









Mobility-Aware Utility Maximization in Digital Twin-Enabled Serverless Edge Computing

Jing Li , Song Guo , *Fellow, IEEE*, Weifa Liang , *Senior Member, IEEE*, Jianping Wang , *Fellow, IEEE*, Quan Chen , *Member, IEEE*, Wenchao Xu , *Member, IEEE*, Kang Wei , *Member, IEEE*, and Xiaohua Jia , *Fellow, IEEE*

Abstract—Driven by data and models, the digital twin technique presents a new concept of optimizing system design, process monitoring, decision-making and more, through performing comprehensive virtual-reality interaction and continuous mapping. By introducing serverless computing to Mobile Edge Computing (MEC) environments, the emerging serverless edge computing paradigm facilitates the communication-efficient digital twin services, and promises agile, fine-grained and cost-efficient provisioning of limited edge resources, where serverless functions are implemented by containers in cloudlets (edge servers). However, the nonnegligible cold start delay of containers deteriorates the responsiveness of digital twin services dramatically and the perceived user service experience. In this paper, we investigate delay-sensitive query service provisioning in digital twin-empowered serverless edge computing by considering user mobility. With digital twins of users deployed in the remote cloud, referred to as primary digital twins, we deploy their digital twin replicas based

on serverless functions in cloudlets to mitigate the query service delay while enhancing user service satisfaction that is expressed as a utility function. We study two optimization problems with the aim of maximizing the accumulative utility gain: the digital twin replica placement problem per time slot, and the dynamic digital twin replica placement problem over a finite time horizon. We first formulate an Integer Linear Program (ILP) solution for the digital twin replica placement problem when the problem size is small; otherwise, we propose an approximation algorithm for the problem with a provable approximation ratio. We then design an online algorithm for the dynamic digital twin replica placement problem, and a performance-guaranteed online algorithm for a special case of the problem by assuming each user issues a query at each time slot. Finally, we evaluate the performance of the proposed algorithms for placing digital twin replicas in MEC networks through simulations. The results demonstrate the proposed algorithms are promising, outperforming their counterparts.

Index Terms—Digital twin, serverless computing, mobile edge computing, user mobility, approximation algorithm, online algorithm, user service satisfaction, delay-sensitive service, utility maximization, resource allocation and optimization.

Manuscript received 8 May 2023; revised 16 January 2024; accepted 10 April 2024. Date of publication 16 April 2024; date of current version 11 June 2024. The work of Jing Li, Song Guo, Weifa Liang, and Jianping Wang was supported by Hong Kong Research Grants Council (RGC) through the Collaborative Research Fund (CRF) under Grant C1042-23GF. The work of Song Guo was also supported in part by the Key-Area Research and Development Program of Guangdong Province under Grant 2021B0101400003; in part by Hong Kong RGC Research Impact Fund under Grant R5060-19 and Grant R5034-18; in part by the Areas of Excellence Scheme under Grant AoE/E-601/22-R; and in part by the General Research Fund under Grant 152203/20E, Grant 152244/21E, Grant 152169/22E, and Grant 152228/23E. The work of Weifa Liang was also supported by Hong Kong RGC under Grant CityU 7005845, Grant CityU 8730094, Grant CityU 9043510, and Grant CityU 9380137, respectively. The work of Quan Chen was supported in part by the NSFC under Grant 62372118, and in part by Guangdong Basic and Applied Basic Research Foundation under Grant 2024A1515030136. The work of Wenchao Xu was supported by Hong Kong RGC under Grant PolyU15222621 and Grant PolyU15225023. Recommended for acceptance by A. Karanth. (*Corresponding authors: Quan Chen; Song Guo.*)

Jing Li, Weifa Liang, Jianping Wang, and Xiaohua Jia are with the Department of Computer Science, City University of Hong Kong, Hong Kong 999077, P. R. China (e-mail: jing.li@cityu.edu.hk; weifa.liang@cityu.edu.hk; jianwang@cityu.edu.hk; csjia@cityu.edu.hk).

Song Guo is with the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Hong Kong 999077, P. R. China (e-mail: songguo@cse.ust.hk).

Quan Chen is with the School of Computers, Guangdong University of Technology, Guangzhou 510006, P. R. China, and also with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong 999077, P. R. China (e-mail: quan.c@gdut.edu.cn).

Wenchao Xu and Kang Wei are with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong 999077, P. R. China (e-mail: wenchao.xu@polyu.edu.hk; kangwei@polyu.edu.hk).

