}^{|N|} E cost_{i,j}^{obj} \cdot x_{i,j} + \sum_{s_k \in \mathbb{S}} \sum_{m=1}^{|N|} cost_{k,m}^{ns} \cdot y_{k,m}. \quad (8)$$

The optimization objective (8) can be rewritten as follows.

$$\begin{aligned} \text{Minimize } & \sum_{v_i \in V} \sum_{j=1}^{|N|} \sum_{l=1}^{|N|} p_{i,l} \cdot cost(v_i, c_j, c_l) \cdot x_{i,j} \\ & + \sum_{s_k \in \mathbb{S}} \sum_{m=1}^{|N|} \sum_{v_i \in V_k} \sum_{j=1}^{|N|} vol(DT_i) comm(path_{j,m}) x_{i,j} \cdot y_{k,m} \end{aligned}$$

$$+ \sum_{s_k \in \mathbb{S}} \sum_{m=1}^{|N|} comp_k^{ns} \cdot \mu_m \cdot y_{k,m} \quad (9)$$

subject to

$$(1), (2), (3), (4), (5), (6), (7)$$

$$\sum_{v_i \in V} comp_i^{obj} x_{i,l} + \sum_{s_k \in \mathbb{S}} comp_k^{ns} y_{k,l} \leq C_l, \forall c_l \in N \quad (10)$$

$$EAoI_{i,j}^{obj} \cdot x_{i,j} \leq \theta_i^{obj}, \forall v_i \in V, \forall c_j \in N \quad (11)$$

$$EAoI_{k,m}^{ns} \cdot y_{k,m} \leq \theta_k^{ns}, \forall s_k \in \mathbb{S}, \forall c_m \in N \quad (12)$$

$$\sum_{j=1}^{|N|} x_{i,j} \leq 1, \forall v_i \in V \quad (13)$$

$$\sum_{m=1}^{|N|} y_{k,m} \leq 1, \forall s_k \in \mathbb{S} \quad (14)$$

$$x_{i,j} \in \{0, 1\} \quad \forall v_i \in V, \forall c_j \in N \quad (15)$$

$$y_{k,m} \in \{0, 1\} \quad \forall s_k \in \mathbb{S}, \forall c_m \in N, \quad (16)$$

where the first term of the objective function (8) is the expected cost of DT placements, the second term is the cost of DT transfer from their locations to the home service centers, and the last term is the cost of service model retraining using the update data obtained from their DTs.

Constraint (11) ensures that the total computing resource demanded by all DTs and home service centers at each cloudlet is no greater than its capacity. For the convenience of expression, we use l to denote the cloudlet index in variables $x_{i,j}$ and $y_{k,m}$ instead of using c_j and c_m in (11). Constraints (12) and (13) ensure that the EAoI requirements of each DT and each DTN slicing request are met. Constraint (14) ensures that the DT of each object is placed in one cloudlet at most. Constraint (15) ensures that each DTN slicing request chooses at most one cloudlet as its home service center.

The optimization objective (8) becomes the following linear function, by introducing a new binary variable $z_{i,j,k,m}$.

$$\begin{aligned} Cost = & \sum_{v_i \in V} \sum_{j=1}^{|N|} \sum_{l=1}^{|N|} p_{i,l} \cdot cost(v_i, c_j, c_l) \cdot x_{i,j} \\ & + \sum_{s_k \in \mathbb{S}} \sum_{m=1}^{|N|} \sum_{v_i \in V_k} \sum_{j=1}^{|N|} vol(DT_i) \cdot comm(path_{j,m}) \cdot z_{i,j,k,m} \\ & + \sum_{s_k \in \mathbb{S}} \sum_{m=1}^{|N|} comp_k^{ns} \cdot \mu_m \cdot y_{k,m}, \end{aligned} \quad (17)$$

$$z_{i,j,k,m} \leq x_{i,j}, \forall v_i \in V, \forall s_k \in \mathbb{S}, \forall c_j, c_m \in N \quad (18)$$

$$z_{i,j,k,m} \leq y_{k,m}, \forall v_i \in V, \forall s_k \in \mathbb{S}, \forall c_j, c_m \in N \quad (19)$$

$$\begin{aligned} z_{i,j,k,m} & \geq x_{i,j} + y_{k,m} - 1, \\ & \forall v_i \in V, \forall s_k \in \mathbb{S}, \forall c_j, c_m \in N \end{aligned} \quad (20)$$

$$z_{i,j,k,m} \in \{0, 1\}, \forall v_i \in V, \forall s_k \in \mathbb{S}, \forall c_j, c_m \in N. \quad (21)$$

Constraint (13) can be equivalently rewritten as follows.

$$\begin{aligned} & \sum_{j=1}^{|N|} (EAoI_{i,j}^{obj} + vol(DT_i) \cdot d(path_{j,m})) \cdot z_{i,j,k,m} \\ & + t_k^{comp} \cdot y_{k,m} \leq \theta_k^{ns}, \forall s_k \in \mathbb{S}, \forall c_m \in N, \forall v_i \in V_k. \end{aligned} \quad (22)$$

An ILP formulation for the expected cost minimization problem is given as follows.

Minimize $Cost$

subject to

$$(1), (2), (3), (5), (6), (11), (12), (14), (15), (16), (17), \\ (19), (20), (21), (18), (23), \text{ and } (22).$$

Although the ILP formulation provides an exact solution to the expected cost minimization problem, its running time is prohibitively high when the problem size is large. In the following, we develop efficient algorithms for the problem when the problem size is large. We decompose the expected cost minimization problem into two sub-problems: *the DT placement problem* that places the DT of each object to a cloudlet while meeting the EAoI requirement of its DT such that the total cost of DT placements is minimized; and *the home service center placement problem* that places the home service center for each admitted DTN slicing request such that the total placement cost of home service centers is minimized, assuming that DT placements have been done already.

B. Approximation Algorithm for the DT Placement Problem With Bounded Resource Violations

We here devise an approximation algorithm for the DT placement problem. Since the problem is NP-hard, by a simple reduction from the knapsack problem, we approach the problem by reducing it to the minimum-cost GAP.

There are $|N|$ bins with capacity C_j for each bin $c_j \in N$, and there are $|V|$ items for bin packing. If $EAoI_{i,j}^{obj} \leq \theta_i^{obj}$ and $comp_i^{obj} \leq C_j$, then item $v_i \in V$ can be packed into bin $c_j \in N$ with cost $cost(i, j) = Ecost_{i,j}^{obj}$ and weight $w(i, j) = comp_i^{obj}$. Otherwise, $cost(i, j) = \infty$ and $w(i, j) = comp_i^{obj}$.

There is an approximation algorithm for the minimum-cost GAP, which delivers an optimal solution, at the expense of twice the computing capacity violations, by applying the approximation algorithm due to Shomys and Tardos in [30].

The detailed algorithm for the DT placement problem is given in Algorithm 1.

C. Heuristic Algorithm for the DT Placement Problem Without Resource Violations

Although Algorithm 1 delivers an optimal solution to the problem at the expense of computing resource violations, we here develop an algorithm for the DT placement problem without computing resource violations, by reducing it to a series

Algorithm 1: Approximation Algorithm for the DT Placement Problem.

Input: An MEC network $G = (N, E)$, a set V of mobile objects with a given mobility profile and the EAoI requirement for each object $v_i \in V$.

Output: Minimize the DT placement cost while meeting their EAoI thresholds.

- 1: **for** each object $v_i \in V$ **do**
- 2: **for** each cloudlet $c_j \in N$ **do**
- 3: Calculate $EAoI_{i,j}^{obj}$ and $Ecost_{i,j}^{obj}$;
- 4: **if** $EAoI_{i,j}^{obj} \leq \theta_i^{obj}$ and $comp_i^{obj} \leq C_j$ **then**
- 5: $cost(i, j) \leftarrow Ecost_{i,j}^{obj}$; $weight(i, j) \leftarrow comp_i^{obj}$;
- 6: **else**
- 7: $cost(i, j) \leftarrow \infty$; $weight(i, j) \leftarrow comp_i^{obj}$;
- 8: **end if**
- 9: **end for**
- 10: **end for**
- 11: Find an approximate solution for the minimum-cost GAP by applying the approximation algorithm in [30];
- 12: **return** An approximate solution for DT placements of objects.

of minimum-cost maximum matching problems in auxiliary bipartite graphs.

