Mobility-Aware Service Provisioning in Edge Computing via Digital Twin Replica Placements

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

***Abstract*—Digital twin (DT) has been emerging as an enabling technology to provide seamless interactions between the virtual cyber world and the real world. The explosion of IoT devices (objects) further fuels the development of the DT technology, and paves the way to real-time monitoring, behavior simulations and decisive predictions on objects through their digital counterparts. Meanwhile, Mobile Edge Computing (MEC) has been envisioned as a promising computing paradigm for various IoT applications with stringent delay requirements. In this paper, we study mobility-aware, delay-sensitive service provisioning in a DT-empowered MEC network with the mobility of both users and objects through DT replica placements of mobile objects. To this end, we first formulate two novel optimization problems: the DT replica placement problem and the dynamic DT replica placement problem, respectively, and show NP-hardness of the two problems. We then formulate an Integer Linear Programming (ILP) solution to the DT replica placement problem when the problem size is small or medium; otherwise we devise a randomized algorithm with high probability, provided that the mobility profiles of each object and each user are given. Meanwhile, we also develop an online algorithm for the dynamic DT replica placement problem, where for a given time horizon, service requests arrive one by one without the knowledge of future arrivals, each arrived request must be responded immediately by accepting or rejecting it. However, the heterogeneity and dynamics of user requests on resource demands may lead to the removals and re-instantiations of DT instances frequently. To mitigate this, we propose an efficient prediction mechanism to reserve a certain number of DTs for future by introducing the timestamp concept. We finally evaluate the performance of the proposed algorithms by simulations. Simulation results show that the proposed algorithms are promising, and outperform the performance of other comparison counterparts.**

***Index Terms*—Digital twin-empowered edge computing, mobility of users and objects, multiple digital twin replica placements, randomized algorithm and online algorithm, seamless service provisioning.**

Manuscript received 3 July 2023; revised 6 April 2024; accepted 23 April 2024. Date of publication 30 April 2024; date of current version 5 November 2024. The work of Yuncan Zhang and Weifa Liang was supported by Hong Kong Research Grants Committee (HKRGC) CityU under Grant 9380137, Grant 7005845, Grant 9043510, and CRF Grant 8730103, respectively. The work of Zichuan Xu was supported in part by the National Natural Science Foundation of China under Grant 61802048 and in part by the "Xinghai Scholar Program" in Dalian University of Technology, China. Recommended for acceptance by W. Gao. *(Corresponding author: Weifa Liang.)*

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

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

Digital Object Identifier 10.1109/TMC.2024.3394839

I. Introduction

Driven by the rapid development of Internet of Things (IoT) in a variety of application scenarios, a vast amount of data generated by IoT devices creates the requirement of adequate organization and management of the information [1]. The Digital Twin (DT) technology has gained increasing attentions for its ability to digitize the physical world [32]. Through DT modeling, the virtual representation of an object can record its historical data traces, emulate its behaviors in future, and provide DT services on objects through data analytics, artificial intelligence and machine learning. According to a market research report, the DT technology has been widely adopted across industries, such as manufacturing, healthcare, autonomous driving, and aerospace. The global DT market size is projected to reach USD 155.83 billion by 2030 [21].

In recent years, the Mobile Edge Computing (MEC) paradigm was conceived in a bid to fill the gap between centralized clouds and mobile devices by deploying more advanced storage and processing functionalities at the network edge, which curtails the service delay and alleviates the burden of the core network. Along with the development of DT, there is growing interest in DT-assisted service provisioning in MEC [7], [29], [32], and the DT placement becomes a fundamental problem, considering limited computing capacity on edge servers, the mobility of both users and objects, and heterogeneous resource demands and stringent delay requirements of user requests. DT migration is a viable solution to cope with service delays of some delay-sensitive IoT applications due to the mobility of users and objects [2], [20]. However, the migration delay cannot be ignored as it may lead to critical service disruptions.

To provide seamless, continuously uninterrupted, delay-sensitive services in an MEC network for mobile users and objects, one efficient method is to adopt redundancy. That is, the DT of each object is replicated into multiple replicas, and the DT replicas are deployed at the nearby locations where the object and its users often visit. Note that an object can upload its updating data to its DT from its current location, while a user can issue its service request from its current location too. Thus, both the DT update data and users requesting DT services at anytime with any location can be achieved while a much less cost incurs. In comparison with delays experienced on DT migrations, the DT replication strategy is desirable for delay-sensitive services, as DT replicas can address the non-trivial relationship between the mobility of users and objects and service delays of users, at expense of a moderate amount of computing resource consumed on synchronizations of the DT replicas. Whenever there is a data update of an object, the object needs to synchronize the data update with all its DT replicas, by uploading the update data to these DT replicas. This results in both computing and communication resource consumptions of the network. Thus, there is a dilemma on the DT replica placement. On one hand, placing more DT replicas for each object implies more synchronization costs incurred on DT replicas. On the other hand, multiple DT replica placements not only serve more mobile users by meeting their delay requirements but also reduce their service costs.

In this paper, we consider inference service provisioning in a DT-assisted MEC, which is essentially different from service provisioning in conventional MEC. That is, DT-assisted service models need to be retrained quite often, by the update data from their source - the DTs of objects, in order to keep service accuracy. The service results in traditional MEC will be identical when there are two requests for the same service with identical inputs. In contrast, the service results by two service requests with identical inputs for a service model in DT-assisted MEC are very likely to be different at different time points due to the service model updating during that period.

The novelty of this paper lies in utilizing DT replicas to provide continuously uninterrupted DT services in an MEC network with mobile users and objects. Most existing work assumed that there is only one DT for each object placed in MEC, we are the first to introduce multiple DT replicas to cope with the mobility of users and objects by formulating two novel DT replica placement problems. It poses great challenges. For each given object, how many DT replicas are needed to cope with its mobility and its users? and where are the DT replicas placed while meeting its user delay requirements? In this paper, we will address the mentioned challenges.

The main contributions of this paper are presented as follows.

- We consider mobility-aware continuous service provisioning in a DT-assisted MEC network with mobile objects and users through DT replica placements. We formulate two novel DT replica placement problems and show that both defined problems are NP-hard.
- We provide an Integer Linear Programming (ILP) solution to the DT replica placement problem when the problem size is small; otherwise, we devise a randomized algorithm with high probability for the problem.
- We devise an online algorithm for the dynamic DT replica placement problem, in which an efficient prediction method was developed to cope with frequent removals and re-instantiations of DT replicas, by assigning a timestamp to each DT replica in the system.
- We evaluate the performance of the proposed algorithms through simulations. Simulation results show that the proposed algorithms are promising, and outperform their comparison baseline algorithms.

The remainder of this paper is organized as follows. Section II reviews the state-of-the-art works in mobility-aware DT-empowered service provisioning in MEC networks. Section III introduces the system model, notions and notations. It also introduces problem definitions and shows NP-hardness of the defined problems. Section IV formulates an ILP solution to the DT replica placement problem if the problem size is small or medium; otherwise, a novel randomized algorithm with high probability is devised. Section V develops an online algorithm for the dynamic DT replica placement problem, and also an efficient prediction method based on the timestamp concept for DT replica reservation is proposed too. Section VI conducts simulations to evaluate the performance of the proposed algorithms, and Section VII concludes the paper.

II. Related Work

Service provisioning in an MEC network has been extensively studied in the past several years [22], [23], [27], [31], [34]. Particularly, Ma et al. [22] investigated Virtual Network Function (VNF) service provisioning in MEC by considering user mobility and service delay requirements with the aim to maximize the accumulative network utility. Notice that the VNF replicas that are used to process VNF service requests do not change once deployed [22], and none of these studies considered DT-assisted service provisioning in MEC that utilizes DTs for behaviour predictions and emulations of objects. In contrast, DT replicas need to continuously synchronize the updated data with their objects, and the mobility of both objects and users should be taken into account for DT replica placement. Existing mobility-aware service provisioning methods did not handle continuously monitoring the state information of objects and thus cannot be directly applied to delay-aware service provisioning in an MEC by DT replicas.

Several recent studies investigated the mobility-aware service provisioning in DT-assisted MEC [7], [9], [10], [12], [19], [32]. For instance, Gong et al. [7] considered the mobility of vehicles in a vehicular edge network, and proposed a holistic network virtualization mechanism for service-centric and user-centric service provisioning, through utilizing DT-oriented network slicing. Lei et al. [9] leveraged DTs of Unmanned Aerial Vehicles (UAVs) to monitor the life cycle of a UAV swarm with high mobility, and proposed a machine learning algorithm to optimize the behaviours of UAVs. Zhang et al. [35] studied multiple Federated Learning (FL) in DT-assisted edge computing. They developed efficient approaches for offline multi-FL services and a Deep Reinforcement Learning (DRL) algorithm for online multi-FL services, respectively. Li et al. [10] mapped each base station in an MEC network to a DT to estimate the channel and workload state of the base station under user mobility assumptions. They developed a DRL algorithm to minimize the total energy consumption of UAVs. Lin et al. [19] investigated dynamic and stochastic DT service demands of mobile users, by devising an incentive-based congestion control scheme to maximize the long-term profit of the network service provider. Liang et al. [18] investigated the non-trivial relationship between the freshness of a service model and Age of Informations (AoIs) of its source DT data for model training. They devised an efficient algorithm for maximizing the accumulative freshness of all models while minimizing the total cost of various resources consumed for DT updating, model training, and users requesting for accurate model services. Vaezi et al. [30] considered the problem of DT placements such that the maximum data request-response delay experienced by an application over all objects is minimized, subject to the maximum data age target constraints on DTs. Li et al. [12] considered utilization maximization under user mobility assumption, they formulated efficient performance-guaranteed solutions to the problem. Wang et al. [32] proposed a Mobility Digital Twin (MDT) framework for transportation in smart cities, i.e., creating digital twins for human, vehicles and traffic infrastructures to implement functionalities such as modelling, learning, and prediction, thereby providing efficient mobility services.

There are several efforts that mitigate the impact of user mobility on the service quality in DT-assisted MEC [11], [15], [20], [29]. For example, Lu et al. [20] developed a DRL-based algorithm to place and migrate DTs of mobile devices in MEC to reduce the system latency. Sun et al. [29] made use of DTs of edge servers to predict their states under uncertain user mobility, and developed a DRL algorithm to minimize the offloading delay of mobile users. Li et al. [11], [15] investigated the dynamic DT placement for improving user service satisfaction with an assumption of two DT replicas for each object: a primary DT located in the remote cloud and a local cached DT. With the aim of maximizing the accumulative user service satisfaction measured by the AoI of DT data, they formulated two utility maximization problems, and proposed a constant approximation algorithm and a performance-guaranteed online algorithm for the two problems. Li et al. [13] dealt with DT synchronizations via continual learning in an MEC environment. They focused on the static DT synchronization problem per time slot and the dynamic DT synchronization problem for a finite time horizon, by developing efficient solutions to the problems. Zhang et al. [36] studied DT placement and migration problems in an MEC network with the mobility assumption of objects and users by jointly considering the freshness of DT data and the service cost of users requesting for DT data.

Unlike the aforementioned studies, there is only one DT for each object in an edge server, in this paper we investigate mobility-aware, continuous service provisioning in a DT-assisted MEC network through multiple DT replica placements. We are the very first to introduce multiple DT replicas for each object to cope with the mobility of both objects and users, and place the DT replicas to nearby locations of objects and users in order to reduce the total service cost of users and the updating cost of DTs.

III. Preliminaries

In this section, we introduce the system model, notions and notations, and define the problems precisely. We also show NP-hardness of the defined problems.

A. System Model

We consider an MEC network $G = (N, E)$, where N is a set of Access Point (AP) nodes, and E is the set of links between cloudlets. Each AP is co-located with a cloudlet, and the communication delay between the AP and its co-located cloudlet is negligible. Without loss of generality, in the rest of the paper, an AP and its co-located cloudlet will be used interchangeably. Each cloudlet $c_l \in N$ has limited computing capacity C_l for hosting DTs of objects. Each link $e \in E$ is a high-speed optical link connecting two APs, and there are two metrics $\xi(e)$ and $d(e)$ representing the transmission cost and transmission delay of routing a unit of data along link e. Each AP covers a certain area and mobile users can access the network through the AP if the users are within the coverage of the AP. Let $Path_{l,l'}$ be the shortest path in G between cloudlets l and l' in terms of transmission delay of a unit data along the path. We assume that the routing cost of data along a path is proportional to the delay of the path.

Given a set V of objects, and a set U of mobile users under the coverage of AP nodes in the MEC network, we deal with user service requests by requesting the DT data of objects. For the sake of convenience, in the rest of our discussions, we use index i, j and l to represent the indices of objects, users, and cloudlets, respectively.