Digital Object Identifier 10.1109/TC.2024.3388897

I. INTRODUCTION

DUBBED “the next iteration of the internet”, the inception of the metaverse introduces a ubiquitous and immersive digital world, where users produce content, manage their digital assets, and communicate with each other, disruptively revolutionizing the lifestyle of people beyond the bounds of the physical world [1]. Postulated as indispensable components in the metaverse to digitalize the physical world, digital twins are the digital portraits of physical objects with their physical being and behaviors projected onto the metaverse [25]. Digital twins mirror their physical counterparts via real-time monitoring and synchronization, and enable vivid simulations for insightful forecasting and optimization [36]. Powered by digital twins, the network service provider tailors data gathering and analysis for individual users and network infrastructure with real-time and seamless physical-digital world interactions, painting a hyper-realistic and satisfactory experience for each individual user [9].

While impressive, the creation and maintenance of digital twins are data-intensive and acquire timely data analytics for continuous synchronization in real time [23], [24], [29], catering to the blossoming deployment of Internet of Things (IoT)

devices and applications [11], [22], [35]. Mobile Edge Computing (MEC) is deemed as one of the enabling techniques of the metaverse, and guarantees low delay in smooth task conduction and digital twin maintenance due to its geographical advantages, mitigating the resource limitations on end devices [4], [20]. Albeit with the benefits of MEC over traditional cloud computing, the cloudlets deployed at the network edge come amid the concern on resource poverty and inefficient management [12]. Following the Function-as-a-Service (FaaS) paradigm, serverless computing is anticipated to endow MEC with fine-grained and high-elasticity resource allocation, by leveraging serverless functions based on container technology [7]. Extending serverless computing to the network edge, the emerging paradigm, serverless edge computing, has received widespread attention [2], [13], [27], [37], [39].

In this paper, we study providing mobility-aware query services in serverless edge computing, built upon digital twin data. Especially, each user has a digital twin, referred to as *primary digital twin*, established in the remote cloud with abundant resources to simulate the states and behaviors of the user, thereby providing personalized and user-centric services. Through performing analysis and simulations on primary digital twins cooperatively by the powerful cloud, each primary digital twin of a user can offer query results to the user for guidance and decision-making, which, however, inevitably causes long query service delays due to the communication between users and the remote cloud [36]. Therefore, we deploy *digital twin replicas* of the primary digital twins in cloudlets, and each digital twin replica can be implemented by a serverless function for efficient management. Rather than querying the primary digital twin with a long service delay, a user can upload the raw data to a digital twin replica for analysis and obtain a query result with a low service delay. The raw data is then compressed by the digital twin replica and sent to the primary digital twin for synchronization.

Placing digital twin replicas in cloudlets to offer delay-sensitive query services to mobile users poses the following fundamental challenges.

(i) The first challenge is how to characterize the utility gain of augmenting user service satisfaction by deploying digital twin replicas for serving mobile users with reduced service delays, instead of users querying only the primary digital twins in the remote cloud?

(ii) A critical issue in serverless computing is the nonnegligible cold start delay caused by plenty of initialization operations to instantiate a container, which deteriorates the perceived user service experience. The mainstream method to address such an issue is to warm up existing containers, i.e., reusing existing containers, and the start delay can be reduced significantly. At each time slot of the time horizon, there is a set of users issuing queries for digital twin data, and the second challenge is how to determine the placement of digital twin replicas in cloudlets to maximize the utility gain of providing query services to users, by instantiating new containers or reusing existing containers?

(iii) Given a finite time horizon and dynamic query requests from users with high mobility, the final challenge is how to

maximize the accumulative utility gain over the time horizon without any future knowledge?

The novelties of this paper lie in handling the dynamic placement of digital twin replicas in MEC networks for users with dynamic mobility and queries, as well as introducing a novel metric to measure user satisfaction on digital twin-enabled query services. We formulate two optimization problems of placing digital twin replicas in MEC networks, and design approximation and online algorithms with guaranteed performance for the problems.

The main contributions of this paper are given as follows.

- We formulate two novel problems: the digital twin replica placement problem per time slot and the dynamic digital twin replica placement problem over a finite time horizon. We demonstrate their NP-hardness too.
- We provide an Integer Linear Program (ILP) solution for the digital twin replica placement problem when the problem size is small. Otherwise, we propose an approximation algorithm for the problem with a provable approximation ratio.
- We devise a performance-guaranteed online algorithm for a special case of the dynamic digital twin replica placement problem, via assuming that each user issues a query at each time slot. Furthermore, we develop an online algorithm for the dynamic digital twin replica placement problem.
- We evaluate the performance of the proposed algorithms through simulations. The results demonstrate the proposed algorithms are promising, outperforming their counterparts.