We construct an auxiliary bipartite graph $G_{XY} = (X, Y, E_{XY})$, where X is the set of cloudlets and Y is the set of objects. If DT_i of object v_i can be hosted by cloudlet c_j while meeting its EAoI threshold θ_i^{obj} (i.e., the computing resource demand $comp_i^{obj} \leq C_j^{res}$ and $EAoI_{i,j}^{obj} \leq \theta_i^{obj}$), then there is an edge in E_{XY} between $c_j \in X$ and $v_i \in Y$ with weight $Ecost_{i,j}^{obj}$, where C_j^{res} is the available computing capacity of cloudlet $c_j \in N$.

The algorithm iteratively proceeds. $G_{XY}^{(1)} = G_{XY}$ and $q = 1$ initially. Within iteration q , we first find a minimum-cost maximum matching M_q in bipartite graph $G_{XY}^{(q)}$. Then, for each edge $(x_i, y_j) \in M_q$, we perform $C_j^{res} = C_j^{res} - comp_i^{obj}$ and remove all matched objects in M_q from the bipartite graph as their DTs will be placed into their matched cloudlets. We then construct the next bipartite graph $G_{XY}^{(q+1)}$. This procedure continues until there are no edges in $G_{XY}^{(q+1)}$.

Let Q be the number of iterations when the algorithm terminates. The union $\cup_{q=1}^Q M_q$ of all found maximum matches forms a feasible solution to the DT placement problem. The detailed algorithm is presented in Algorithm 2.

D. Algorithms for the Home Service Center Placement Problem With and Without Resource Violations

Having DTs of objects in V been deployed in cloudlets, let $h(DT_i)$ be the index of the cloudlet hosting DT_i of object $v_i \in V$. Then, the actual EAoI of DT_i of object $v_i \in V$ is $EAoI_{i,h(DT_i)}^{obj}$ by (2).

Let c_m be a potential location of the home service center of DTN slicing request $s_k \in \mathbb{S}$. Recall that t_k^{comp} is retraining time

Algorithm 2: Matching Algorithm for the DT Placement Problem Without Resource Violation.

Input: An MEC network $G = (N, E)$, a set V of mobile objects with given mobility profiles and the EAoI requirement.

Output: Minimize the DT placement cost while meeting the EAoI threshold requirement of each DT.

- 1: **for** each cloudlet $c_j \in N$ **do**
- 2: **for** each object $v_i \in V$ **do**
- 3: Calculate $EAoI_{i,j}^{obj}$ and $Ecost_{i,j}^{obj}$;
- 4: **if** $EAoI_{i,j}^{obj} \leq \theta_i^{obj}$ **then**
- 5: $cost(i, j) \leftarrow Ecost_{i,j}^{obj}$;
- 6: **end if**;
- 7: **end for**;
- 8: $C_j^{res} \leftarrow C_j$;
- 9: **end for**
- 10: Construct $G_{XY} = (X, Y, E_{XY})$ with the cost $Ecost_{i,j}^{obj}$ of edge (v_i, c_j) ;
- 11: $G_{XY}^{(1)} \leftarrow G_{XY}$; $q \leftarrow 1$;
- 12: $M \leftarrow \emptyset$; /* the set of matched edges */
- 13: **while** $E_{XY} \neq \emptyset$ **do**
- 14: Find a minimum-cost maximum matching M_q in $G_{XY}^{(q)}$;
- 15: **for** each edge $(v_i, c_j) \in M_q$ **do**
- 16: DT_i of object v_i will be placed to cloudlet c_j ;
- 17: $C_j^{res} \leftarrow C_j^{res} - comp_i^{obj}$;
- 18: Construct $G_{XY}^{(q+1)}$ by removing all object nodes in M_q ;
- 19: **end for**;
- 20: $M \leftarrow M \cup M_q$; $q \leftarrow q + 1$;
- 21: **end while**
- 22: **return** A feasible solution for DT placements of objects.

of the inference model at its home service center $c_m \in N$, which is defined in (3). The expected AoI of the home service center of s_k when it is placed at cloudlet c_m is

$$EAoI_{k,m}^{ns} = \max_{v_i \in V_k} \{EAoI_{i,h(DT_i)}^{obj} + vol(DT_i) \cdot d(path_{h(DT_i),m})\} + t_k^{comp}.$$

We now deal with the home service center placement problem by reducing it to the minimum-cost GAP as follows. If the home service center of $s_k \in \mathbb{S}$ can be placed to cloudlet $c_m \in N$ (i.e., $EAoI_{k,m}^{ns} \leq \theta_k^{ns}$ and $comp_k^{ns} \leq C_m$), then the home placement cost of s_k is $cost_{k,m}^{ns}$ with weight $w_{k,m} = comp_k^{ns}$; otherwise, its home placement cost is $cost_{k,m}^{ns} = \infty$ with weight $weight_{k,m} = comp_k^{ns}$. An approximate solution to the minimum-cost GAP can be found at the expense of twice the bin capacity violations, which in turn returns an approximate solution to the home service center placement problem at the expense of twice the computing capacity violations.

The approximation algorithm for the home service center placement problem is similar to Algorithm 1, omitted. To avoid computing resource violations, a heuristic algorithm based on

Algorithm 3: Algorithm for the Expected Cost Minimization Problem.

Input: An MEC network $G = (N, E)$, a set V of mobile objects with each $v_i \in V$ having a mobility profile and an EAoI threshold θ_i^{obj} , and a set \mathbb{S} of DTN slicing requests with each $s_k \in \mathbb{S}$ having an EAoI threshold θ_k^{ns} .

Output: Minimize the total cost while meeting EAoI threshold requirements of both DTs and home service centers.

- 1: Find a scheduling for their DT placements of all objects in V , by applying Algorithms 1 or 2;
 - 2: Find a scheduling for the home service center placements of admitted DTN slicing requests in \mathbb{S} , by applying the similar algorithms as Algorithms 1 or 2;
 - 3: **return** A solution to the expected cost minimization problem.
-

the minimum-cost maximum matching similar to Algorithm 2 can be developed, omitted.

E. Algorithms for the Expected Cost Minimization Problem

Assuming that both DTs and home service centers have been placed, we are ready to propose a heuristic algorithm, Algorithm 3, for the expected cost minimization problem as follows.

F. Algorithm Analysis

The rest is to analyze the time complexity of the proposed algorithms.

Lemma 1: Given an MEC network $G = (N, E)$, a set V of objects, assuming that the mobility profile $p_{i,l}$ of each object v_i for each potential sojourn location $c_l \in N$ is given, and its EAoI threshold requirement θ_i^{obj} is given too, there is an approximation algorithm, Algorithm 1, for the DT placement problem, which delivers an optimal solution at the expense of twice the computing capacity violation, and its time complexity is $O((|N| + |V|)^3)$. And there is another heuristic algorithm, Algorithm 2, delivering a feasible solution without any resource violations, and its time complexity is $O((|V| + |N|)^3 \cdot |V|)$.

Proof: Following Algorithm 1, if item DT_i is deployed to cloudlet c_j in the solution, then the EAoI of DT_i is no greater than θ_i^{obj} ; otherwise DT_i will not be placed to c_j . Notice that the approximate solution may violate the computing capacity on cloudlets. However, such the violation is no greater than twice the computing capacity on any cloudlet by [30].

We show that Algorithm 2 delivers a feasible solution as follows. When the algorithm terminates, there is no edge in the auxiliary bipartite graph, which implies (i) $Y = \emptyset$; or (ii) no available resource for non-matched objects. For case (i), the DT of each object has been deployed, and for case (ii), $Y \neq \emptyset$, either there is no available computing resource in cloudlets for nodes in Y , or there are adequate resource in a cloudlet but placing the DT of any object in Y to the cloudlet will violate the EAoI requirement of the DT. Thus, the solution delivered by Algorithm 2 is a feasible solution, because there is no computing

resource violation, and the EAoI of each placed DT meets its EAoI threshold requirement.