For each object $v_i \in V$, let $U(i) \subset U$ denote the set of users requesting services from the DT of object v_i, and $U(i) \cap U(j) = \emptyset$ if $i \neq j$. The request of each user u_j is represented by a tuple $(b(j), vol(u_j), D_j)$, where $b(j)$ is the index of requested object of the user, $vol(u_j)$ is the amount of request data generated by u_j to be processed, and D_j is the delay requirement. We assume that both objects and users are movable in the network. For each mobile object $v_i \in V$ and each mobile user $u_j \in U$, let $p_{i,l}$ and $q_{j,l}$ denote the probabilities of object v_i and user u_j at location AP_l, respectively, then $\sum_{l=1}^{|N|} p_{i,l} = 1, \forall v_i \in V$, and $\sum_{l=1}^{|N|} q_{j,l} = 1, \forall u_j \in U$. In principle, both objects and users can move to any location in the MEC network.

To measure the quality of service provisioning, each user $u_j \in U$ has an end-to-end delay requirement D_j that indicates the maximum response time of querying a DT replica of its requested object. To provide continuously non-interrupted services to mobile users, it is not uncommon to deploy multiple DT replicas of an object to different locations at which its users often visit. To maintain all DT replica statuses of each object strongly consistent and keep the DT states as fresh as possible, it requires that all of the DT replicas synchronize with the object often whenever there is any update on the object. We here adopt a DT replica strategy to proactively deploy a set of DT replicas for each object in response to its user service requests, under the movement of both objects and users. Furthermore, the data updating frequency of an object usually is determined by concrete applications. For example, if a mobile device is powered by limited battery, it cannot upload its updating data frequently for a long monitoring period; otherwise, its energy will run out of the energy quickly [16].

Generally, an object is either a physical object, e.g., a sensor or a camera, or a piece of software, and an object synchronizes its status with its DT replicas. Users request DT-assisted services where the service models are continuously retrained by the update data of DTs. Fig. 1 is an illustrative example of service provisioning in a DT-assisted MEC network via DT replica placements, where object v_1 is an autonomous vehicle in Vehicle

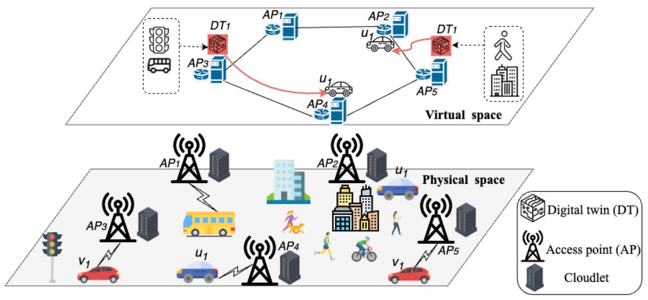


Fig. 1. An illustrative example of an MEC network with DT replica placements.

Internet of Things (V-IoT) that can communicate with various sensors and devices [6]. In this application scenario, v_1 moves from the coverage area of AP_5 to the coverage area of AP_3, and its DT DT_1 has two replicas that are placed at cloudlet c_3 and cloudlet c_5, respectively. User u_1 requests data from DT_1 of vehicle v_1 to detect anomaly for the V-IoT. On one hand, the smart vehicle v_1 gathers comprehensive information about its own performance, its surrounding environment, and the behaviour of its driver, and it then synchronizes its updating data with its DT DT_1. Thus, DT_1 enables to simulate and analyze some crucial factors such as traffic trajectories, mechanical failure, and pedestrian density to avoid fatal accidents. By leveraging various analytical techniques, DT_1 identifies anomalies in the V-IoT system including unusual behaviors from normal patterns or any abnormal deviations, and then notifies user u_1 about possible issues or emergencies. On the other hand, when user u_1 is located under the coverage of AP_4, u_1 requests data from DT_1 at cloudlet c_3; when u_1 moves to the coverage area of AP_2, DT_1 at cloudlet c_5 is used to serve user u_1. Note that a DT replica not only receives data from its physical vehicle, but also collect data from a wide range of data sources within the region like city regulations, weather forecast, traffic signals, and so on. The replicas of DT_1 hosted by different cloudlets provide services with regional customization for user u_1 with the reduced latency.

B. Cost Modeling of DT Replicas

We deal with user service request for a DT replica of an object, where each object has multiple DT replicas deployed in the MEC network. Let $x_{i,l}$ be a binary variable that indicates whether a DT replica of object v_i is placed in cloudlet AP l ($x_{i,l} = 1$) or not ($x_{i,l} = 0$). Since users are mobile in the network, a user can choose one of the DT replicas of its requested object for DT service provisioning. To meet its delay requirement, the user usually chooses the nearest DT replica for service.

DT replica deployment cost: The deployment of DT replicas consumes computing resource for data processing in cloudlets. The total cost of computing resource consumptions by deploying DT replicas for all objects is

$$cost_{DT} = \sum_{v_i \in V} \sum_{l=1}^{|N|} (\mu \cdot comp(DT_i) + Init_i) x_{i,l}, \quad (1)$$

where $comp(DT_i)$ is the amount of computing resource of deploying a DT replica of object v_i, μ is the cost of unit computing resource, and $Init_i$ is the instantiation cost of a DT replica of v_i.

DT replica synchronization cost of each object: The update data generated by object v_i will be uploaded through its gateway (the AP of its current location) first, and then is synchronized with all its DT replicas deployed in the MEC network. Let l' be the current location of object v_i. The uploading cost of the update data of object v_i at AP l' is expressed by

$$upload\_cost\,(v_i, l') = \zeta \cdot P(v_i) \cdot \frac{vol(v_i)}{B\,(v_i, AP_{l'})}, \quad (2)$$

where ζ is the unit cost of transmission power consumption per unit time, $P(v_i)$ is the transmission power of v_i, $vol(v_i)$ is the volume of the update data of v_i, and $B(v_i, AP_{l'})$ is the uploading rate of v_i to its connected $AP_{l'}$.

The update data of object v_i will be sent to each of its DT replicas in order to ensure that these DT replica statuses be strongly consistent, i.e., users who request the DT service of object v_i from any of its DT replicas will receive the same result.

The DT replica synchronization cost $cost_{obj}(v_i)$ of object v_i when uploading its update data $vol(v_i)$ is calculated as follows. When object v_i is located at AP l', its synchronization cost $cost_{obj}(i, l')$ is

$$\begin{aligned} cost_{obj}\,(i, l') = p_{i,l'} \Bigg(& upload\_cost\,(v_i, l') \\ & + \sum_{l=1}^{|N|} \left(vol(v_i) cost\,(Path_{l',l}) + \frac{\kappa_i \cdot vol(v_i)}{f(DT_i)} \right) x_{i,l} \Bigg), \end{aligned} \quad (3)$$

where $p_{i,l'}$ is the probability of v_i at AP l', $cost(Path_{l',l}) = \sum_{e \in Path_{l',l}} \xi(e)$ is the transmission cost of a unit data along the shortest path $Path_{l',l}$ in G between cloudlets l' and l in terms of transmission delays on its edges, κ_i is the cost of data processing per unit time of DT_i, and the data processing rate $f(DT_i)$ of DT_i depends on the amount $comp(DT_i)$ of computing resource allocated to it.

The synchronization cost of object v_i at all potential locations in N thus is

$$cost_{obj}(v_i) = \sum_{l'=1}^{|N|} cost_{obj}(i, l'). \quad (4)$$

Service cost of each user request: For each user $u_j \in U$ with a service request represented by a tuple $(b(j), vol(u_j), D_j)$, it needs to choose one DT replica of its requested object for service provisioning at each sojourn location. Let $y_{j,l',l}$ be a binary variable to indicate whether the DT replica located at c_l will be chosen by user u_j located at $c_{l'}$, i.e., $y_{j,l',l} = 1$ if u_j chooses the DT replica placed in cloudlet c_l when user u_j is located at cloudlet $c_{l'}$, and $y_{j,l',l} = 0$, otherwise. We admit a user request only when there is a DT replica of its request object to serve the request while meeting the delay requirement of the request.

In the following we first detail the calculation of the end-to-end delay of each user u_j, and then consider its service cost.

The end-to-end delay of each user consists of (i) the data uploading delay of the request between the user and its connected AP, (ii) the transmission delay between the AP and the cloudlet that hosts the DT replica for processing the request, and (iii) the request processing delay at the cloudlet hosting the DT replica. The end-to-end delay of u_j at location AP l' thus is

$$d_{e2e}(u_j, AP_{l'}) = \frac{vol(u_j)}{B(u_j, AP_{l'})} + (vol(u_j)d(Path_{l',l}) + \frac{vol(u_j)}{f(DT_i)})y_{j,l',l}, \tag{5}$$

where $vol(u_j)$ is the amount of request data generated by u_j, $B(u_j, AP_{l'})$ is the uploading rate of u_j to $AP_{l'}$, $d(Path_{l',l}) = \sum_{e \in Path_{l',l}} d(e)$ is the transmission delay of a unit data along the shortest path $Path_{l',l}$ between cloudlets $c_{l'}$ and c_l, and $f(DT_i)$ is the data processing rate at the DT replica DT_i of object v_i.

The service cost of user u_j at location $AP_{l'}$ consists of (i) the uploading cost to its connected $AP_{l'}$, (ii) the update data transmission cost from $AP_{l'}$ to cloudlet c_l at which its chosen DT replica is located, and (iii) the processing cost of the DT replica data at cloudlet c_l. The calculation of the uploading cost is similar to that of an object, i.e., $upload\_cost(v_i, l') = \zeta \cdot P(u_j) \cdot \frac{vol(u_j)}{B(u_j, AP_{l'})}$, where $P(u_j)$ is the transmission power of u_j and $B(u_j, AP_{l'})$ is the uploading rate of u_j to its connected $AP_{l'}$. The service cost $cost_{user}(j, l')$ of user u_j at location $AP_{l'}$ is

$$cost_{user}(j, l') = q_{j,l'}(upload\_cost\,(u_j, l') + \sum_{l=1}^{|N|} \left(vol(u_j)cost\,(Path_{l',l}) + \frac{\kappa_i \cdot vol(u_j)}{f\,(DT_i)} \right) y_{j,l',l} \quad \text{if } \mathrm{d_{e2e}\ (u_j,\ AP_{l'}) \le D_j}, \tag{6}$$

otherwise, $cost_{user}(j, l') = 0$.

The service cost $cost_{user}(u_j)$ of user u_j for its request admissions at all locations is

$$cost_{user}(u_j) = \sum_{l'=1}^{|N|} cost_{user}(j, l'). \tag{7}$$

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

C. Problem Formulations

We consider mobility-aware, delay-sensitive DT service provisioning in an MEC network for mobile users with meeting their delay requirements. In reality, both objects and users typically move freely in the network and their movement patterns may change over time. On one hand, when an object moves to a new location, if none of its DT replicas is placed into cloudlets near to its new location, the increase on the synchronization delay may result in the violation of the delay requirements of its users. On the other hand, when a user moves to a new location that is far away from the DT replicas of its requested object, it is very likely that none of the DT replicas of the object can provide services to the user without violating its delay requirement.

To provide seamless and uninterrupted services for mobile users, a viable solution is to deploy multiple DT replicas at different locations to meet their delay requirements. We thus consider two problems: the DT replica placement problem by assuming the mobility profiles of users and objects and a set of user requests are given; and the dynamic DT replica placement problem by assuming that both users and objects and user requests dynamically change overtime. The two problems are defined as follows.

Definition 1: Given an MEC $G = (N, E)$ with a set N of cloudlets, a set V of mobile objects, and a set U of mobile users that move around in the network, each cloudlet $c_l \in N$ has computing capacity C_l. For each object $v_i \in V$, there is a set $U(i) \subset U$ of users requesting services from its DT replicas, the request of each user $u_j \in U$ is represented by a tuple $(b(j), vol(u_j), D_j)$. Assuming that the mobility profiles of both users and objects are given, *the DT replica placement problem* in G is to place a proper number of DT replicas for each object to meet the service demands of its users such that the total service cost of all user requests is minimized, subject to computing capacity on each cloudlet.

Definition 2: Given an MEC network $G = (N, E)$ with a set N of cloudlets with each cloudlet $c_l \in N$ having computing capacity C_l, a finite time horizon that is divided into T equal time slots, a set V of objects with each object $v_i \in V$ having one or multiple DT replicas placed in cloudlets, and a set $U^{(t)}$ of users (existing and new users) requesting for DT services in the beginning of time slot t with $t \in T$, *the dynamic DT replica placement problem* is to find a scheduling to admit requests through DT replica placements and removals of objects at each time slot such that the number of requests admitted for the given time horizon is maximized while the total service cost of admitted requests is minimized, subject to computing capacity on each cloudlet in G.

D. NP Hardness of Problems

Theorem 1: The DT replica placement problem in $G = (N, E)$ is NP-hard.

Proof: We prove the NP hardness of the DT replica placement problem by reducing a well-known NP-hard problem - the minimum-cost generalized assignment problem (GAP) to it.

There are n bins with each bin j with capacity C_j, and there is a set of items to be assigned to the bins. If item i is assigned to bin j, item i consumes the amount $size(i, j)$ of computing resource of bin j and incurs cost $cost(i, j)$. *The minimum-cost GAP* is to assign as many items as possible to the bins such that the total cost of assigned items is minimized, subject to the computing resource capacity on each bin [28].

We consider a special case of the DT replica placement problem where each user is stationary at a specified AP node and each object has only one user, i.e., $|U(i)| = 1, \forall i$. Under this assumption, deploying one DT replica for each object is sufficient to satisfy the delay requirement of its user as the user stays at the co-cloudlet of its AP. We assume that each cloudlet (bin) l has

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.
C_l	The computing capacity of cloudlet $c_l \in N$.
$\xi(e)$ and $d(e)$	The transmission cost and transmission delay of routing a unit of data along link $e \in E$.
$Path_{l,l'}$	The shortest path in G between cloudlets l and l' in terms of transmission delay/cost of a unit data along the path.
$cost(Path_{l',l})$, $d(Path_{l',l})$	The transmission cost and the transmission delay of a unit data along the shortest path $Path_{l',l}$ in G between cloudlets l' and l, respectively.
V and U	A set of objects and a set of mobile users.
$comp(DT_i)$	Computing resource required by DT_i.
$f(DT_i)$	The data processing rate of DT_i.
κ_i	The cost of data processing per unit time of DT_i.
$U(i)$	The set of users requesting services from the DT of object v_i, and $U(i) \cap U(j) = \emptyset$ if $i \neq j$.
$(b(j), vol(u_j), D_j)$	The request of user $u_j \in U$, where $b(j)$ is the index of requested object of the user, $vol(u_j)$ is the amount of request data generated by u_j to be processed and D_j is the delay requirement.
$p_{i,l}$ and $q_{j,l}$	The probabilities of object v_i and user u_j at location AP_l, respectively, where $\sum_{l=1}^{\lvert N\rvert} p_{i,l} = 1$, $\forall v_i \in V$, and $\sum_{l=1}^{\lvert N\rvert} q_{j,l} = 1$, $\forall u_j \in U$.
$cost_{DT}$	DT replica deployment cost.
$upload\_cost(v_i, l')$	The uploading cost of the update data of object v_i at AP l'.
ζ	The cost of transmission power consumption per time unit.
$P(u_j)$ and $P(v_i)$	The transmission power of user u_j and object v_i, respectively.
$B(v_i, AP_{l'})$	The uploading rate of v_i to its connected $AP_{l'}$.
$cost_{obj}(v_i)$	The DT replica synchronization cost of object v_i.
$cost_{obj}(v_i)$	The synchronization cost of object v_i at all possible locations in N.
$d_{e2e}(u_j, AP_{l'})$	The end-to-end delay of user u_j at location AP l'.
$cost_{user}(j, l')$	The service cost of user u_j at location $AP_{l'}$.
$cost_{user}(u_j)$	The service cost of user u_j for its request admissions at all locations.
$Cost$	The total cost of the DT placement problem.
$x_{i,l}$	Binary variable to indicate whether a DT replica of object v_i is placed in cloudlet AP l ($x_{i,l} = 1$) or not ($x_{i,l} = 0$).
$y_{j,l',l}$	Binary variable indicating whether user u_j at location l' will use the DT replica at cloudlet l for its service processing ($y_{j,l',l} = 1$) or not ($y_{j,l',l} = 0$).
T	The set of time slots within the time horizon.
$U^{(t)}$	The set of user requests arrived in the beginning of time slot t.
$U^{(t)}_{exist}$	The set of users that can make use of an existing DT replica for its service while meeting its delay requirement.
$U^{(t)}_{non}$	The set of users that cannot find any existing DT replica to satisfy its service delay requirement.
$U^{(t)}_{admit}$	The set of admitted requests at time slot $t \in T$.
$cost^{(t)}_{user}(u_j, l', l)$	The service cost of user u_j located at l' chooses the DT replica placed at cloudlet l for its service provisioning.
$N_{i,t}$	The set of cloudlets with sufficient computing resource to instantiate a DT replica of v_i in it at time slot t.
$ratio^{(t)}(i, l)$	The ratio of the number $\lvert U^{(t)}_{non}(i, l)\rvert$ of admitted users in $U^{(t)}_{non}(i)$ using the DT replica of v_i deployed in cloudlet $c_l \in N_{i,t}$ for their services to the total cost of their admissions.
$ratio^{(t)}(i, l_i)$	The maximum ratio of deploying a DT replica of object v_i among all potential cloudlets in N.
$ratio^{(t)}(i_0, l_{i_0})$	The maximum value of $ratio^{(t)}(i, l_i)$ from all objects.
$\delta^{(t)}_{obj}(i, l)$	The extra update cost (or synchronization) of object v_i between its current location l'' and location l at time slot t when a new DT replica of v_i is placed at cloudlet l.
$Init_i$	The instantiation cost of a DT replica of v_i.
θ	The predefined timestamp threshold.

computing capacity C_l. If a DT replica (item) of object v_i with $U(i) = \{u(j)\}$ is placed in cloudlet (bin) l, it consumes computing resource $comp(DT_i)$; if the delay requirement of u_j is satisfied, it incurs a cost $comp(DT_i) + cost_{obj}(v_i) + cost_{user}(u_j)$, which consists of the DT replica deployment cost, the DT replica synchronization cost of v_i, and the service request cost of u_j; otherwise incurs a cost ∞. We aim to minimize the total cost, subject to the computing capacity on each cloudlet. It can be seen that this special DT replica placement problem is equivalent to the minimum-cost GAP. Therefore, The DT replica placement problem in $G = (N, E)$ is NP-hard.

Theorem 2: the dynamic DT replica placement problem in $G = (N, E)$ is NP-hard.

Proof: We show the NP hardness of the dynamic DT replica placement problem, by reducing the well-known NP-hard problem - the maximum-profit GAP to a special case of the problem where the monitoring period consists of a single time slot only.

There are $|N|$ bins with each bin j having capacity C_j, and there is a set of items to be assigned to the bins. If item i is assigned to bin j, item i consumes the amount $size(i, j)$ of computing resource of bin j and will bring a profit $profit(i, j)$. *The maximum-profit GAP* is to assign as many items as possible to the bins such that the total profit of assigned items is maximized, subject to the computing resource capacity on each bin [3].

We show that an instance of the maximum-profit GAP can be reduced to an instance of the dynamic DT replica placement problem at one time slot. We consider a special case of the online DT replica placement problem where each user does not have a stringent delay requirement, and thus can be admitted by requesting a service from any replica of its requested object. We assume each cloudlet (bin) l with computing capacity C_l. If a DT replica (item) of object v_i is placed in cloudlet (bin) l, it consumes the amount $comp(DT_i)$ of computing resource and obtains a profit of $profit(i, l)$ $(= |U(i)|)$ as all users in $U(i)$ can be admitted. We aim to maximize the number of users admitted by deploying DT replicas of objects, subject to the computing capacity on each cloudlet. It can be seen that the special online DT replica placement problem at one time slot is equivalent to the maximum-profit GAP. As the maximum-profit GAP is NP-hard, so is the dynamic DT replica placement problem.

IV. Algorithms for the DT Replica Placement Problem

In this section, we consider the DT replica placement problem under the assumption that the mobility profiles of both objects and users are given in advance. We formulate an ILP solution for the problem if the problem size is small or medium; otherwise we develop a novel randomized algorithm with high probability for it. We also analyze the approximation ratio and the upper bound on resource violations of the proposed randomized algorithm.

A. ILP Formulation

We assume that the mobility information of each object and each user are given in advance, i.e., the probabilities of $p_{i,l}$ and $q_{j,l}$ of each object v_i and each user u_j at each location (an AP location) are given, which can be estimated through analyzing the historical movement traces of objects and users, by leveraging data mining and machine learning techniques on their DT data [5], [8], [25]. We further assume that there is sufficiently accumulated computing resource in the MEC network to accommodate all DT replicas such that all user requests can be admitted no matter where the users are located. We aim to minimize the total service cost of all user request admissions, which consists of (1) the DT replica synchronization cost of each object at each location, (2) the resource consumption cost of DT replica placements, and (3) the processing cost of responding user requests. We say that a request is admitted if the end-to-end service delay of the request is no greater than its delay requirement.

The optimization objective of the DT placement problem is to minimize

$$Cost = w \cdot \left(cost_{DT} + \sum_{v_i \in V} cost_{obj}(v_i)\right) + (1 - w) \cdot \sum_{u_j \in U} cost_{user}(u_j), \tag{8}$$

where w is a constant coefficient to reflect the cost relationship between the cost of DT replica deployments, and object updating, and the service cost of users.

The total cost defined in (8) consists of three components: the first and second components are related to the data updating of objects, the updated data of an object must be synchronized with all its DT replicas, and each DT replica placement consumes a certain amount of computing resource, resulting in the resource consumption cost and DT instantiation cost. The third component is the service cost of all mobile users when their requested DT replicas have been placed to proper locations while meeting their delay requirements. The optimization objective (8) can be equivalently rewritten as follows.

$$\begin{aligned} Cost = {} & w \sum_{v_i \in V} \sum_{l=1}^{|N|} \left(\mu \cdot comp\left(DT_i\right) + Init_i\right) x_{i,l} \\ & + w \sum_{v_i \in V} \sum_{l'=1}^{|N|} p_{i,l'} \sum_{l=1}^{|N|} \\ & \times \left(vol(v_i) cost(Path_{l',l}) + \frac{\kappa_i \cdot vol(v_i)}{f(DT_i)}\right) x_{i,l} \\ & + (1 - w) \sum_{u_j \in U} \sum_{l'=1}^{|N|} q_{j,l'} \sum_{l=1}^{|N|} \\ & \times \left(vol(u_j) cost\left(Path_{l',l}\right) + \frac{\kappa_i \cdot vol(u_j)}{f(DT_i)}\right) y_{j,l',l} + C, \end{aligned} \tag{9}$$

where $C = w \sum_{v_i \in V} \sum_{l'=1}^{|N|} p_{i,l'} upload\_cost(v_i, l')$ $+ (1 - w)$ $\sum_{u_j \in U} \sum_{l'=1}^{|N|} q_{j,l'} upload\_cost(u_j, l')$ is a constant.

Recall that $y_{j,l',l}$ is a binary variable indicating whether user u_j at location l' will use the DT replica at cloudlet l for its service processing or not. Note that for each pair (j, l'), if $q_{j,l'} \neq 0$, then there is only one l such that $y_{j,l',l} = 1$ and the rest $y_{j,l',l''} = 0$

when $l'' \neq l$, in other words, the DT replica of $v_{b(j)}$ located at cloudlet l will be used by the request of user u_j located at l' for its service processing. Clearly, there must be a DT replica of object $v_{b(j)}$ placed in cloudlet l already, i.e., $x_{b(j),l} = 1$. Thus, $x_{b(j),l} \geq y_{j,l',l}$ must hold when $y_{j,l',l} = 1$. An ILP formulation for the DT replica placement problem is given as follows.

$$\text{Minimize} \quad Cost \tag{10}$$

$$\text{s.t.} \quad \sum_{v_i \in V} comp(DT_i) x_{i,l} \leq C_l \quad \forall c_l \in N, \tag{11}$$

$$\frac{vol(u_j)}{B(u_j, AP_{l'})} + (vol(u_j) d(Path_{l',l}) + \frac{vol(u_j)}{f(DT_i)}) y_{j,l',l} \leq D_j, \forall u_j \in U,\ \forall c_{l'}, c_l \in N, \tag{12}$$

$$x_{b(j),l} \geq y_{j,l',l}, \forall u_j \in U,\ \forall c_{l'} \in N,\ c_l \in N \tag{13}$$

$$\sum_{l=1}^{|N|} y_{j,l',l} = 1, \quad \forall u_j \in U,\ \forall c_{l'} \in N, q_{j,l'} \neq 0 \tag{14}$$

$$x_{i,l} \in \{0,1\} \quad \forall v_i \in V,\ \forall c_l \in N, \tag{15}$$

$$y_{j,l',l} \in \{0,1\} \quad \forall u_j \in U,\ \forall c_l, c_{l'} \in N. \tag{16}$$

Constraint (11) ensures that the total computing resource consumption for all DT replicas at any cloudlet is no more than its computing capacity. Constraint (12) ensures that the end-to-end delay of user u_j at location $AP_{l'}$ by using the DT replica at location c_l is no greater than its delay requirement. Constraint (13) implies that if user u_j located at $AP_{l'}$ uses the DT replica located at c_l as its service, then the DT replica must be placed at location c_l already, i.e., $x_{b(j),l} \geq y_{j,l',l}$. Constraint (14) says that any user u_j at any location $AP_{l'}$, its request can only be serviced at most by one DT replica of its requested object $v_{b(j)}$ at location c_l.

B. Randomized Algorithm

As the ILP solution is not attainable when the problem size is large, we instead devise a randomized algorithm based on the linear relaxation of the ILP solution, by relaxing the value range of binary variables $x_{i,l}$ and $y_{j,l',l}$ to real numbers between 0 and 1. Specifically, an optimal fractional solution for the Linear Programming (LP) relaxation of the ILP solution can be obtained in polynomial time. An integral solution for the ILP is then derived by randomly rounding on the fractional solution with the specified probability [24], and this integral solution finally becomes a feasible solution for the original problem with high probability.