We now analyze the time complexity of Algorithms 1 and 2. The calculation of $EaOI_{i,j}^{obj}$ for all possible pairs of i and j takes $O(|V| \cdot |N|)$ time. It takes $O((|V| + |N|)^3)$ time for the minimum-cost GAP by applying the approximation algorithm in [30]. Thus, Algorithm 1 takes $O((|N| + |V|)^3)$ time. The time complexity of Algorithm 2 lies in the construction of the weighted bipartite graph G_{XY} that takes $O(|V| \cdot |N|)$ time. Finding a minimum-weighted maximum matching in G_{XY} takes $O((|V| + |N|)^3)$ time. There are $|V| - 1$ iterations in Algorithm 2, because at least one matched edge can be found at each iteration. Thus, Algorithm 2 takes $O((|N| + |V|)^3 \cdot |V|)$ time. \square

Lemma 2: Given an MEC network $G = (N, E)$, the DTs of all objects in set V have been placed with each $v_i \in V$ meeting its EAoI threshold requirement θ_i^{obj} , a set \mathbb{S} of DTN slicing requests with each $s_k \in \mathbb{S}$ having an EAoI threshold requirement θ_k^{ns} , there is an approximation algorithm for the home service center placement problem, which delivers an optimal solution, at the expense of twice the computing capacity violation on any cloudlet. Its time complexity is $O((|N| + |\mathbb{S}|)^3 + |V| \cdot |N| \cdot |\mathbb{S}|)$. Alternatively, there is another heuristic algorithm similar to Algorithm 2 for the home service center problem, delivering a feasible solution without any resource violations, and its time complexity is $O((|\mathbb{S}| + |N|)^3 \cdot |\mathbb{S}|)$.

Proof: We first show that the home service center placement of each DTN slicing request meets its EAoI requirement. It can be seen that each DTN slicing request s_k can be admitted only if its EAoI is no greater than a given EAoI threshold θ_k^{ns} , otherwise it will be rejected, because its cost will be infinity in either the approximation algorithm or the minimum-cost maximum matching algorithm.

We then analyze the time complexity of algorithms for the home service center placement problem. For the algorithm based on the minimum-cost GAP, it takes $O(|V| \cdot |N|^3 \cdot |\mathbb{S}|)$ time for the calculation of $EaOI_{k,m}^{ns}$ and $cost_{k,m}^{ns}$ for all possible pairs of k and m , and it takes $O((|N| + |\mathbb{S}|)^3)$ time for finding an approximate solution for the minimum-cost GAP [30]. Thus, the approximation algorithm similar to Algorithm 1 takes $O((|N| + |\mathbb{S}|)^3 + |V| \cdot |N|^3 \cdot |\mathbb{S}|)$ time. For the algorithm based on the minimum-cost maximum matching, it takes $O(|N| \cdot |\mathbb{S}|)$ time for the construction of the bipartite graph, $O((|\mathbb{S}| + |N|)^3)$ time for finding a minimum-cost maximum matching in G_{XY} , and up to $|\mathbb{S}| - 1$ iterations in the proposed algorithm. The algorithm thus takes $O((|N| + |\mathbb{S}|)^3 \cdot |\mathbb{S}|)$ time. \square

Theorem 2: Given an MEC network $G = (N, E)$, a set V of objects, and a set \mathbb{S} of DTN slicing requests, each object $v_i \in V$ has a mobility profile $p_{i,l}$ and an EAoI threshold θ_i^{obj} with $1 \leq l \leq |N|$, and each DTN slicing request $s_k \in \mathbb{S}$ has an EAoI threshold θ_k^{ns} . There is an algorithm, Algorithm 3 for the expected cost minimization problem, and its time complexity is analyzed as follows. If it is based on the minimum-cost GAP, the time complexity of Algorithm 3 is $O(|V| + |N|)^3 + (|N| + |\mathbb{S}|)^3 + |V| \cdot |N|^3 \cdot |\mathbb{S}|)$, otherwise, its time complexity is $O((|V| + |N|)^3 \cdot |V| + (|N| + |\mathbb{S}|)^3 \cdot |\mathbb{S}|)$ when it is based on the minimum-cost maximum matching.

Proof: The time complexity of Algorithm 3 can be derived from Lemmas 1 and 2 directly, omitted. \square

VI. ALGORITHM FOR THE DYNAMIC DTN SLICING REQUEST ADMISSION PROBLEM

In this section, we study the dynamic DTN slicing request admission problem. We first formulate an ILP solution for its offline version. We then devise an online algorithm for the problem, by making use of the primal-dual dynamic updating technique [4].

A. ILP Formulation for the Offline Version of the Problem

The ILP for the offline version of the dynamic DTN slicing request admission problem is given as follows.

$$\text{LP1: Maximize } \sum_{s_k \in \mathbb{S}} \sum_{m=1}^{|N|} y_{k,m} \quad (23)$$

subject to

$$\sum_{s_k \in \mathbb{S}} \text{comp}_k^{ns} \cdot y_{k,m} \leq C_m^{res}, \forall c_m \in N \quad (24)$$

$$EAoI_{k,m}^{ns} \cdot y_{k,m} \leq \theta_k^{ns}, \forall s_k \in \mathbb{S}, \forall c_m \in N \quad (25)$$

$$\sum_{m=1}^{|N|} y_{k,m} \leq 1, \forall s_k \in \mathbb{S} \quad (26)$$

$$y_{k,m} \in \{0, 1\}, \forall s_k \in \mathbb{S}, \forall c_m \in N, \quad (27)$$

where $EAoI_{k,m}^{ns}$ is given when DTs of objects in V_k have been placed, and C_m^{res} is the residual computing resource capacity on cloudlet c_m after DT placements.

The optimization objective (23) is to maximize the number of admitted DTN slicing requests for the finite time horizon \mathbb{T} . The actual EAoI of DTN slicing request s_k when its service home center is located at cloudlet c_m , $EAoI_{k,m}^{ns}$, is calculated, by using (1), (2), (3), and (4). Constraint (24) ensures that the capacity of each cloudlet will not be violated. Constraint (25) guarantees that the EAoI requirement of a slicing request is met if the request is admitted. Constraint (26) ensures that the home service center of each admitted slicing request will be placed in one cloudlet only.

B. Online Algorithm

In the following, we devise an online algorithm for the dynamic DTN slicing request admission problem. The basic idea is to utilize the duality of the offline ILP formulation to find a dynamic updating strategy for tackling the online problem through the primal-dual dynamic updating technique [4].

Recall that LP1 is the offline version of the dynamic slicing request admission problem, where each DTN slicing request for the finite time horizon is given in advance. Denote by $\mathcal{N}(k)$ the feasible set of cloudlets for DTN slicing request s_k , where its EAoI requirement is fulfilled when its service home center is placed at any cloudlet in $\mathcal{N}(k)$, i.e., $\mathcal{N}(k) = \{c_m \mid c_m \in N, EAoI_{k,m}^{ns} \leq \theta_k^{ns}\}$. Let LP2 be the relaxed linear program of

LP1, by replacing Constraint (25) with the set $\mathcal{N}(k)$ of cloudlets satisfying the EAoI requirement θ_k^{ns} of request $s_k \in \mathbb{S}$. Then, LP2 is given as follows.

$$\text{LP2: Maximize } \sum_{s_k \in \mathbb{S}} \sum_{c_m \in \mathcal{N}(k)} \tilde{y}_{k,m} \quad (28)$$

subject to

$$\sum_{s_k \in \mathbb{S}} \text{comp}_k^{ns} \cdot \tilde{y}_{k,m} \leq C_m^{res}, \forall c_m \in N \quad (29)$$

$$\sum_{m=1}^{|N|} \tilde{y}_{k,m} \leq 1, \forall s_k \in \mathbb{S} \quad (30)$$

$$\tilde{y}_{k,m} \in [0, 1], \forall s_k \in \mathbb{S}, \forall c_m \in N. \quad (31)$$

By Constraint (30) and (31), the binary variable $y_{k,m}$ in LP1 is relaxed to a real variable $\tilde{y}_{k,m}$ between 0 and 1 in LP2, i.e., $0 \leq \tilde{y}_{k,m} \leq 1$. Denote by DP2 the dual of LP2 that is defined as follows.

$$\text{DP2: Minimize } \sum_{m=1}^{|N|} C_m^{res} \cdot \alpha_m + \sum_{k=1}^{|\mathbb{S}|} \gamma_k \quad (32)$$

subject to

$$\text{comp}_k^{ns} \cdot \alpha_m + \gamma_k \geq 1, \forall s_k \in \mathbb{S}, \forall c_m \in N \quad (33)$$

$$\alpha_m \geq 0, \gamma_k \geq 0, \forall s_k \in \mathbb{S}, \forall c_m \in N, \quad (34)$$

where α_m and γ_k are the dual real variables of Constraint (29) and (30), respectively.

Let R_{\max} be the ratio of the maximum amount of computing resource consumption of any slicing request to the residual computing capacity on any cloudlet $c_m \in N$. Let ϕ_{\max} , ϕ_{\min} , and a be the parameters to guide resource reservation for future slicing requests, where $\phi_{\max} = \max\{\frac{1}{\text{comp}_k^{ns}} \mid s_k \in \mathbb{S}\}$, $\phi_{\min} = \min\{\frac{1}{\text{comp}_k^{ns}} \mid s_k \in \mathbb{S}\}$, $R_{\max} = \max\{\frac{\text{comp}_k^{ns}}{C_m^{res}} \mid c_m \in N, s_k \in \mathbb{S}\}$, $a = (1 + R_{\max})^{\frac{1}{R_{\max}}}$, and C_m^t is the residual computing capacity on cloudlet c_m at time slot $t \in \mathbb{T}$, and $C_m^0 = C_m^{res}$ initially. Notice that comp_k^{ns} is given in advance.

The *admission control policy* for the online algorithm is as follows. For each incoming request $s_k \in \mathbb{S}$ at time slot $t \in \mathbb{T}$, there is a corresponding variable γ_k^t . If $\gamma_k^t > 0$, slicing request s_k is admitted; otherwise rejected. Once s_k is admitted, the index m^* of the cloudlet that is chosen to host the home service center of s_k is determined too, where $m^* = \arg \max_m \{1 - \text{comp}_k^{ns} \cdot \alpha_m \mid c_m \in \mathcal{N}(k)\}$. Then, all dual variables in DP2 related to the admission of request s_k will be updated by the following updating rules accordingly.

$$\gamma_k^t \leftarrow \max\{1 - \text{comp}_k^{ns} \cdot \alpha_m \mid c_m \in \mathcal{N}(k)\} \quad (35)$$

$$\alpha_{m^*}^t \leftarrow \alpha_{m^*}^{t-1} \cdot \left(1 + \frac{\text{comp}_k^{ns}}{C_{m^*}^t}\right) + \frac{\phi_{\max}}{a-1} \cdot \frac{\text{comp}_k^{ns}}{C_{m^*}^t}. \quad (36)$$

The detailed online algorithm for the dynamic DTN slicing request admission problem is presented in Algorithm 4.

Algorithm 4: Online Algorithm for Dynamic DTN Slicing Request Admissions at Time Slot t With $1 \leq t \leq T$.

Input: A sequence \mathbb{S} of user slicing requests, and the available computing resource capacity C_m^t on cloudlet c_m at t .

Output: Maximize the number of admitted slicing requests with meeting their EAoI requirements.

```

1: Initialize  $\alpha_m^t, \gamma_k^t \leftarrow 0, \forall k, m$ ;
2:  $A \leftarrow 0$ ;  $S_{admit} \leftarrow \emptyset$ ; /*  $A$  is the number of requests
   admitted and  $S_{admit}$  is the set of admitted requests */;
3: while there is an incoming request  $s_k$  do
4:    $\mathcal{N}(k) \leftarrow \emptyset$ ;
5:   for each cloudlet  $c_m \in N$  do
6:     if  $EAoI_{k,m}^{ns} \leq \theta_k$  then
7:        $\mathcal{N}(k) \leftarrow \mathcal{N}(k) \cup \{c_m\}$ ;
8:   end if
9:   end for
10:   $\gamma_k^t \leftarrow \max\{1 - comp_k^{ns} \cdot \alpha_m^t \mid c_m \in \mathcal{N}(k)\}$  by
    Rule (35);
11:   $m^* \leftarrow \arg \max_m \{1 - comp_k^{ns} \cdot \alpha_m^t \mid c_m \in \mathcal{N}(k)\}$ ;
12:  if  $\gamma_k^t > 0$  then
13:    Admit request  $s_k$ ,  $S_{admit} \leftarrow S_{admit} \cup \{s_k\}$ ;
14:     $A \leftarrow A + 1$ ;
15:     $\alpha_{m^*}^t \leftarrow \alpha_{m^*}^t \cdot (1 + \frac{comp_k^{ns}}{C_m^t}) + \frac{\phi_{\max}}{a-1} \cdot \frac{comp_k^{ns}}{C_m^t}$ , by
      Rule (36);
16:  end if
17: end while

```

C. Algorithm Analysis

The rest is to show the solution feasibility and analyze the competitive ratio of the online algorithm, Algorithm 4.

Lemma 3: Following the updating rules of (35) and (36), the dual feasibility of the dual variables is preserved.

Proof: Upon the arrival of slicing request s_k , all dual variables will be updated only when slicing request s_k is admitted. As all dual variables γ_k^t and α_m^t are initialized as 0s when $t = 0$, the value of the updated variable γ_k^t must be no less than 0, by the updating rule (35), and the updated value of α_m^t increases, by updating rules (36). Thus, for all the dual variables, the updating rules always keep them non-negative. Hence, we have shown that the dual constraint (34) is met. By updating rule (35), the dual constraint (33) is met too. The fractional solution of the dual linear program DP2 thus is a feasible solution. \square

Lemma 4: Let $\Delta P(t)$ and $\Delta D(t)$ be the value gains of the primary linear program LP2 and its dual DP2 by the online algorithm, Algorithm 4, to respond to an incoming request $s_k \in \mathbb{S}$ at time slot t , respectively. Then,

$$\Delta D(t) \leq \Delta P(t) \cdot \left(1 + \frac{1}{a-1} \cdot \frac{\phi_{\max}}{\phi_{\min}}\right). \quad (37)$$

Proof: If slicing request s_k is admitted and its service model is placed in cloudlet c_{m^*} , then dual variables γ_k^t and $\alpha_{m^*}^t$ are updated by rules (35) and (36), respectively. The objective value of the dual linear program, DP2, increases by $1 + \frac{\phi_{\max}}{a-1} \cdot comp_k^{ns}$,

which is derived as follows.

$$\begin{aligned}
\Delta D(t) &= \Delta \alpha_{m^*}^t \cdot C_{m^*}^t + \gamma_k^t \\
&= \left(\alpha_{m^*}^t \cdot \frac{comp_k^{ns}}{C_{m^*}^t} + \frac{\phi_{\max}}{a-1} \cdot \frac{comp_k^{ns}}{C_{m^*}^t} \right) \cdot C_{m^*}^t \\
&\quad + (1 - comp_k^{ns} \cdot \alpha_{m^*}^t) \\
&= 1 + \frac{\phi_{\max}}{a-1} \cdot comp_k^{ns}.
\end{aligned}$$

As $\Delta P(t) = 1$ (if request s_k is admitted at time slot t), we have

$$\frac{\Delta D(t)}{\Delta P(t)} = 1 + \frac{\phi_{\max}}{a-1} \cdot \frac{1}{\frac{1}{comp_k^{ns}}} \leq 1 + \frac{\phi_{\max}}{a-1} \cdot \frac{1}{\phi_{\min}}. \quad (38)$$

Let $\rho = 1 + \frac{1}{a-1} \cdot \frac{\phi_{\max}}{\phi_{\min}}$, we then have

$$\Delta D(t) \leq \Delta P(t) \cdot \rho. \quad (39)$$

The lemma then follows.

We now analyze computing resource capacity violations on each cloudlet $c_m \in N$ after admitting DTN slicing request s_k at time slot t .

Lemma 5: After the arrival of slicing request s_k , the dual variable α_m^t corresponding to computing capacity on cloudlet $c_m \in N$ satisfies

$$\alpha_m^t \geq \frac{\phi_{\max}}{a-1} \cdot \left(a \frac{\sum_k \sum_{m=1}^{|N|} y_{k,m} \cdot comp_k^{ns}}{C_m^t} - 1 \right). \quad (40)$$

Proof: We show the claim by induction. As all dual variables are set to 0 initially, Inequality (40) holds.