The randomized algorithm for the DT replica placement problem is detailed in Algorithm 1. Let $\tilde{x}_{i,l}$ and $\tilde{y}_{j,l',l}$ be the values of variables $x_{i,l}$ and $y_{j,l',l}$ in the linear relaxation LP of the ILP, respectively, where $\tilde{x}_{i,l} \in [0,1]$ and $\tilde{y}_{j,l',l} \in [0,1]$. In the rounding procedure, as a user u_j can only choose a DT replica that has been placed, $\hat{y}_{j,l',l}$ is set to 1 with probability $\frac{\tilde{y}_{j,l',l}}{\tilde{x}_{i,l}}$ when $\hat{x}_{i,l} = 1$. Note that we have $\frac{\tilde{y}_{j,l',l}}{\tilde{x}_{i,l}} \leq 1$ due to constraint $x_{b(j),l} \geq y_{j,l',l}$ in (13).

Algorithm 1: Randomized Algorithm for the DT Replica Problem.

```
Input: An MEC network $G(N, E)$, and a set $U$ of mobile
    users with the mobility probability $q_{j,l}$ of user $u_j \in U$ to
    location $c_l \in N$ is given, a set $V$ of mobile objects with the
    mobility probability $p_{i,l}$ of object $v_i \in V$ moving to
    location $c_l \in N$ is given.
Output: For each object, place a proper number of its DT
    replicas in different cloudlets such that the sum of the total
    cost of DT placements and updating of objects, and the
    total service cost of all users is minimized.
1: Calculate the shortest paths $Path_{l,l'}$ between each pair
     of APs $l$ and $l'$.
2: Solve the linear relaxation LP of the ILP (10) in
     polynomial time;
3: Let $\widetilde{OPT}$ be the optimal solution of the LP, and let $\tilde{x}_{i,l}$
     and $\tilde{y}_{j,l',l}$ be the values of variables $x_{i,l}$ and $y_{j,l',l}$
     respectively, where $\tilde{x}_{i,l} \in [0,1]$ and $\tilde{y}_{j,l',l} \in [0,1]$;
4: An integral solution that consists of $\hat{x}_{i,l}$ and $\hat{y}_{j,l',l}$ for the
     ILP (10) is obtained by the randomized rounding
     approach in [24]. That is, $\hat{x}_{i,l}$ is set to 1 with probability
     $\tilde{x}_{i,l}$, and $\hat{y}_{j,l',l}$ is set to 1 with probability $\frac{\tilde{y}_{j,l',l}}{\tilde{x}_{i,l}}$ if
     $\hat{x}_{i,l} = 1$;
5: return A candidate integral solution $\hat{S}$ based on $\hat{x}_{i,l}$ and
     $\hat{y}_{j,l',l}$ will be a feasible solution to the ILP with high
     probability.
```

C. Analysis of the Randomized Algorithm

The rest is to analyze the approximation ratio of Algorithm 1 and bound the computing resource violation on each cloudlet and the delay requirement violation for each user.

We first analyze the upper bound on the cost of the solution delivered by Algorithm 1. Let $\widetilde{OPT}$ and OPT be the optimal solutions of the LP and ILP (10), respectively. The value of $\widetilde{OPT}$ is a lower bound on the optimal solution of the ILP as this is a minimization optimization problem, i.e., $\widetilde{OPT} \leq OPT$. Denote by

$$\mathbb{P}_{i,l} = w\left(\mu \cdot comp(DT_i) + Init_i\right) + w \sum_{l'=1}^{|N|} p_{i,l'} \left(vol(v_i) cost\left(Path_{l',l}\right) + \frac{\kappa_i \cdot vol(v_i)}{f(DT_i)} \right)$$

$$\mathbb{Q}_{j,l',l} = (1-w) q_{j,l'} \times \left(vol(u_j) cost\left(Path_{l',l}\right) + \frac{\kappa_i \cdot vol(u_j)}{f(DT_i)} \right),$$

where $\mathbb{P}_{i,l}$ is the DT replicas deployment cost and update cost of object v_i at location l, and $\mathbb{Q}_{j,l',l}$ is the service cost of user u_j at location l' by using the DT replica located at l. Denote by Γ, which is defined as follows and will be used in the analysis.

$$\begin{aligned}\Gamma = \max\{ & \max\{\mu \cdot comp(DT_i) + Init_i \mid v_i \in V\}, \\ & \max\{\mathbb{Q}_{j,l',l} \mid u_j \in U, c_l, c_{l'} \in N\}, \\ & \max\left\{\frac{vol(u_j)}{B(u_j, AP_{l'})} + vol(u_j) d(Path_{l',l}) + \frac{vol(u_j)}{f(DT_i)} \right| \\ & \quad u_j \in U, c_{l'} \in N\}, \\ & \max\{C_l \mid c_l \in N\}, \max\{\mathbb{P}_{i,l} \mid v_i \in V, c_l \in N\}\}. \end{aligned} \tag{17}$$

By Algorithm 1, the probabilities of $\hat{x}_{i,l}$ and $\hat{y}_{j,l',l}$ to 1 are $\mathbf{Pr}[\hat{x}_{i,l} = 1] = \tilde{x}_{i,l}$ and $\mathbf{Pr}[\hat{y}_{j,l',l} = 1] = \mathbf{Pr}[\hat{y}_{j,l',l} = 1 \mid \hat{x}_{i,l} = 1]\mathbf{Pr}[\hat{x}_{i,l} = 1] = \frac{\tilde{y}_{j,l',l}}{\tilde{x}_{i,l}} \cdot \tilde{x}_{i,l} = \tilde{y}_{j,l',l}$, respectively. The expected total cost of the solution delivered by the randomized algorithm then is

$$\mathbb{E}\left(\sum_{v_i \in V}\sum_{c_l \in N} \mathbb{P}_{i,l} \cdot \hat{x}_{i,l} + \sum_{j=1}^{|U|}\sum_{l'=1}^{|N|}\sum_{l=1}^{|N|} \mathbb{Q}_{j,l',l} \cdot \hat{y}_{j,l',l}\right)$$
$$= \sum_{v_i \in V}\sum_{c_l \in N} \mathbb{P}_{i,l} \cdot \tilde{x}_{i,l} + \sum_{j=1}^{|U|}\sum_{l'=1}^{|N|}\sum_{l=1}^{|N|} \mathbb{Q}_{j,l',l} \cdot \tilde{y}_{j,l',l}.$$

Let $\tilde{\eta}_1 = \sum_{v_i \in V}\sum_{c_l \in N} \mathbb{P}_{i,l} \cdot \tilde{x}_{i,l}$ and $\tilde{\eta}_2 = \sum_{j=1}^{|U|}\sum_{l'=1}^{|N|}\sum_{l=1}^{|N|} \mathbb{Q}_{j,l',l} \cdot \tilde{y}_{j,l',l}$. We then have $\widetilde{OPT} = \tilde{\eta}_1 + \tilde{\eta}_2 + C$. Correspondingly, we have $OPT = \eta_1 + \eta_2 + C$.

Lemma 1: The value of $\sum_{v_i \in V}\sum_{c_l \in N} \mathbb{P}_{i,l} \cdot \hat{x}_{i,l}$ is no more than $2 \cdot \eta_1$ with high probability of $1 - \frac{1}{|V|}$.

Proof: Let α_1 be a constant with $\alpha_1 \leq 1$. By Chernoff Bounds [24], we have

$$\begin{aligned} & \mathbf{Pr}\left[\sum_{v_i \in V}\sum_{c_l \in N} \mathbb{P}_{i,l}\hat{x}_{i,l} \geq (1+\alpha_1)\eta_1\right] \\ & \leq \mathbf{Pr}\left[\sum_{v_i \in V}\sum_{c_l \in N} \mathbb{P}_{i,l} \cdot \hat{x}_{i,l} \geq (1+\alpha_1)\tilde{\eta}_1\right], \text{ since } \tilde{\eta}_1 \leq \eta_1 \\ & = \mathbf{Pr}\left[\sum_{v_i \in V}\sum_{c_l \in N} z_{i,l} \geq (1+\alpha_1)\frac{\tilde{\eta}_1}{\Gamma}\right], \text{ let } z_{i,l} = \frac{\mathbb{P}_{i,l} \cdot \hat{x}_{i,l}}{\Gamma} \\ & = \mathbf{Pr}\left[\sum_{v_i \in V} Z_i \geq (1+\alpha_1)\frac{\tilde{\eta}_1}{\Gamma}\right], \text{ let } Z_i = \sum_{c_l \in N} z_{i,l} \\ & \leq \exp\left(\frac{-{\alpha_1}^2 \cdot \tilde{\eta}_1}{(2+\alpha_1)\cdot\Gamma}\right), \text{ for all } \alpha_1 > 0, \end{aligned} \tag{18}$$

where (18) is obtained by the Chernoff bound as variables $Z_1, Z_2, \ldots, Z_{|V|}$ are independent random variables. Variable Z_i is a random variable derived from the integral solution $\hat{x}_{i,l}$. The value range of Z_i is within [0, 1] as $\frac{\mathbb{P}_{i,l} \cdot \hat{x}_{i,l}}{\Gamma} \leq \frac{\mathbb{P}_{i,l} \cdot \hat{x}_{i,l}}{\max\{\mathbb{P}_{i,l} \mid v_i \in V,\ c_l \in N\}}$. The value of Z_i is $\frac{\mathbb{P}_{i,l}}{\Gamma}$ with probability $\tilde{x}_{i,l}$; otherwise, Z_i is 0 with probability $1 - \tilde{x}_{i,l}$. Then $\mathbb{E}[\sum_{v_i \in V} Z_i] = \sum_{v_i \in V} \frac{\mathbb{P}_{i,l} \cdot \tilde{x}_{i,l}}{\Gamma} = \frac{\tilde{\eta}_1}{\Gamma}$.

We further assume that $\exp(\frac{-{\alpha_1}^2 \cdot \tilde{\eta}_1}{3\Gamma}) \leq \frac{1}{|V|}$. If $0 < \alpha_1 \leq 1$, then $\exp(\frac{-{\alpha_1}^2 \cdot \tilde{\eta}_1}{(2+\alpha_1)\cdot\Gamma}) \leq \exp(\frac{-\alpha_1^2 \cdot \tilde{\eta}_1}{3\Gamma}) \leq \frac{1}{|V|}$. Since $\tilde{\eta}_1 \leq \eta_1 \leq OPT$, we then have

$$\alpha_1 \geq \sqrt{\frac{3\Gamma \ln(|V|)}{\tilde{\eta}_1}} \geq \sqrt{\frac{3\Gamma \ln(|V|)}{\eta_1}} \geq \sqrt{\frac{3\Gamma \ln(|V|)}{OPT}}. \tag{19}$$

We must have $OPT \geq 3\Gamma \ln 2|V|$ by (19) as $\alpha_1 \leq 1$. The ratio of $\sum_{v_i \in V}\sum_{c_l \in N} \mathbb{P}_{i,l} \cdot \hat{x}_{i,l}$ to η_1 is 2 with high probability of $1 - \frac{1}{|V|}$, due to that $1 + \alpha_1 \leq 2$.

Lemma 2: The value of $\sum_{j=1}^{|U|}\sum_{l'=1}^{|N|}\sum_{l=1}^{|N|} \mathbb{Q} \cdot \hat{y}_{j,l',l}$ is no more than $2 \cdot \eta_2$ with high probability of $1 - 1/|U|$.