Assuming that Inequality (40) holds for all admitted slicing requests in \mathbb{S} with indices no greater than k . We now admit request $s_{k'} \in \mathbb{S}$ by deploying its service model to cloudlet c_m with $k < k'$. Let $\alpha_m^t(start)$ and $\alpha_m^t(end)$ be the values of α_m^t before and after the admission of request $s_{k'}$. We have

$$\begin{aligned}
\alpha_m^t(end) &= \alpha_m^t(start) \cdot \left(1 + \frac{comp_{k'}^{ns}}{C_m^t}\right) + \frac{\phi_{\max}}{a-1} \cdot \frac{comp_{k'}^{ns}}{C_m^t} \\
&\geq \frac{\phi_{\max}}{a-1} \cdot \left(a \frac{\sum_{k < k'} \sum_{m=1}^{|N|} y_{k,m} \cdot comp_k^{ns}}{C_m^t} - 1 \right) \cdot \left(1 + \frac{comp_{k'}^{ns}}{C_m^t}\right) \\
&\quad + \frac{\phi_{\max}}{a-1} \cdot \frac{comp_{k'}^{ns}}{C_m^t} \quad (41)
\end{aligned}$$

$$\begin{aligned}
&= \frac{\phi_{\max}}{a-1} \cdot \left(\left(1 + \frac{comp_{k'}^{ns}}{C_m^t}\right) \cdot a \frac{\sum_{k < k'} \sum_{m=1}^{|N|} y_{k,m} \cdot comp_k^{ns}}{C_m^t} - 1 \right) \\
&\quad (42)
\end{aligned}$$

$$\geq \frac{\phi_{\max}}{a-1} \cdot \left(a \frac{\sum_k \sum_{m=1}^{|N|} y_{k,m} \cdot comp_k^{ns}}{C_m^t} - 1 \right). \quad (43)$$

Inequality (41) holds by the induction assumption. Since $\frac{1}{x} \log(1+x)$ is a decreasing function with $x > 0$ and $R_{\max} \geq \frac{comp_{k'}^{ns}}{C_m^t}$, the inequality from (42) to (43) holds by the following

arguments.

$$\log a = \frac{1}{R_{\max}} \log(1 + R_{\max}) \leq \frac{C_m^t}{\text{comp}_{k'}^{ns}} \log \left(1 + \frac{\text{comp}_{k'}^{ns}}{C_m^t} \right). \quad (44)$$

Inequality (44) can be rewritten as follows.

$$\frac{\text{comp}_{k'}^{ns}}{C_m^t} \log a \leq \log \left(1 + \frac{\text{comp}_{k'}^{ns}}{C_m^t} \right). \quad (45)$$

Inequality (45) can be rewritten as follows.

$$a^{\frac{\text{comp}_{k'}^{ns}}{C_m^t}} \leq 1 + \frac{\text{comp}_{k'}^{ns}}{C_m^t}. \quad (46)$$

The lemma then follows. \square

Lemma 6: In the solution delivered by Algorithm 4, the computing capacity violation of the service model of s_k placed at any cloudlet $c_m \in N$ is upper bounded by a multiplicative factor of $1 + R_{\max}$, where $R_{\max} = \max\{\frac{\text{comp}_{k'}^{ns}}{C_m^t} \mid c_m \in N, s_k \in \mathcal{S}\}$.

Proof: We claim that the computing capacity violation on any cloudlet c_m is upper bounded by $\max\{\text{comp}_{k'}^{ns} \mid s_k \in \mathcal{S}\}$. By Lemma 5, if the computing capacity constraint (24) is violated (i.e., $\frac{\sum_k \sum_{m=1}^{|N|} y_{k,m} \cdot \text{comp}_{k'}^{ns}}{C_m^t} > 1$), then $\alpha_m^t \geq \phi_{\max}$. We thus have $\gamma_k^t = 1 - \text{comp}_{k'}^{ns} \cdot \alpha_m^t \leq 1 - \text{comp}_{k'}^{ns} \cdot \phi_{\max} \leq 0$, as $\text{comp}_{k'}^{ns} \cdot \phi_{\max} \geq 1$.

According to updating rules (35) and (36), both the variables related to the computing capacity on cloudlet c_m will not be updated, and for any later arriving request $s_{k'}$ that uses cloudlet c_m to accommodate its service model, it must have $y_{k',m} = 0$. Hence, Constraint (24) will be violated by at most once, and the computing capacity violation on any cloudlet $c_m \in N$ is upper bounded by $\max\{\text{comp}_{k'}^{ns} \mid s_k \in \mathcal{S}\}$, i.e., $\sum_{k=1}^{|\mathcal{S}|} y_{k,m} \cdot \text{comp}_{k'}^{ns} \leq C_m^t + \max\{\text{comp}_{k'}^{ns} \mid c_m \in N, s_k \in \mathcal{S}\}$. We have $\max\{\frac{C_m^t + \text{comp}_{k'}^{ns}}{C_m^t} \mid s_k \in \mathcal{S}, c_m \in N\} = 1 + R_{\max}$. The lemma then follows. \square

Theorem 3: Given an MEC network $G = (N, E)$, assume that DTs of all mobile objects in V have been deployed already and the expected AoI, $E\text{AoI}_i^{\text{obj}}$, of the DT, DT_i , for object $v_i \in V$ has been given, each DTN slicing request s_k has an EAoI requirement θ_k^{ns} . There is an online algorithm, Algorithm 4, with a competitive ratio of $\frac{1}{\rho}$ for the dynamic slicing request admission problem, at the expense of the computing capacity violation with no more than $1 + R_{\max}$. The running time of the online algorithm is $O(|N|)$ for the admission of a slicing request at each time slot t with $1 \leq t \leq T$, where $\rho = 1 + \frac{1}{a-1} \cdot \frac{\phi_{\max}}{\phi_{\min}}$.

Proof: It can be seen that the solution delivered by Algorithm 4 is feasible. As for each admitted slicing request s_k , its service model can be deployed into one cloudlet in set $\mathcal{N}(k)$, which meets its specified EAoI requirement, at the expense of computing capacity violation no greater than $1 + R_{\max}$ by Lemma 6.

Let OPT_1 and OPT_2 be the optimal solutions of LP1 and LP2, respectively. Since LP2 is the LP relaxation of LP1, $OPT_2 \geq OPT_1$. Denote by P and D the values of the feasible solutions of LP2 and DP2 delivered by Algorithm 4, respectively. By Lemma 4, we have that $D \leq P \cdot \rho$, and by the weak duality

property, we have $D \geq OPT_2$. There is

$$P \geq \frac{D}{\rho} \geq \frac{OPT_2}{\rho} \geq \frac{OPT_1}{\rho}.$$

The running time of Algorithm 4 is analyzed as follows. For each incoming request, the algorithm takes $O(|N|)$ time to determine whether there is a cloudlet to host its service model. If the request is admitted, the updating of the dual variables takes $O(1)$ time. The algorithm thus takes $O(|N|)$ time per request at each time slot t . \square

VII. PERFORMANCE EVALUATION

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

A. Environment Settings

We considered an MEC network that consists of 20 APs, which is generated by NetworkX [26]. Each cloudlet has computing capacity randomly drawn in the range of [3,000, 8,000] MHz [32], and the cost of computing resource per MHz is randomly drawn in the range of \$[0.005, 0.015] [38]. The transmission delay for transmitting a unit of data (one MB) along each link $e \in E$ is randomly drawn in [0.01, 0.05] millisecond [38], and the communication cost on link e is randomly drawn from \$0.0001 to \$0.0005 per MB [15]. We further assumed that the number of potential sojourn locations of an object is no greater than 20% of the number of APs [20]. The amount of computing resource demanded by a DT is randomly drawn in [100, 200] MHz [16]. The volume of the update data of each object is within [50, 100] MB [34], and the data volume after being processed at its DT is within [10, 50] MB. The transmission power P_i of each object $v_i \in V$ is randomly drawn from 10 dBm and 23 dBm [2]. The cost ζ of unit transmission power consumption per unit time is \$0.004. The uploading rate B_i of each object v_i is calculated by $B_i = W_l \log(1 + SNR)$, where W_l is the bandwidth of AP l and SNR is the signal-to-noise ratio. The bandwidth of AP l is randomly chosen in the range [20, 40] MHz, and the signal-to-noise ratio of each object is randomly chosen from 10dB to 30dB [13]. We assume that the number of DTs in each DTN slice request varies from 5 to 10, and the required amount of computing resource by the service home center of each DTN slice is within [300, 500] MHz [16]. The EAoI of each object is randomly drawn in [10, 20] millisecond [38] and the EAoI of each DTN slicing request is drawn from 1.1 to 1.3 times of its maximum object DT's threshold. The data processing rate of a DT or a service home center per MHz is randomly chosen in the range of [0.1, 0.5] MB per millisecond [14].

To evaluate Algorithm 3, a greedy algorithm Greedy is proposed, which chooses a cloudlet with sufficient residual computing resource within each iteration if the cloudlet can meet the EAoI requirement of the DTN slice and achieve the minimum cost. We refer to algorithms as Appro, Match, and Greedy by invoking Algorithms 1 and 2, and the greedy algorithm, respectively. We refer to the linear relaxation of the ILP as LP, and notice that the value of LP is a lower bound on the cost

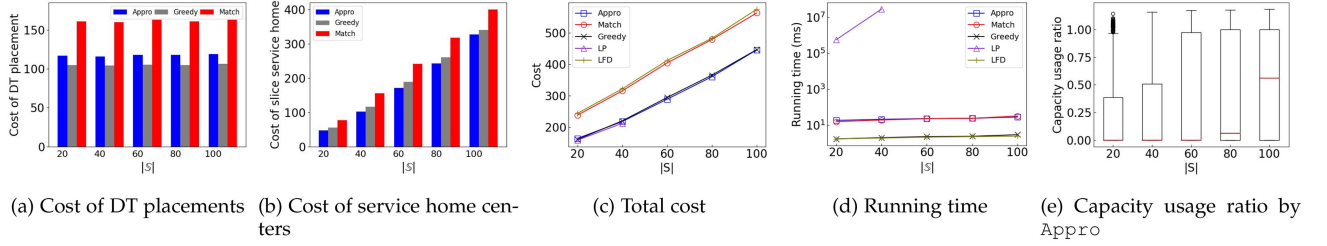


Fig. 2. Performance of algorithms for the expected cost minimization problem, by varying the number $|S|$ of slicing requests.

of the optimal solution to the problem. We also compare the performance of the proposed algorithms against algorithm LFD, which is a variant of an existing work [21], [22]. In LFD, services are deployed in non-increasing order of resource demands, and an unplaced service will be placed to the server with the largest remaining resource.

To evaluate Algorithm 4 for the dynamic DTN slicing request admission problem, two benchmarks No_Control and LP2 are used, where a DTN slicing request s_k is admitted by No_Control if there exists a cloudlet in $\mathcal{N}(k)$ meeting the computing resource requirement of s_k . The solution of LP2 is an upper bound on the optimal solution to the dynamic DTN slicing request admission problem.

The value in each figure is the average over 25 different MEC network topologies with the same size. The running time of each algorithm is based on a desktop with a 3.60 GHz Intel 8-Core i7 CPU and 16 GB RAM. Unless otherwise specified, these parameters are adopted by default.

B. Performance of Different Algorithms for the Expected Cost Minimization Problem

We evaluated the performance of different algorithms. Considering that there is capacity violation by Appro, we partition the computing capacity on each cloudlet into two parts that are allocated to the two sub-problems, using parameters σ and β respectively, where $\sigma \cdot C_j$ is reserved for DT placements, βC_j of each cloudlet is reserved for home service center placements, and σ and β are set at 0.4 and 0.6 by default, respectively.

We first studied the performance of Appro, Match, LP, Greedy, and LFD, by varying the number of DTN slicing requests $|S|$ while fixing the number of objects at 100. It can be seen from Fig. 2(c) and (d) that LP achieves the lowest cost with the longest running time. Particularly, when the number of DTN slicing requests is larger than 40, it is not achievable to find a solution of LP in the reasonable amount of running time. From Fig. 2(a) and (b), it can be seen that Greedy has a lower DT placement cost than those of Appro and Match, while Appro achieves the lowest service home cost among the three comparison algorithms. This is because during the DT placements, the capacity of each cloudlet suffices, and Greedy can choose the cloudlet with the lowest DT placement cost that meets the EAoI requirement of each object. After the capacities of cloudlets have been consumed for DT placements, Greedy may choose a cloudlet with a large service home cost for DTN slicing requests

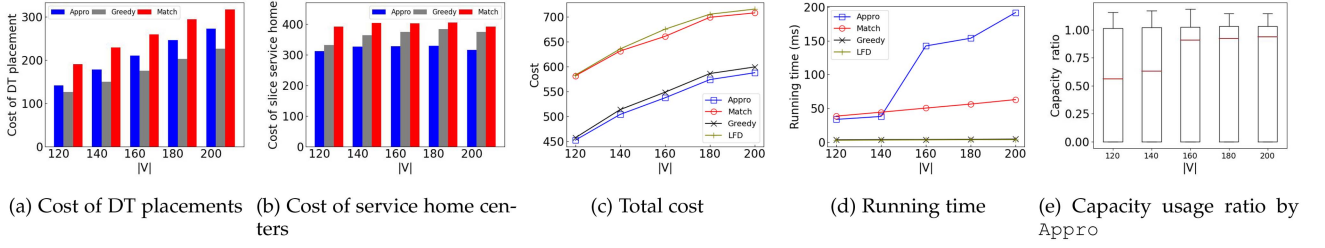
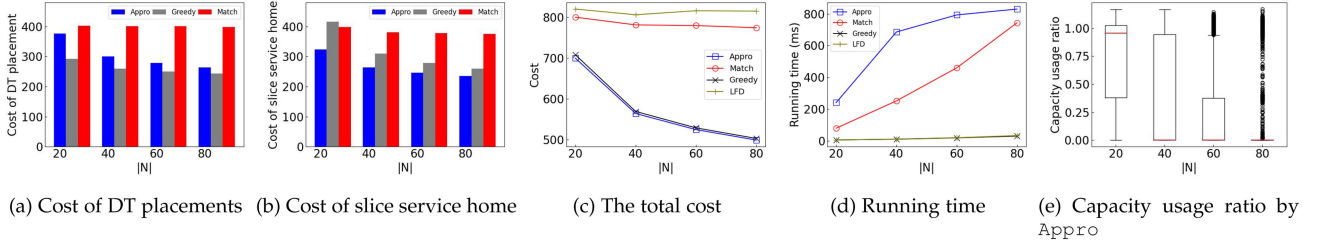
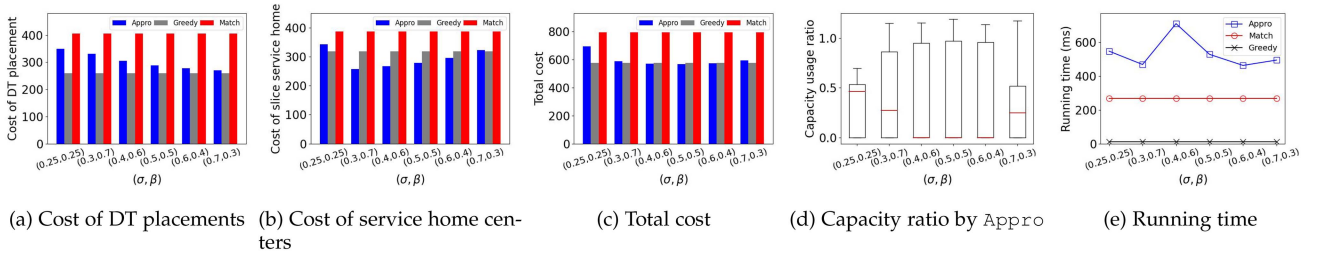
as cloudlets with less service home costs will have no sufficient capacity. Furthermore, Greedy considers DTN slicing requests one by one, and resource allocation for their service home centers is myopic. In contrast, Appro comprehensively considers the requirements of all DTN slicing requests that leads to a lower service home cost. Match determines DT placements for a subset of objects or service home placements for a subset of DTN slicing requests within each iteration, by finding a minimum-cost maximum matching. Such a strategy easily makes objects or DTN slicing requests miss choosing a cloudlet with the smallest cost, as each cloudlet is exactly matched with one object or DTN slicing request in an iteration of Match.

We then studied the performance of different algorithms, by varying the number of objects $|V|$ while fixing the number of DTN slicing requests at 100. It can be seen that Appro achieves the lowest cost among the four comparison algorithms with moderate capacity violations on some cloudlets as shown in Fig. 3(e). The costs of DT placements and service home placements by each algorithm are shown in Fig. 3(a) and (b), respectively. When the number of DTN slicing requests is fixed, it can be seen from Fig. 3(b) that the cost of service home placements by Greedy increases with the increase on the number $|V|$ of objects. The rationale behind is that the residual computing resource for service home placements decreases with the increase on the number of objects. Greedy only considers the requirement of one DTN slicing request at each time while Appro examines the requirements of all DTN slicing requests simultaneously when performing resource allocations. With less residual computing resource for service home placements with the growth on the number $|V|$ of objects, the disadvantage of Greedy becomes obvious, resulting in the increase on the service home cost. It is noticed that the solution delivered by LFD has the largest total cost among the comparison algorithms, since it does not make use of the resource consumption cost to guide service deployments.

C. Impact of Parameters on the Performance of Different Algorithms

We evaluated the impact of important parameters on the performance of the proposed algorithms. The numbers of objects and DTN slicing requests are fixed at 250 and 100, respectively.

Fig. 4 plots the performance curves of algorithms Appro, Match, Greedy, and LFD, by varying the network size $|N|$ from 20 to 80. With the growth on network size, it can be seen


 Fig. 3. Performance of different algorithms for the expected cost minimization problem, by varying the number $|V|$ of objects.