Proof: Let α_2 be a constant with $\alpha_2 \leq 1$. By Chernoff Bounds [24], we have

$$\begin{aligned} & \mathbf{Pr}\left[\sum_{j=1}^{|U|}\sum_{l'=1}^{|N|}\sum_{l=1}^{|N|} \mathbb{Q}_{j,l',l} \cdot \hat{y}_{j,l',l} \geq (1+\alpha_2)\eta_2\right] \\ & \leq \mathbf{Pr}\left[\sum_{j=1}^{|U|}\sum_{l'=1}^{|N|}\sum_{l=1}^{|N|} \mathbb{Q}_{j,l',l} \cdot \hat{y}_{j,l',l} \geq (1+\alpha_2)\tilde{\eta}_2\right], \\ & \qquad \text{since } \tilde{\eta}_2 \leq \eta_2 \\ & = \mathbf{Pr}\left[\sum_{j=1}^{|U|}\sum_{l'=1}^{|N|}\sum_{l=1}^{|N|} a_{j,l',l} \geq (1+\alpha_2)\frac{\tilde{\eta}_2}{\Gamma}\right], \\ & \qquad \text{let } a_{j,l',l} = \frac{\mathbb{Q}_{j,l',l} \cdot \hat{y}_{j,l',l}}{\Gamma} \\ & = \mathbf{Pr}\left[\sum_{j=1}^{|U|} A_j \geq (1+\alpha_2)\frac{\tilde{\eta}_2}{\Gamma}\right], \text{ let } A_j = \sum_{l'=1}^{|N|}\sum_{l=1}^{|N|} a_{j,l',l} \\ & \leq \exp\left(\frac{-{\alpha_2}^2 \cdot \tilde{\eta}_2}{(2+\alpha_2)\cdot\Gamma}\right), \text{ for all } \alpha_2 > 0, \end{aligned} \tag{20}$$

where (20) is obtained by the Chernoff bound as variables $A_1, A_2, \ldots, A_{|U|}$ are independent random variables. Variable A_j is a variable derived from the integral solution $\hat{y}_{j,l',l}$. The value range of A_j is within [0, 1] as $\frac{\mathbb{Q}_{j,l',l} \cdot \hat{y}_{j,l',l}}{\Gamma} \leq \frac{\mathbb{Q}_{j,l',l} \cdot \hat{y}_{j,l',l}}{\max\{\mathbb{Q}_{j,l',l} \mid u_j \in U, c_l, c_{l'} \in N\}}$. The value of A_j is $\frac{\mathbb{Q}_{j,l',l}}{\Gamma}$ with probability $\tilde{y}_{j,l',l}$; otherwise, A_j is 0 with probability $1 - \tilde{y}_{j,l',l}$. Then $\mathbb{E}[\sum_{u_j \in U} A_j] = \sum_{u_j \in U} \frac{\mathbb{Q}_{j,l',l} \cdot \tilde{y}_{j,l',l}}{\Gamma} = \frac{\tilde{\eta}_2}{\Gamma}$. We further assume that

$$\exp\left(\frac{-{\alpha_2}^2 \cdot \tilde{\eta}_2}{3\Gamma}\right) \leq \frac{1}{|U|} \quad \text{with } \alpha_2 > 0. \tag{21}$$

Also, if $0 < \alpha_2 \leq 1$, then (21) can be transformed as $\exp(\frac{-{\alpha_2}^2 \cdot \tilde{\eta}_2}{(2+\alpha_2)\cdot\Gamma}) \leq \exp(\frac{-\alpha_2^2 \cdot \tilde{\eta}_2}{3\Gamma}) \leq \frac{1}{|U|}$. Since $\tilde{\eta}_2 \leq \eta_2 \leq OPT$, we have

$$\alpha_2 \geq \sqrt{\frac{3\Gamma \ln(|U|)}{\tilde{\eta}_2}} \geq \sqrt{\frac{3\Gamma \ln(|U|)}{\eta_2}} \geq \sqrt{\frac{3\Gamma \ln(|U|)}{OPT}}. \tag{22}$$

We thus must have $OPT \geq 3\Gamma\ln(|U|)$ by (22). The ratio of $\sum_{j=1}^{|U|}\sum_{l'=1}^{|N|}\sum_{l=1}^{|N|}\mathbb{Q}_{j,l',l}\cdot\hat{y}_{j,l',l}$ to η_2 is 2 with high probability of $1-\frac{1}{|U|}$ due to that $1+\alpha_2 \leq 2$.

Theorem 3: Given an MEC network $G(N,E)$, a set U of user requests and a set V of objects, there is a randomized algorithm, Algorithm 1, with high probability of $\min\{1-\frac{1}{|U|}, 1-\frac{1}{|V|}, 1-\frac{1}{|N|}\}$, for the DT replica placement problem. The expected approximation ratio of Algorithm 1 is 2, at the expense of no more than twice of the computing capacity violation on any cloudlet and of the delay requirement violation of any user request, provided that $OPT \geq 3\Gamma\max\{\ln|U|,\ \ln|V|\}$, $\min\{C_l \mid c_l \in N\} \geq 6\Gamma\ln|N|$, and $\min\{D_j \mid u_j \in U\} \geq 6\Gamma\ln|U|$, where Γ is defined in (17), OPT is the optimal solution of the problem.

Proof: We first show the approximation ratio of the randomized algorithm, Algorithm 1, and then show that the computing resource capacity violation on each cloudlet and the delay violation of each user request are upper bounded.

By Lemma 1 and Lemma 2, let $\alpha = \max\{\alpha_1,\alpha_2\}$, then

$$\begin{aligned}&\mathbf{Pr}\left[\sum_{v_i\in V}\sum_{c_l\in N}\mathbb{P}_{i,l}\cdot\hat{x}_{i,l}+\sum_{j=1}^{|U|}\sum_{l'=1}^{|N|}\sum_{l=1}^{|N|}\mathbb{Q}_{j,l',l}\cdot\hat{y}_{j,l',l}\right.\\&\qquad\left.+C\geq(1+\alpha)OPT\right]\\&\leq\mathbf{Pr}\left[\sum_{v_i\in V}\sum_{c_l\in N}\mathbb{P}_{i,l}\cdot\hat{x}_{i,l}+\sum_{j=1}^{|U|}\sum_{l'=1}^{|N|}\sum_{l=1}^{|N|}\mathbb{Q}_{j,l',l}\cdot\hat{y}_{j,l',l}\right.\\&\qquad\left.+C\geq(1+\alpha)\widetilde{OPT}\right],\ \text{since }\widetilde{OPT}\leq OPT\\&\leq\min\left\{\frac{1}{|V|},\frac{1}{|U|}\right\},\end{aligned}\tag{23}$$

The solution delivered by the randomized algorithm for the DT replica placement problem is twice the optimal one with high probability $\min\{\ 1-\frac{1}{|V|},\ 1-\frac{1}{|U|}\ \}$ when $0<\alpha\leq 1$.

We then analyze the computing resource capacity violation on each cloudlet $c_l \in N$ by Algorithm 1. Let $g_{i,l} = \frac{comp(DT_i)}{\Gamma}\cdot\hat{x}_{i,l}$ be a random variable derived from the integral solution $x_{i,l}$. It can be seen that the value of $g_{i,l}$ is $\frac{comp(DT_i)}{\Gamma}$ with probability of $\tilde{x}_{i,l}$; otherwise 0 with probability $1-\tilde{x}_{i,l}$.

$$\mathbb{E}\left[\sum_{v_i\in V}g_{i,l}\right]=\sum_{v_i\in V}\frac{comp(DT_i)\cdot\tilde{x}_{i,l}}{\Gamma}=\frac{\tilde{C}_l}{\Gamma},\tag{24}$$

where $\tilde{C}_l$ is the amount of computing resource consumed at cloudlet c_l in the solution of the LP, and $\tilde{C}_l \leq C_l$.

By Algorithm 1, the computing resource consumption on cloudlet c_l is $\sum_{v_i\in V} comp(DT_i)\cdot\hat{x}_{i,l}$. Let $\beta>0$ be a constant with $\beta\leq 1$. Since there are $|N|$ cloudlets, the probability of the computing resource capacity violation on any cloudlet $c_l\in N$ is

$$\begin{aligned}&\mathbf{Pr}\left[\bigvee_{c_l\in N}\sum_{v_i\in V}comp(DT_i)\hat{x}_{i,l}\geq(1+\beta)C_l\right]\\&\leq\mathbf{Pr}\left[\bigvee_{c_l\in N}\sum_{v_i\in V}comp(DT_i)\hat{x}_{i,l}\geq(1+\beta)\tilde{C}_l\right],\ \text{since }\tilde{C}_l\leq C_l\\&\leq\sum_{c_l\in N}\mathbf{Pr}\left[\sum_{v_i\in V}g_{i,l}\geq(1+\beta)\frac{\tilde{C}_l}{\Gamma}\right],\\&\quad\text{by }g_{i,l}=\frac{comp(DT_i)\hat{x}_{i,l}}{\Gamma}\ \&\ \text{the union bound inequality},\\&\leq|N|\exp\left(\frac{-\beta^2\,\tilde{C}_l}{(2+\beta)\Gamma}\right),\ \text{for }\beta>0,\\&\qquad\text{by (24) and Chernoff Bounds.}\end{aligned}\tag{25}$$

Let $\exp(\frac{-\beta^2\tilde{C}_l}{(2+\beta)\Gamma})\leq\frac{1}{|N|^2}$. If $0<\beta\leq 1$, we have $\exp(\frac{-\beta^2\cdot\tilde{C}_l}{3\Gamma})\leq\frac{1}{|N|^2}$. Then,

$$\beta\geq\sqrt{\frac{6\Gamma\ln|N|}{\tilde{C}_l}}\geq\sqrt{\frac{6\Gamma\ln|N|}{C_l}},\ \text{since }\tilde{C}_l\leq C_l.\tag{26}$$

As $\beta\leq 1$, we must have $C_l\geq 6\Gamma\ln|N|$ by (26). To ensure that $C_l\geq 6\Gamma\ln|N|$ for every cloudlet $c_l\in N$, we have $\min\{\ C_l\ \mid\ c_l\in N\}\geq 6\Gamma\ln|N|$. We thus have $\mathbf{Pr}[\bigvee_{c_l\in N}\sum_{v_i\in V}comp(DT_i)\cdot x_{i,l}\ \geq\ (1+\beta)\cdot C_l]\ \leq\ |N|\cdot\frac{1}{|N|^2}=\frac{1}{|N|}$. The computing resource consumption on any cloudlet $c_l\in N$ is no more than twice its capacity, with high probability $1-\frac{1}{|N|}$ due to the fact that $1+\beta\leq 2$.

We finally show the delay requirement violation. With Constraint (12), the end-to-end delay of admitting request $u_j\in U$ by the randomized algorithm is no greater than its delay requirement D_j. Let $\widetilde{D_j}$ be the total delay experienced by user request $u_j\in U$, which is obtained in the LP relaxation of ILP formulation (10). A new random variable $d_{j,l',l}$ within the interval [0,1] is derived from random variable $\hat{y}_{j,l',l}$, with the expectation of the end-to-end delay of admitting request u_j, where

$$d_{j,l',l}=\frac{vol(u_j)\cdot(\frac{1}{B(u_j,AP_{l'})}+d(Path_{l',l})+\frac{1}{f(DT_i)})\cdot\hat{y}_{j,l',l}}{\Gamma}.$$

Given a constant $\gamma\in(0,1]$, the probability of violating the delay requirement of any request $u_j\in U$ by exceeding twice of its delay requirement D_j is given as follows.

$$\begin{aligned}&\mathbf{Pr}\left[\bigvee_{j=1}^{|U|}\sum_{l'=1}^{|N|}\sum_{l=1}^{|N|}\left(\frac{vol(u_j)}{B(u_j,AP_{l'})}+vol(u_j)\left(d\left(Path_{l',l}\right)\right.\right.\right.\\&\qquad\left.\left.\left.+\frac{1}{f\left(DT_i\right)}\right)\hat{y}_{j,l',l}\right)\geq 2D_j\right],\\&\leq\mathbf{Pr}\left[\bigvee_{j=1}^{|U|}\sum_{l'=1}^{|N|}\sum_{l=1}^{|N|}\left(\frac{vol(u_j)}{B\left(u_j,AP_{l'}\right)}+vol(u_j)\left(d\left(Path_{l',l}\right)\right.\right.\right.\\&\qquad\left.\left.\left.+\frac{1}{f\left(DT_i\right)}\right)\hat{y}_{j,l',l}\right)\geq(1+\gamma)\,D_j\right]\\&\leq\mathbf{Pr}\left[\bigvee_{j=1}^{|U|}\sum_{l'=1}^{|N|}\sum_{l=1}^{|N|}vol(u_j)\left(\frac{1}{B\left(u_j,AP_{l'}\right)}+d\left(Path_{l',l}\right)\right.\right.\end{aligned}$$

$$+\frac{1}{f(DT_i)}\Big)\hat{y}_{j,l',l} \geq (1+\gamma)\widetilde{D_j}\Big], \text{ since } \widetilde{D_j} \leq D_j$$

$$\leq \sum_{u_j \in U} \mathbf{Pr}\left[\sum_{l'=1}^{|N|}\sum_{l=1}^{|N|} d_{j,l',l} \geq (1+\gamma)\frac{\widetilde{D_j}}{\Gamma}\right],$$

by the union bound inequality

$$= \sum_{u_j \in U} \mathbf{Pr}\left[\sum_{c_{l'} \in N} Y_{j,l'} \geq (1+\gamma)\frac{\widetilde{D_j}}{\Gamma}\right], \text{ let } Y_{j,l} = \sum_{l'=1}^{|N|} d_{j,l',l}$$

$$\leq \sum_{u_j \in U} \exp\left(\frac{-\widetilde{D_j}\gamma^2}{3\Gamma}\right), \text{ by the Chernoff Bounds [24]}$$

$$\leq |U| \exp\left(\frac{-\min\{\widetilde{D_j} \mid u_j \in U\}\gamma^2}{3\Gamma}\right) \tag{27}$$

Let $\exp(\frac{-\min\{\widetilde{D_j} \mid u_j \in U\}\cdot\gamma^2}{3\Gamma}) \leq \frac{1}{|U|^2}$. If $0 < \gamma \leq 1$, it then can be transformed as $\gamma \geq \sqrt{\frac{6\Gamma\cdot\ln|U|}{\min\{\widetilde{D_j} \mid u_j \in U\}}} \geq \sqrt{\frac{6\Gamma\cdot\ln|U|}{\min\{D_j \mid u_j \in U\}}}$, since $\widetilde{D_j} \leq D_j$. With $0 < \gamma \leq 1$, we have $1 \geq \sqrt{\frac{6\Gamma\cdot\ln|U|}{\min\{D_j \mid u_j \in U\}}}$, i.e., $\min\{D_j \mid u_j \in U\} \geq 6\Gamma \cdot \ln|U|$. Hence, the probability of any admitted request with a delay more than twice of its delay requirement is no greater than $\frac{1}{|U|}$. The theorem then follows.

V. Online Algorithm for the DT Replica Placement Problem

In this section, we consider the dynamic DT replica placement problem under the mobility of both objects and users. We start by the basic idea of the proposed algorithm. We then devise an online algorithm for the problem, in which an efficient prediction mechanism based on the DT timestamp concept is developed for some DT replica reservations. We finally show that the solution delivered by the online algorithm is feasible, and analyze its time complexity.

A. Overview of the Algorithm

Let $U^{(t)}$ be the set of user requests arrived in the beginning of time slot t, we admit or reject requests in $U^{(t)}$ at time slot t immediately, and our ultimate goal is to maximize the number of user requests admitted for a finite time horizon T. Specifically, within each time slot $t \in T$, existing users may leave from the network or move to a new location, and new users may join the network at any location. Similarly, each object can move from one location to another location as well. Under such a dynamic setting, we propose online algorithms for the dynamic DT replica placement problem.

Since users can join and depart from the network, the DT replicas need to be removed and instantiated accordingly to admit as many requests as possible while meeting the delay requirements of admitted user requests. Note that both the processing of user requests and the deployment of new DT replicas incur cost. We refer to user request $u_j \in U^{(t)}$ as admitted with a service cost $cost^{(t)}_{user}(u_j, l', l)$, which is defined as follows.

$$cost^{(t)}_{user}(u_j, l', l) = vol(u_j, t)\left(\frac{\zeta \cdot P(u_j)}{B(u_j, AP_{l'})} + cost(Path_{l',l}) + \frac{\kappa_i}{f(DT_i)}\right), \tag{28}$$

where user u_j located at l' chooses the DT replica placed at cloudlet l for its service provisioning, and uploads the volume $vol(u_j, t)$ of its data for requesting DT_i service. When a new DT replica of object v_i is deployed at l at time slot $t \in T$, its cost $\delta^{(t)}_{obj}(i, l)$ is given as follows.

$$\delta^{(t)}_{obj}(i, l) = vol(v_i, t)\left(\frac{\zeta \cdot P(v_i)}{B(v_i, AP_{l''})} + cost(Path_{l'',l}) + \frac{\kappa_i}{f(DT_i)}\right) + \mu \cdot comp(DT_i) + Init_i, \tag{29}$$

$\delta^{(t)}_{obj}(i, l)$ is the extra update cost (or synchronization) of object v_i between its current location l'' and location l at time slot t when a new DT replica of v_i is placed at cloudlet l, and $Init_i$ is the instantiation cost of a DT replica of v_i.

Let $x^t_{i,l}$ be a binary variable that indicates whether a DT replica of object v_i is newly instantiated at AP l at time slot $t \in T$ ($x^t_{i,l} = 1$) or not ($x^t_{i,l} = 0$). Let $y^t_{j,l',l}$ be a binary variable to indicate whether the DT replica located at c_l will be chosen by user $u_j \in U^{(t)}$ located at $c_{l'}$ for request processing at time slot $t \in T$ ($y^t_{j,l',l} = 1$) or not ($y^t_{j,l',l} = 0$). The objective of the online DT replicas placement problem is to maximize the number of admitted requests while minimizing the total cost of admitted requests, i.e.,

$$\text{maximize} \sum_{t \in T}\left(\sum_{u_j \in U^{(t)}}\sum_{l=1}^{|N|} y^t_{j,l',l} - \gamma \times \left(\sum_{v_i \in V}\sum_{l=1}^{|N|} \delta^{(t)}_{obj}(i,l)\cdot x^t_{i,l} + \sum_{u_j \in U^{(t)}}\sum_{l=1}^{|N|} cost^{(t)}_{user}(u_j, l', l)\cdot y^t_{j,l',l}\right)\right),$$

where γ is a constant to balance between the number of requests admitted and the total service cost incurred for their admissions.

B. Online Algorithm

We observe that if a DT replica is not used by any of requests at the current time slot, then it will be removed and its computing resource will be released back to the system in the end of the current time slot. However, in an extreme case where the removed DT replica may be re-instantiated again at the next time slot, due to that new requests request its service again. Such a removal and re-instantiation of a DT replica could happen at every second time slot, which results in a significant overhead on the re-instantiation cost of a specific DT replica.

To mitigate frequent removals and re-instantiations of existing DT replicas, we propose an efficient method on the removal of DT replicas in the end of each time slot as follows. We do not remove an unused DT replica at the current time slot immediately. Instead, we assign each DT replica a timestamp when it is instantiated with a value of 1, and the value range of each timestamp is between 0 and a predefined integer threshold θ. If a DT replica is not used at the current time slot, its timestamp increases by one. A DT replica is removed at a time slot only if the value of its timestamp strictly greater than θ, i.e., the DT replica is removed at the current time slot if it has not been used by any requests in the past θ consecutive time slots. Otherwise, if a DT replica with timestamp $\theta > 0$ is requested by a user request at the current time slot, then its timestamp is reset to zero. This procedure continues until either all requests in $U^{(t)}_{non}$ are admitted, or no further DT replica placements will satisfy any of requests in $U^{(t)}_{non}$ any more.

If a user request can be served by an existing DT replica, the request will be assigned to that DT replica. Otherwise, a DT replica of the requested object by the request is chosen and placed to a cloudlet if the ratio of the accumulative benefit (the number of requests for that service) of admitted requests to the total cost that consists of the user service cost and the extra update cost is maximized. In other words, a DT replica of an object is placed to a location if the cost per user request admission by the DT placement is minimized.

The algorithm proceeds iteratively. Within each time slot t, for each user request in set $U^{(t)}$, if it can be served by an existing DT replica, it is assigned to that DT replica. If there are multiple ones, the one with the minimum service cost is chosen. The rest is to admit user requests in $U^{(t)}$ that have not been admitted by instantiating new DT replicas for them such that as many requests as possible can be admitted until all requests are admitted, or none of the remaining requests can be admitted without violating its delay requirement no matter where its requested DT replica is placed.

Recall that $U^{(t)}$ is the set of users to be considered at time slot $t \in T$. We first identify users in $U^{(t)}$ whose delay requirements can be met by requesting services from existing DT replicas. We then remove those DT replicas that none of users in $U^{(t)}$ makes use of their DT services to leave room for new DT replica placements. We finally determine to place which new DT replicas at each cloudlet to maximize the number of user requests admitted while minimizing their total admission cost.

Assume that there are DT replicas of some objects deployed in the MEC network after time slot $t-1$. The set $U^{(t)}$ can be partitioned into two disjoint sets $U^{(t)}_{exist}$ and $U^{(t)}_{non}$, where each user $u_j \in U^{(t)}_{exist}$ can make use of an existing DT replica for its service while meeting its delay requirement, and each user $u_j \in U^{(t)}_{non}$ cannot find any existing DT replica to satisfy its service delay requirement. If there are multiple DT replicas for a user in $U^{(t)}_{exist}(i)$, the one with the minimum service cost is chosen. Recall that the service cost of user request u_j located at l' choosing the DT replica placed at cloudlet l for its service provisioning is $cost^{(t)}_{user}(u_j, l', l)$ defined in (28). For each user $u_j \in U^{(t)}_{non} = \bigcup_{v_i \in V} U^{(t)}_{non}(i)$, it can be shown that none of existing DT replicas of object v_i can serve it without violating its delay requirement. We thus aim to admit as many requests in $U^{(t)}_{non}$ as possible if there is sufficient computing resource for instantiating new DT replicas for the requests. A greedy algorithm for the dynamic DT replica placement problem is presented as follows.

For each cloudlet $c_l \in N$, if a DT replica in it has not been used by any user $u_j \in U^{(t)}_{exist}$, it then can be removed and its occupied computing resource can be released back to the system. For a given object $v_i \in V$, let $N_{i,t}$ be the set of cloudlets with sufficient computing resource to instantiate a DT replica of v_i in it at time slot t. It can be seen that each cloudlet in $N_{i,t}$ does not contain any DT replica of v_i yet. As each cloudlet has limited residual computing resource to deploy DT replicas, to determine which DT replicas to be deployed in a cloudlet, we define a metric, which is the ratio of the number of admitted users by deploying the DT replica at a cloudlet to the total service cost of users using the deployed DT replica for their service provisioning.

Denote by $ratio^{(t)}(i,l)$ the ratio of the number $|U^{(t)}_{non}(i,l)|$ of admitted users in $U^{(t)}_{non}(i)$ using the DT replica of v_i deployed in cloudlet $c_l \in N_{i,t}$ for their services to the total cost of their admissions, i.e.,

$$ratio^{(t)}(i,l) = \frac{|U^{(t)}_{non}(i,l)|}{(1-w)\sum_{u_j \in U^{(t)}_{non}(i,l)} cost^{(t)}_{user}(j,l',l) + w\delta^{(t)}_{obj}(i,l)}, \tag{30}$$

where $U^{(t)}_{non}(i,l)$ ($\subseteq U^{(t)}_{non}(i)$) is the set of users can be admitted by requesting service from the DT replica of object v_i at location l, and $\delta^{(t)}_{obj}(i,l)$ is the cost of newly deployed DT replica at location l for object v_i that is defined in (29). Recall that w is a constant to reflect the cost relationship between the cost of DT replica deployment and object updating and the service cost of users.

The maximum ratio of deploying a DT replica of object v_i among all potential cloudlets in N is given by

$$ratio^{(t)}(i,l_i) = \max_{c_l \in N_{i,t}} \left\{ ratio^{(t)}(i,l) \right\}. \tag{31}$$

We then choose one ratio with the maximum value from all objects, i.e.,

$$ratio^{(t)}(i_0, l_{i_0}) = \max_{v_i \in V} \left\{ ratio^{(t)}(i,l_i) \right\} \tag{32}$$

We finally deploy a DT replica of v_{i_0}, DT_{i_0}, at cloudlet l_{i_0} and admit all user requests in $U^{(t)}_{non}(i_0, l_{i_0})$. Then, $U^{(t)}_{non}(i) = U^{(t)}_{non}(i) \setminus U^{(t)}_{non}(i_0, l_{i_0})$. Let $U^{(t)}_{admit}$ be the set of admitted requests so far. Then, $U^{(t)}_{admit} = \bigcup_{v_i \in V} U^{(t)}_{exist}(i)$ initially, which is updated by $U^{(t)}_{admit} = U^{(t)}_{admit} \cup U^{(t)}_{non}(i_0, l_{i_0})$.

The detailed online algorithm is given in Algorithm 2.

C. Algorithm Analysis

The rest is to show the correctness and analyze the time complexity of online algorithm Algorithm 2.

Lemma 3: Following Algorithm 2, for any object $v_i \in V$, (i) if a DT replica of object v_i deployed in cloudlet $c_l \in N$ has