 Fig. 4. Performance of different algorithms for the cost minimization problem, by varying network size $|N|$.

 Fig. 5. Performance of Appro by varying capacity reservation ratios (σ, β) for the sub-problems.

from Fig. 4(c) and (d) that the running time of each of the algorithms increases while the value of the solution delivered by the algorithm decreases. This is due to the fact that the larger the network size, the more cloudlets can be used for DTs and home service center placements, and each algorithm can choose cloudlets with less cost from a larger number of cloudlets. It can also be observed that the costs of the solutions delivered by Appro and Greedy decrease much faster than that of Match with the growth of the network size. Fig. 4(e) shows the capacity usage ratio by Appro, from which it can be seen that the capacity violation percentage of cloudlets decreases, since more computing resource in a large network can be used for DT and home service center placements.

Fig. 5 evaluated the performance of Appro against Greedy and Match, by varying capacity reservation ratios (σ, β) . It can be seen from Fig. 5(c) that the cost by Appro is largest when α and β are set to 0.25, and there is no capacity violation by Appro under this setting, as shown in Fig. 5(d). The cost of Match is always higher than that of Appro under any capacity reservation ratios. When the capacity reservation ratio is set to (0.4, 0.6), (0.5, 0.5) or (0.6, 0.4), the cost by Appro is lower

than that by Greedy under the three different settings of α and β . It can be seen from Fig. 5(d) that the median capacity usage ratio by Appro is lower than those under the other settings, and the 75th percentile of higher than those under the other settings. This implies that DTs and service home centers that are placed in a small number of cloudlets will result in a less cost. Fig. 5(a) and (b) showed that the larger the ratio of reserved capacity, the lower the cost of the solution by Appro to the sub-problems.

D. Performance of Different Algorithms for the Dynamic DTN Slicing Request Problem

We evaluated the performance of Algorithm 4 against two benchmarks No_Control and LP2, by varying the number of DTN slicing requests from 100 to 500, assuming that there are 100 objects and their DTs have been placed in cloudlets already. Fig. 6 illustrates the performance of the three comparison algorithms. It can be seen from Fig. 6(a) that with the increase on the number $|S|$ of DTN slicing requests, the number of requests admitted by each algorithm increases first and then gradually flattens. If the number of requests is small, all of them can be

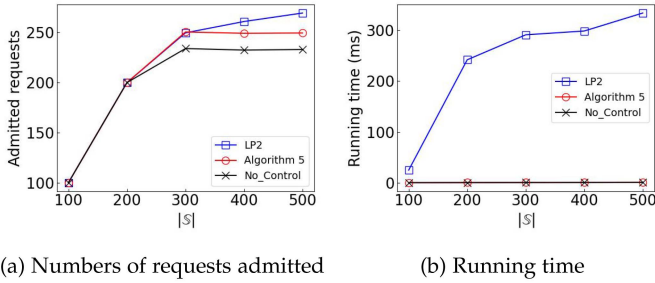


Fig. 6. Performance of different algorithms for the dynamic DTN slicing request problem, by varying the number of DTN slicing requests.

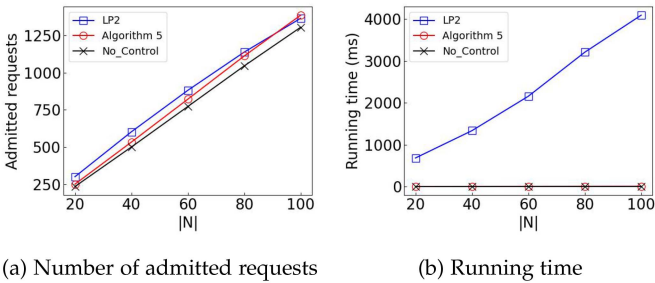


Fig. 7. Performance of different algorithms for the dynamic DTN slicing request problem, by varying the network size.

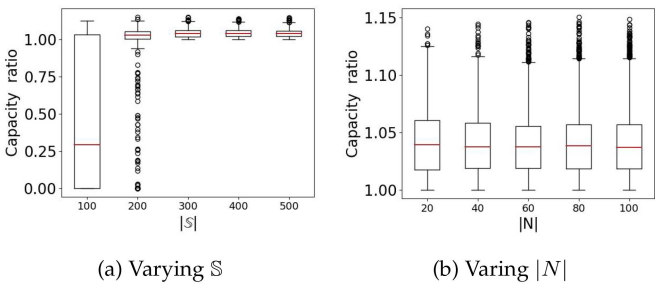


Fig. 8. Capacity usage ratio of Algorithm 4 for the dynamic DTN slicing request problem.

admitted; otherwise, only some of them can be admitted, due to limited computing resource in the network. It can also be seen from Fig. 6(a) that Algorithm 4 outperforms No_Control when the number $|S|$ of DTN slicing requests is greater than 300. It is noticed that the number of requests admitted by LP2 is the maximum one among the comparison algorithms, the solution however is an upper bound on the optimal solution of the problem, which takes the longest running time as shown in Fig. 6(b). Fig. 8(a) plots the capacity utilization ratios of cloudlets by Algorithm 4. Recall that Lemma 6 provided a conservative upper bound on the computing capacity violation on any cloudlet. The empirical results in Fig. 8(a) indicates that the resource violation on each cloudlet is moderate.

We finally investigated the impact of network size $|N|$ on the performance of algorithms while fixing the number of objects at 100 and the number of DTN slicing requests at 1,500, respectively. Fig. 7 plots the performance results of LP2, Algorithm 4, and No_Control. Fig. 7(a) shows that with the increase on the

network size, the number of requests admitted by each algorithm increases too, and Algorithm 4 improves the number of admitted requests by more than 6% than No_Control. This is because a large network has more computing resource to accommodate the service home centers of DTN slicing requests. Fig. 7(b) indicates that LP2 takes a much longer running time than that of either Algorithm 4 or No_Control for any network size. Fig. 8(b) demonstrates that the computing resource violation ratios of cloudlets do not change rapidly with the growth of network size. It can be seen that from Figs. 2 to 5, and Fig. 8 that the solutions delivered by Appro and Algorithm 4 have resource violation. In reality, such resource violations can be mitigated through scaling the resource capacity down with the given rate, e.g., if one resource capacity is violated by 10%, then the capacity used for algorithm is scaling down the original one with 90% to meet the resource constraint. The update frequency of each object also impacts the AoI of DT data, thereby affecting the performance of the algorithm. With the growth in the update frequency of each object, more resource will be consumed for updating the DTs of the objects and model retraining that the updating DTs are the source DTs of the models. On the other hand, the model retraining will improve their service accuracy, thereby admitting more requests by meeting their QoS requirements.

VIII. CONCLUSION

In this paper, we studied AoI-aware inference service provisioning in an MEC network through DTN slicing requests. We first introduced a novel AoI-aware inference service framework. We then formulated the expected cost minimization problem of AoI-aware service provisioning through placing DTs and inference service model instances, and developed efficient algorithms for the problem. Meanwhile, we also considered dynamic DTN slicing request admissions by devising an online algorithm with provable competitive ratio, provided that the DTs of all objects have already been placed. We finally validated the proposed framework and evaluated the performance of the proposed algorithms via simulations. Simulation results demonstrated that the proposed algorithms based on the framework outperform their comparison counterparts, and the proposed online algorithm improves the number of admitted requests by more than 6% than its counterpart.

ACKNOWLEDGMENT

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

REFERENCES

- [1] P. Bellavista, C. Giannelli, M. Mamei, M. Mendula, and M. Picone, "Application-driven network-aware digital twin management in industrial edge environments," *IEEE Trans. Ind. Informat.*, vol. 17, no. 11, pp. 7791–7801, Nov. 2021.
- [2] 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.