Algorithm 2: Online Algorithm for the Dynamic DT Replica Placement Problem.

```
Input: An MEC $G = (N, E)$, a set $V$ of mobile objects, and a set
  $U = \cup_{t=1}^{(t)} U^{(t)}$ of mobile users, a given integer timestamp threshold
  $\theta > 0$. In the beginning of each time slot $t$, the locations $l'$ and $l''$ of
  mobile user $u_j \in U^{(t)}$ and each mobile object $v_i$ are given.
Output: A solution to maximize the number of requests admitted for the
  time horizon $T$ while minimizing the total service cost of admitted
  requests.
 1: $U_{admit} \leftarrow \emptyset$;
 2: for each time slot $t$ do
 3:   $U^{(t)}_{admit} \leftarrow \emptyset$;
 4:   for each object $v_i \in V$ do
 5:     $N_{i,t} \leftarrow N$; /* the set of cloudlets in which the DT replica of $v_i$ can
        be placed */
 6:     Calculate $U^{(t)}_{exist}(i)$ $(\subseteq U^{(t)}(i))$;
 7:     Reset the timestamps of DT replicas to 0 s if they are used by
        $U^{(t)}_{exist}(i)$; otherwise increase their timestamps by one;
 8:     $U^{(t)}_{non}(i) \leftarrow U^{(t)}(i) \setminus U^{(t)}_{exist}(i)$;
 9:     Calculate $cost_{user}(u_j, l', l)$ for each $u_j \in U^{(t)}_{exist}(i)$;
10:     $N_{i,t} \leftarrow N_{i,t} \setminus \{c_l \mid \text{if } u_j \in U^{(t)}_{exist} \text{ and } DT_i \text{ at } c_l\}$;
11:     $U^{(t)}_{admit} \leftarrow U^{(t)}_{admit} \cup U^{(t)}_{exist}(i)$
12: end for ;
13: for each cloudlet $c_l \in N$ do
14:     for each object $v_i \in V$ do
15:       Remove the DT replica of $v_i$ in $c_l$ if its timestamp is greater than
          $\theta$; $N_{i,t} \leftarrow N_{i,t} \setminus \{c_l\}$
16:     end for ;
17:     Calculate the available computing resource $C'_l$ of cloudlet $c_l$;
18:     for each object $v_i \in V$
19:       if $C'_l < comp(DT_i)$ then
20:         $N_{i,t} \leftarrow N_{i,t} \setminus \{c_l\}$
21:       end if ;
22:     end for ;
23:   end for ;
24:   while $\exists u_j \in \cup_{v_i \in V} U^{(t)}_{non}(i)$ is admissible do
25:     for each object $v_i \in V$ do
26:       for each cloudlet $c_l \in N_{i,t}$ do
27:         Calculate $ratio^{(t)}(i, l)$ by (30) if $U^{(t)}_{non}(i) \neq \emptyset$
28:       end for ;
29:       $ratio^{(t)}(i, l_i) \leftarrow \arg\max_{c_l \in N_{i,t}} \{ratio^{(t)}(i, l)\}$
30:     end for ;
31:     $ratio^{(t)}(i_0, l_{i_0}) \leftarrow \arg\max_{v_i \in V} \{ratio^{(t)}(i, l_i)\}$;
32:     Deploy a DT replica of $v_{i_0}$ into cloudlet $l_{i_0}$ and its timestamp is set
        0;
33:     $N_{i_0,t} \leftarrow N_{i_0,t} \setminus \{c_{l_{i_0}}\}$; $C'_{l_{i_0}} \leftarrow C'_{l_{i_0}} - comp(DT_{i_0})$;
34:     $U^{(t)}_{admit} \leftarrow U^{(t)}_{admit} \cup U^{(t)}_{non}(i_0, l_{i_0})$;
35:     $U^{(t)}_{non}(i_0) \leftarrow U^{(t)}_{non}(i_0) \setminus U^{(t)}_{non}(i_0, l_{i_0})$
36:   end while ;
37:   $U_{admit} \leftarrow U_{admit} \cup U^{(t)}_{admit}$;
38:   Admit requests in $U^{(t)}_{admit}$ and reject requests in $U^{(t)} \setminus U^{(t)}_{admit}$
39: end for ;
40: return $U_{admit}$.
```

not been used by any request in $U^{(t)}_{exist}(i)$, then it would not be used by any request in $U^{(t)}_{non}(i)$; (ii) assume that $U^{(t)}_{non}(i) \neq \emptyset$ in the end of the algorithm, then, for any cloud $c_l \in N_{i,t}$, none of requests in $U^{(t)}_{non}(i)$ can be served by the DT replica of v_i placed in c_l while meeting their delay requirements. For any cloudlet $c_l \in N \setminus N_{i,t}$, none of requests in $U^{(t)}_{non}(i)$ can be admitted without violating its delay requirement.

Proof: We first show Case (i). If the DT replica can be used by a user u_j in $U^{(t)}_{non}(i)$, then following the definition, $u_j \in U^{(t)}_{exist}(i)$, not in $U^{(t)}_{non}(i)$.

We then prove Case (ii). Following the definition of $N_{i,t}$, each cloudlet in it has sufficient computing resource to accommodate the DT replica of v_i. Furthermore, only a DT replica of v_i that is placed in a cloudlet in $N_{i,t}$ can be used by user requests in $U^{(t)}_{non}(i)$. If a user request in $U^{(t)}_{non}(i)$ cannot be admitted if and only if its end-to-end service delay is larger than its delay requirement, no matter which cloudlet in $N_{i,t}$ that the DT replica is placed. The lemma then follows.

Theorem 4: Given a finite time horizon T, an MEC network $G = (N, E)$ with a set $\mathcal{AP}$ of APs and a set N of cloudlets with each $c_l \in N$ having computing capacity C_l, a set $U^{(t)}$ of mobile users requesting services from DT replicas, and a set V of mobile objects updating their DT replica states, there is an online algorithm, Algorithm 2, for the dynamic DT replica placement problem, which delivers a solution within $O((|U^{(t)}|)^2 \cdot |V| \cdot |N|)$ time per time slot t with $1 \leq t \leq T$.

Proof: Following Algorithm 2, At each iteration of the `while` loop, only a DT replica instance can be instantiated in a cloudlet if the cloudlet has sufficient computing resource to implement at least one user request while meeting the use delay requirement. Also, the computing resources occupied by the removed DT replicas will be released back to the system. Since the computing resource capacity on each cloudlet is not violated in any iteration, the solution delivered by Algorithm 2 is feasible.

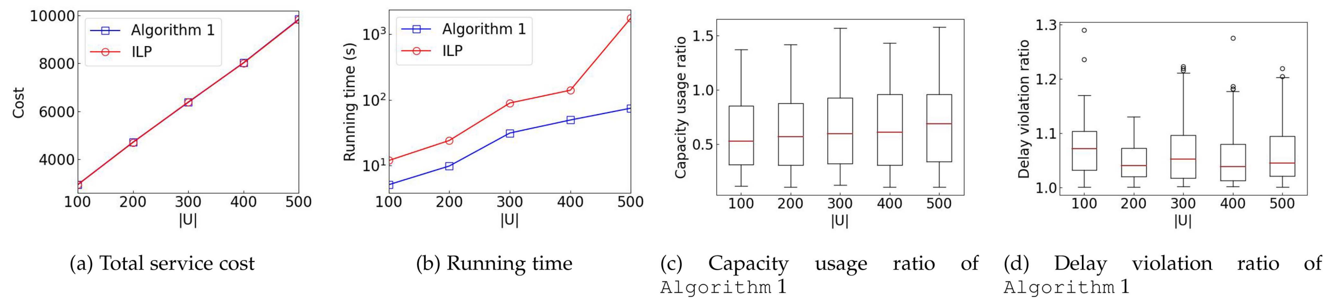
The time complexity of Algorithm 2 is analyzed as follows. In each iteration, Algorithm 2 takes $O(|U^{(t)}| \cdot |V| \cdot |N|)$ time to admit one user request. Since there are $|U^{(t)}|$ requests, Algorithm 2 takes $O((|U^{(t)}|)^2 \cdot |V| \cdot |N|)$ time.

VI. Performance Evaluation

In this section, we evaluated the performance of the proposed algorithms for the (dynamic) DT replica placement problem through experimental simulations. We also investigate the impact of parameters, the threshold θ, the number of users, and the value of w, on the performance of proposed algorithms.

A. Experimental Settings

We generate each MEC network instance via NetworkX [26]. The computing capacity C_l of each cloudlet $c_l \in N$ is randomly selected from the range in [2000, 4000] MHz, and the cost μ per unit computing resource (MHz) is \$0.02 [33]. The transmission delay $d(e)$ of per MB data on each link $e \in E$ is randomly chosen in the range of [0.02, 0.05] ms, and the transmission cost $\xi(e)$ of a unit data on link e varies from \$0.01 to \$0.04 per MB randomly [33]. The transmission powers $P(v_i)$ and $P(u_j)$ of each object v_i and each user u_j are drawn in the range between 12 dBm and 23 dBm [4]. The cost ζ of transmission power consumption per unit time is \$0.4. The uploading rate B of an object or a user is calculated by $B = W_l \log(1 + \frac{P \cdot H}{\rho})$, where



(a) Total service cost (b) Running time (c) Capacity usage ratio of Algorithm 1 (d) Delay violation ratio of Algorithm 1

Fig. 2. Performance comparison of Algorithm 1 and ILP by varying the number of users.

W_l is the bandwidth of AP l, P is the transmission power of the object or the user that is proportional to the distance between the object or the user and AP l, H is the channel power gain, and ρ is the noise power. The bandwidth W_l of AP l at each time slot is randomly drawn from 20 MHz to 40 MHz, and the signal-to-noise ratio $\frac{P(v_i)\cdot H}{\rho}$ for any object $v_i \in V$ and any user $u_j \in U$ is chosen from 10 dB to 30dB [14].

Objects and users are randomly distributed initially under APs in the network. The amount of computing resources $comp(DT_i)$ for deploying a DT replica of object v_i varies from 400 MHz to 800MHz [17]. The cost κ_i of DT_i data processing per unit time is randomly chosen from \$0.1 to \$0.5. The processing rate of a DT replica is proportional to the amount of computing resources allocated to it. We thus assume that the data processing rate of unit computing resource is 0.1 MB per millisecond. The instantiation cost $Init_i$ of deploying a DT replica for object $v_i \in V$ is drawn randomly between \$5 and \$10. The data volumes $vol(v_i)$ and $vol(u_j)$ of an object v_i and a user u_j are randomly drawn from the range of $[100, 200]$MB.

It is mentioned that the value of each figure is the average of results of 20 different topologies of MEC networks with the same size. The actual running time of each algorithm is based on a desktop equipped with an Intel(R) Xeon(R) Platinum 8280 2.70 GHz CPU, with 10 GB RAM. The parameters are adopted as the default settings unless otherwise specified.

B. Performance of Different Algorithms for the DT Replica Placement Problem

We first studied the performance of Algorithm 1 and the ILP solution for the DT replica placement problem. We consider 50 objects, and the delay requirement of each user is set from 15 ms to 50 ms randomly [14].

We vary the number of users $|U|$ while fixing the network size $|N|$ at 20. Meanwhile, we vary the network work size $|N|$ while fixing the number of users $|U|$ at 200. Fig. 2 plots the performance curves of Algorithm 1 and the ILP solution, respectively. From Fig. 2(a) it can be seen that the total cost of the solution by Algorithm 1 is amost equal to the cost of the ILP solution, while Fig. 2(b) indicates that the running itme of Algorithm 1 is only fractional one of the ILP one. It is noted that the ILP solution can provide DT services for all users while satisfying their delay requirements, but there are capacity and delay violations of the solution by Algorithm 1. Table II shows the percentages of capacity satisfied and violation by varying $|U|$. It can be seen from Table II that no more than 16.5% of cloudlets have their capacity violated. Fig. 2(c) depicts the capacity usage ratios of cloudlets by Algorithm 1, where the ratio is the amount of computing resource consumed by each cloudlet divided by its capacity.

TABLE II
CLOUDLET CAPACITY USAGE DELIVERED BY ALGORITHM 1 BY VARYING $|U|$

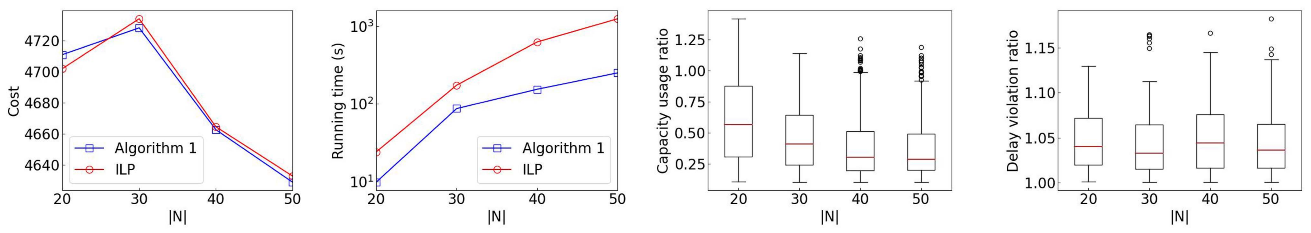
$|U|$	100	200	300	400	500
Satisfied	0.9025	0.8675	0.86	0.835	0.8425
Violated	0.0975	0.1325	0.14	0.165	0.1575

TABLE III
USER DELAYS DELIVERED BY ALGORITHM 1 BY VARYING $|U|$

$|U|$	100	200	300	400	500
Satisfied	0.9773	0.9841	0.9680	0.9682	0.9508
Violated	0.0012	0.0011	0.0012	0.0005	0.0011
No assigned	0.0215	0.0148	0.0308	0.0313	0.0481

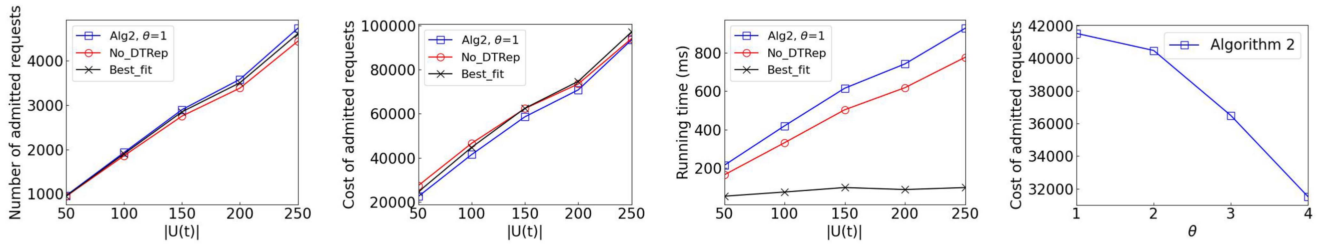
The delay satisfaction ratio of each user u_j is calculated by $\sum_{l\in N_j} q_{j,l}$, where the delay requirement of user u_j can be met when it stays at cloudlet $c_l \in N_j$ and $N_j \subseteq N$, and $q_{j,l}$ is the probability of user u_j moving to cloudlet c_l. The dissatisfaction of delay requirement for a user u_j moving to location $c_{l'}$ are either u_j has a longer service delay than its delay requirement; or u_j has no assigned DT at its current location. Table III displays the ratios of delay satisfaction, delay violation, and no assigned DTs, respectively, with varying $|U|$. It can be seen that the delay satisfaction ratios for given $|U|$ are all above 95%. We analyze the delay violations, and the results are shown in Fig. 2(d), where the service delay of a user is violated by no more than 30% of its delay requirement.

Fig. 3 plots the performance curves of Algorithm 1 and the ILP by varying network size. It can be seen from Fig. 3(a) and (b) that the solution of Algorithm 1 has the similar cost as the ILP solution with much less running time. Table IV shows the percentages of capacity violation by varying $|N|$. Fig. 3(c) plots the capacity usage ratios of cloudlets by Algorithm 1. With the increase on network size, the percentage of cloudlets with capacity violation decreases, the minimum, median, and maximum capacity usage ratios decrease too, since there are more cloudlets for DT deployment in a larger network. Table V



(a) Total service cost (b) Running time (c) Capacity usage ratio of `Algorithm` 1 (d) Delay violation ratio of `Algorithm` 1

Fig. 3. Performance comparison of Algorithm 1 and ILP by varying the network size.



(a) Number of admitted requests (b) Cost of admitted requests (c) Running time (d) Impact of θ on `Algorithm` 2

Fig. 4. Performance comparison of Algorithm 2 and `No_DTRep` by varying the number of users.

TABLE IV
CLOUDLET CAPACITY USAGE DELIVERED BY ALGORITHM 1 BY VARYING $|N|$

$\lvert N\rvert$	20	30	40	50
Satisfied	0.8675	0.96	0.9825	0.99
Violated	0.1325	0.04	0.0175	0.01

TABLE V
USER DELAYS DELIVERED BY ALGORITHM 1 BY VARYING $|N|$

$\lvert N\rvert$	20	30	40	50
Satisfied	0.9841	0.9879	0.9975	0.9925
Violated	0.0011	0.0008	0.0010	0.0005
No assigned	0.0148	0.0113	0.0015	0.0070

displays the delay related ratios by varying $|N|$, where the delay satisfaction ratio for a given $|N|$ is always above 98%, and the analysis of delay violations is shown in Fig. 3(d).

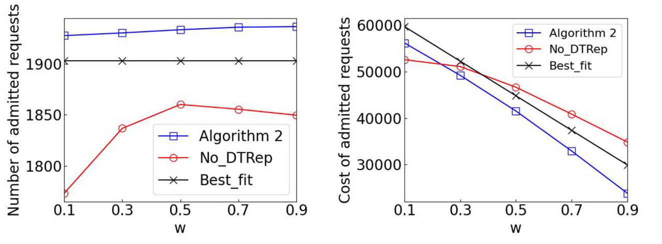
C. Performance of Different Algorithms for the Dynamic DT Replica Placement Problem

We then evaluated the performance of Algorithm 2 for the dynamic DT replica placement problem against two benchmarks. One benchmark is the one without any DT replica placement `No_DTRep`, i.e., there is only one DT placed for each object in `No_DTRep`. The other is referred to `Best_fit`, which assigns an incoming request to its requested DT replica (an existing one or instantiating a new one) at a random chosen cloudlet if there are no violations. We considered an MEC network with 30 APs and 50 objects. The delay requirement of each user is set from 5 ms to 30 ms randomly [14]. There are 20 time slots.

We evaluated the performance of Algorithm 2 and `No_DTRep`, and `Best_fit` by varying the number $|U(t)|$ of users. Fig. 4 plots the performance curves of the three comparison algorithms. It can be seen from Fig. 4(a) and (b) that Algorithm 2 admits more user requests than those by `No_DTRep` and `Best_fit` at less cost. Since `No_DTRep` only allows to deploy only one DT for each object, it needs frequently removing and instantiating DTs to admit requests distributed in different locations of the network while meeting their delay requirements. This leads to a higher instantiation cost. Algorithm 2 outperforms `Best_fit` in terms of both the number of admitted requests and the service cost, which justifies the advantages of the the ratio metric-based method.

What followed is to evaluate the impact of θ on the performance of Algorithm 2. Fig. 4(d) shows that the service cost of admitted requests decreases with the increase on the value of θ. The rationale behind lies in two aspects. On one hand, the increase on the value of θ reduces the overhead on re-instantiations of DT replicas. On the other hand, since the placements of DT replicas are not adjusted immediately at each time slot due to the movement of users and objects, the number of admitted user requests will decrease, resulting in the cost decrease of the admitted requests.

We also considered the impact of parameter w on the performance of Algorithm 2, `Best_fit`, and `No_DTRep` while fixing $|U(t)|$ at 100. Fig. 5 plots the performance curves of the three algorithms. Fig. 5(a) indicates that the number of admitted requests by Algorithm 2 is larger than those by `No_DTRep` and `Best_fit` for each different value of w. It can be seen from



(a) Number of admitted requests (b) Cost of admitted requests

Fig. 5. Performance comparison of Algorithm 2 and `No_DTRep` by varying the value of w.

Fig. 5(b) that the accumulative cost of the solutions by the three algorithms decreases with the growth on the value of w. This indicates that the service cost of users is higher than the DT updating cost of objects for the given setting.

VII. Conclusion

In this paper, we considered mobility-aware, delay-sensitive inference service provisioning in a DT-assisted MEC network with mobile objects and users, via DT replica placements. We formulated two novel DT replica placement problems: the DT replica placement problem and the dynamic DT replica placement problem, respectively. We first formulated an ILP solution for the former when the problem size is small or medium; otherwise we developed a randomized algorithm with high probability. We then proposed an online algorithm for the latter, which takes into account the overhead on frequent removals and re-instantiations of DT replicas, by developing an efficient prediction mechanism to reserve some existing DTs in order to reduce the service cost further. We finally conducted simulations to evaluate the performance of the proposed algorithms. Simulation results demonstrate that the proposed algorithms are promising against their comparison counterparts.

Acknowledgments

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

References

[1] L. Atzori, A. Iera, and G. Morabito, “The Internet of Things: A survey,,” *Comput. Netw.*, vol. 54, no. 15, pp. 2787–2805, 2010.
[2] O. Chukhno, N. Chukhno, G. Araniti, C. Campolo, A. Iera, and A. Molinaro, “Placement of social digital twins at the edge for beyond 5G IoT networks,” *IEEE Internet Things J.*, vol. 9, no. 23, pp. 23927–23940, Dec. 2022.
[3] R. Cohen, L. Katzir, and D. Raz, “An efficient approximation for the generalized assignment problem,” *Inf. Process. Lett.*, vol. 100, pp. 162–166, 2006.
[4] 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.
[5] I. Farris, T. Taleb, M. Bagaa, and H. Flick, “Optimizing service replication for mobile delay-sensitive applications in 5G edge network,” in *Proc. IEEE Int. Conf. Commun.*, 2017, pp. 1–6.
[6] D. Gupta, S. S. Moni, and A. S. Tosun, “Integration of digital twin and federated learning for securing vehicular Internet of Things,” in *Proc. Int. Conf. Res. Adaptive Convergent Syst.*, 2023, pp. 1–8.
[7] Y. Gong, Y. Wei, Z. Feng, F. R. Yu, and Y. Zhang, “Resource allocation for integrated sensing and communication in digital twin enabled Internet of Vehicles,” *IEEE Trans. Veh. Technol*, vol. 72, no. 4, pp. 4510–4524, Apr. 2023.
[8] K. Lei, M. Qin, B. Bai, G. Zhang, and M. Yang, “GCN-GAN: A non-linear temporal link prediction model for weighted dynamic networks,” in *Proc. IEEE Conf. Comput. Commun.*, 2019, pp. 388–396.
[9] L. Lei, G. Shen, L. Zhang, and Z. Li, “Toward intelligent cooperation of UAV swarms: When machine learning meets digital twin,” *IEEE Netw.*, vol. 35, no. 1, pp. 386–392, Jan./Feb. 2021.
[10] B. Li, Y. Liu, L. Tan, H. Pan, and Y. Zhang, “Digital twin assisted task offloading for aerial edge computing and networks,” *IEEE Trans. Veh. Technol*, vol. 71, no. 10, pp. 10863–10877, Oct. 2022.
[11] 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.
[12] J. Li et al., “Mobility-aware utility maximization in digital twin-enabled serverless edge computing,” *IEEE Trans. Comput.*, early access, Apr. 16, 2024, doi: 10.1109/TC.2024.3388897.
[13] J. Li et al., “Digital twin-enabled service provisioning in edge computing via continual learning,” *IEEE Trans. Mobile Comput.*, vol. 23, no. 6, pp. 7335–7350 , Jun. 2024.
[14] J. Li et al., “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.
[15] J. Li, J. Wang, Q. Chen, Y. Li, and A. Zomaya, “Digital twin-enabled service satisfaction enhancement in edge computing,” in *Proc IEEE Conf. Comput. Commun.*, 2023, pp. 1–10.
[16] Y. Li et al., “Data collection maximization in IoT sensor networks via an energy-constrained UAV,” *IEEE Trans. on Mobile Comput.*, vol. 22, no. 1, pp. 159–174, Jan. 2023.
[17] W. Liang, Y. Ma, W. Xu, Z. Xu, X. Jia, and W. Zhou, “Request reliability augmentation with service function chain requirements in mobile edge computing,” *IEEE Trans. Mobile Comput.*, vol. 21, no. 12, pp. 4541–4554, Dec. 2022.
[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.*, early access, Dec. 12, 2023, doi: 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. Lu, S. Maharjan, and Y. Zhang, “Adaptive edge association for wireless digital twin networks in 6G,” *IEEE Internet Things J.*, vol. 8, no. 22, pp. 16219–16230, Nov. 2021.
[21] Digital twin market size, share & trends analysis report by end-use (automotive & transport, retail & consumer goods, agriculture, manufacturing, energy & utilities), by region, and segment forecasts, 2023–2030. Accessed: Apr. 2023. [Online]. Available: https://www.researchandmarkets.com/reports/5415584/digital-twin-market-size-share-and-trends
[22] 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.
[23] E. F. Maleki, L. Mashayekhy, and S. M. Nabavinejad, “Mobility-aware computation offloading in edge computing using machine learning,” *IEEE Trans. Mobile Comput.*, vol. 22, no. 1, pp. 324–340, Jan. 2023.
[24] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2005.
[25] 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.
[26] NetworkX. Accessed: Jul. 2022. [Online]. Available: https://networkx.org/
[27] M. Polese, R. Jana, V. Kounev, K. Zhang, S. Deb, and M. Zorzi, “Machine learning at the edge: A data-driven architecture with applications to 5G cellular networks,” *IEEE Trans. Mobile Comput.*, vol. 20, no. 12, pp. 3367–3382, Dec. 2021.
[28] D. Shomys and E. Tardos, “An approximation algorithm for the generalized assignment problem,” *Math. Program.*, vol. 62, pp. 461–474, 1993.

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

[30] 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.

[31] S. Wang, Y. Guo, N. Zhang, P. Yang, A. Zhou, and X. Shen, "Delay-aware microservice coordination in mobile edge computing: A reinforcement learning approach," *IEEE Trans. Mobile Comput.*, vol. 20, no. 3, pp. 939–951, Mar. 2021.

[32] Z. Wang et al., "Mobility digital twin: Concept, architecture, case study, and future challenges," *IEEE Internet Things J.*, vol. 9, no. 18, pp. 17452–17467, Sep. 2022.

[33] Z. Xu, W. Liang, M. Huang, M. Jia, S. Guo, and A. Galis, "Efficient NFV-enabled multicasting in SDNs," *IEEE Trans. Commun.*, vol. 67, no. 3, pp. 2052–2070, Mar. 2019.

[34] Z. Xu, W Liang, M Jia, M. Huang, and G. Mao, "Task offloading with network function requirements in a mobile edge-cloud network," *IEEE Trans. Mobile Comput.*, vol. 18, no. 11, pp. 2673–2685, Nov. 2019.

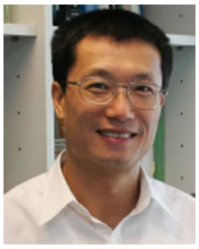
[35] R. Zhang, Z. Xie, D. Yu, W. Liang, and X. Chang, "Digital twin-assisted federated learning service provisioning over mobile edge networks," *IEEE Trans. Comput.*, vol. 73, no. 2, pp. 586–598, Feb. 2024.

[36] 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, May 2024.

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 currently a postdoc with the Department of Computer Science, City University of Hong Kong. She worked with Baidu, Beijing, China, from 2016 to 2017. Her research interests include mobile edge computing, network function virtualization, and optimization problems.

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

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



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