- [3] H. H. Esmat, B. Lorenzo, and W. Shi, "Toward resilient network slicing for satellite-terrestrial edge computing IoT," *IEEE Internet Things J.*, vol. 10, no. 16, pp. 14621–14645, Aug. 2023.
- [4] M. X. Goemans and D. P. Williamson, "The primal-dual method for approximation algorithms and its application to network design problems," in *Book Chapter of Approximation Algorithms for NP-Hard Problems*, Boston, MA, USA: PWS, 1997, pp. 144–191.
- [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, Art. no. 7.
- [6] A. Hossain and N. Ansari, "5G multi-band numerology-based TDD RAN slicing for throughput and latency sensitive services," *IEEE Trans. Mobile Comput.*, vol. 22, no. 3, pp. 1263–1274, Mar. 2023.
- [7] 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.
- [8] S. Jošilo and G. Dán, "Joint wireless and edge computing resource management with dynamic network slice selection," *IEEE/ACM Trans. Netw.*, vol. 30, no. 4, pp. 1865–1878, Aug. 2022.
- [9] S. Karunaratna, S. Wijethilaka, P. Ranaweera, K. T. Hemachandra, T. Samarasinghe, and M. Liyanage, "The role of network slicing and edge computing in the metaverse realization," *IEEE Access*, vol. 11, no. 4, pp. 25502–25530, 2023.
- [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., "AoI-aware, digital twin-empowered IoT query services in mobile edge computing," *IEEE/ACM Trans. Netw.*, to be published, 2024, doi: [10.1109/TNET.2024.3395709](https://doi.org/10.1109/TNET.2024.3395709).
- [12] 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.
- [13] 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.
- [14] J. Li, W. Liang, Z. Xu, X. Jia, and W. Zhou, "Service provisioning for multi-source IoT applications in mobile edge computing," *ACM Trans. Sensor Netw.*, vol. 18, no. 2, 2022, Art. no. 17.
- [15] J. Li, W. Liang, W. Xu, Z. Xu, Y. Li, and X. Jia, "Service home identification of multiple-source IoT applications in edge computing," *IEEE Trans. Mobile Comput.*, vol. 16, no. 2, pp. 1417–1430, Mar./Apr. 2023.
- [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] L. Li, L. Tang, Q. Liu, Y. Wang, X. He, and Q. Chen, "DTN assisted dynamic cooperative slicing for delay-sensitive service in MEC-enabled IoT via deep deterministic policy gradient with variable action," *IEEE Internet Things J.*, vol. 10, no. 12, pp. 10708–10724, Jun. 2023.
- [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.*, to be published, 2023, doi: [10.1109/TSC.2023.3341988](https://doi.org/10.1109/TSC.2023.3341988).
- [19] 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.
- [20] 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.
- [21] Y. Mao, X. Shang, and Y. Yang, "Near-optimal resource allocation and virtual network function placement at network edges," in *Proc. IEEE 27th Int. Conf. Parallel Distrib. Syst.*, 2021, pp. 18–25.
- [22] Y. Mao, X. Shang, Y. Liu, and Y. Yang, "Joint virtual network function placement and flow routing in edge-cloud continuum," *IEEE Trans. Comput.*, vol. 73, no. 3, pp. 872–886, Mar. 2024.
- [23] R. Minerva, G. M. Lee, and N. Crespi, "Digital twin in the IoT context: A survey on technical features, scenarios, and architectural models," *Proc. IEEE*, vol. 108, no. 10, pp. 1785–1824, Oct. 2020.
- [24] A. Nadembega, T. Taleb, and A. Hafid, "A destination prediction model based on historical data, contextual knowledge and spatial conceptual maps," in *Proc. IEEE Int. Conf. Commun.*, 2012, pp. 1416–1420.
- [25] F. Naeem, G. Kaddom, and M. Tariq, "Digital twin-empowered network slicing in B5G networks: Experience-driven approach," in *Proc. IEEE Globecom Workshops*, 2021, pp. 1–5.
- [26] NetworkX, 2020. Accessed: Jul. 2022. [Online]. Available: <https://networkx.org/>
- [27] R. Prasad and A. Sunny, "Scheduling slice requests in 5G networks," *IEEE/ACM Trans. Netw.*, vol. 31, no. 6, pp. 3025–3036, Dec. 2023.
- [28] 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.
- [29] H. Shen, Y. Tian, T. Wang, and G. Bai, "Slicing-based task offloading in space-air-ground integrated vehicular networks," *IEEE Trans. Mobile Comput.*, vol. 23, no. 5, pp. 4009–4024, May 2024.
- [30] D. Shomys and E. Tardos, "An approximation algorithm for the generalized assignment problem," *Math. Program.*, vol. 62, pp. 461–474, 1993.
- [31] 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.
- [32] L. Tang, Y. Du, Q. Liu, J. Li, S. Li, and Q. Chen, "Digital twin assisted resource allocation for network slicing in industry 4.0 and beyond using distributed deep reinforcement learning," *IEEE Internet Things J.*, vol. 10, no. 19, pp. 16989–17006, Oct. 2023.
- [33] H. Tong, S. Wang, Z. Yang, J. Zhao, M. Bennis, and C. Yin, "Semantic-aware remote state estimation in digital twin with minimizing age of incorrect information," in *Proc. IEEE Glob. Commun. Conf.*, 2023, pp. 4534–4539.
- [34] 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.
- [35] H. Wang, Y. Wu, G. Min, and W. Miao, "A graph neural network-based digital twin for network slicing management," *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1367–1376, Feb. 2022.
- [36] Y. Wu, K. Zhang, and Y. Zhang, "Digital twin networks: A survey," *IEEE Internet Things J.*, vol. 8, no. 18, pp. 13789–13804, Sep. 2021.
- [37] G. Xu, S. Gao, M. Daneshmand, C. Wang, and Y. Liu, "A survey for mobility Big Data analytics for geolocation prediction," *IEEE Wireless Commun.*, vol. 24, no. 1, pp. 111–119, Feb. 2017.
- [38] Z. Xu et al., "Stateful serverless application placement in MEC with function and state dependencies," *IEEE Trans. Comput.*, vol. 72, no. 9, pp. 2701–2716, Sep. 2023.
- [39] R. D. Yates, Y. Sun, D. R. Brown, S. K. Kaul, E. Modiano, and S. Ulukus, "Age of information: An introduction and surveys," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 5, pp. 1183–1210, May 2021.
- [40] Y. Yi, G. Zhang, and H. Jiang, "Online digital twin-empowered content resale mechanism in age of information-aware edge caching networks," 2024, *arXiv:2403.07868*, doi: [10.48550/arXiv.2403.07868](https://doi.org/10.48550/arXiv.2403.07868).
- [41] 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.
- [42] 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.
- [43] 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.*, to be published, 2024, doi: [10.1109/TMC.2024.3394839](https://doi.org/10.1109/TMC.2024.3394839).
- [44] L. Zhao, Z. Bi, A. Hawbani, K. Yu, Y. Zhang, and M. Guizani, "ELITE: An intelligent digital twin-based hierarchical routing scheme for software-defined vehicular networks," *IEEE Trans. Mobile Comput.*, vol. 22, no. 9, pp. 5231–5247, Sep. 2023.
- [45] J. Zheng, A. Banchs, and G. de Veciana, "Constrained network slicing games: Achieving service guarantees and network efficiency," *IEEE/ACM Trans. Netw.*, vol. 31, no. 6, pp. 2698–2713, Dec. 2023.



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 is a post-doc with the Department of Computer Science, City University of Hong Kong. Her research interests include edge computing, digital twin, network function virtualization, and optimization problems.



Weifa Liang (Senior Member, IEEE) received the PhD degree in computer science from Australian National University, in 1998. He is a full professor with the Department of Computer Science, City University of Hong Kong. Prior to that, he was a full professor with 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*.



Wenzheng Xu (Member, IEEE) received the BSc, ME, and PhD degrees in computer science from Sun Yat-Sen University, Guangzhou, China, in 2008, 2010, and 2015, respectively. He currently is an associate professor with Sichuan University, China. He was a visitor with both the Australian National University, Australia and the Chinese University of Hong Kong, Hong Kong. His research interests include wireless ad hoc and sensor networks, mobile computing, approximation algorithms, combinatorial optimization, online social networks, and graph theory.



Zichuan Xu (Member, IEEE) received the PhD degree in computer science from 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 and serverless computing, network function virtualization, algorithmic game theory, and optimization problems.



Min Chen (Fellow, IEEE) has been a full professor with the School of Computer Science and Engineering, South China University of Technology. He is also the director of Embedded and Pervasive Computing (EPIC) Lab, Huazhong University of Science and Technology, China. He is the founding chair of IEEE Computer Society Special Technical Communities on Big Data. He was an assistant professor with the School of Computer Science and Engineering, Seoul National University before he joined HUST. He is the chair of IEEE Globecom 2022 eHealth Symposium. His Google Scholar Citations reached more than 40,000 with an h-index of 96. His top paper was cited more than 4,304 times. He was selected as Highly Cited Researcher from 2018 to 2022. He got IEEE Communications Society Fred W. Ellersick Prize, in 2017, the IEEE Jack Neubauer Memorial Award, in 2019, and IEEE ComSoc APB Outstanding Paper Award, in 2022. He is a fellow of IET.