The rest of the article is arranged as follows. Section II surveys related studies. Section III includes problem definitions and provides NP-hardness proofs of the proposed problems. Section IV proposes an approximation algorithm for the digital twin replica placement problem, and Section V develops a performance-guaranteed online algorithm for a special dynamic digital twin replica placement problem, along with an online algorithm for the dynamic digital twin replica placement problem, respectively. Section VI provides the performance evaluation of the proposed algorithms. The article is concluded by Section VII.

II. RELATED WORK

Recent studies have recognized the advantages of serverless computing, and extended it to edge scenarios to realize serverless edge computing [2], [7], [13], [27], [28], [31], [37], [39]. Ascigil et al. [2] put efforts into optimizing the resource allocation for processing user requests within deadlines in serverless edge clouds, and proposed fully-decentralized algorithms. Ko et al. [13] designed a resource management framework based on per-function queues in serverless edge computing, and formulated a Markov decision process-based problem, with the aim to reduce the demanded memory resource of warm starting containers, whilst completing tasks within deadlines. Pan et al. [27] settled on a problem of caching containers in

serverless edge computing to minimize the system cost. To solve the problem, they proposed online algorithms by mapping the problem to the ski-rental problem. Through utilizing the extensive collected telemetry data, Pelle et al. [28] presented a novel mechanism for efficient placement and configuration of delay-sensitive serverless functions in an edge environment. Tang et al. [31] considered the competition for task scheduling, and formulated a problem based on a partially observable stochastic game in order to optimize the collected utility of each edge node in serverless edge computing. Xu et al. [37] concentrated on minimizing the age of the results of big data queries executed in serverless edge clouds. They devised an approximation algorithm to deal with the admission of a single query, and an online learning algorithm for multiple queries with dynamic arrivals. However, the above-mentioned studies did not consider adopting serverless edge computing to enhance the management of digital twin services.

The convergence of MEC and digital twin technologies introduces a new dimension in improving network service provisioning, and has attracted lots of attention recently [14], [15], [17], [18], [23], [25], [33], [41]. Li et al. [14], [15] considered a digital twin-assisted MEC network, and proposed efficient algorithms for reliable service function chain-enabled network services. Lin et al. [23] adopted an incentive-based strategy to deal with dynamic digital twin service demands to maximize the long-term profit, by leveraging the Lyapunov optimization technique. Liu et al. [25] studied edge collaboration in intelligent task offloading with the assistance of digital twins, and devised a learning-based algorithm to mask the consumed power and network delay. Wang et al. [33] implemented a hierarchical digital twin IoT framework in MEC based on the blockchain technology. They aimed to mitigate service delay and energy consumption, while achieving real-time computation with reliability and security. Zhang et al. [41] devised a deep deterministic policy gradient-based approach to mitigate the energy consumption for offloading tasks, caching services and allocating resources in a digital twin edge network. User or object mobility has also been investigated by existing research in providing digital twin services in MEC environments [9], [16], [21], [30], [42]. Gong et al. [9] considered the mobility of vehicles in a vehicle edge network, and proposed a holistic network virtualization mechanism in service-centric and user-centric manners by incorporating digital twins into network slicing. Li et al. [16], [21] took the mobility of physical objects into account, and proposed approximation and online algorithms to offer query results with low age of information to users in MEC under the static and dynamic digital twin placement schemes, respectively. Sun et al. [30] explored uncertain user mobility and system dynamics, and designed a mobile offloading framework in a digital twin edge network, where digital twins are utilized to estimate the states of edge servers. Zheng et al. [42] approached the synchronization issue between vehicles and their digital twins, through developing a learning-based algorithm for network selection. However, the above-mentioned studies considered neither deploying digital twin replicas in edge networks to provide satisfactory query services to users, nor applying serverless edge computing for

implementing digital twin replicas based on serverless functions to improve the resource efficiency at network edge.

Unlike the aforementioned works, in this paper, we focus on user satisfaction enhancement on query services in serverless edge computing, built upon digital twin data. We investigate the dynamic digital twin replica placement in MEC, where the digital twin replicas are implemented based on serverless functions, considering the dynamic user query arrivals and the dynamic mobility of users.

III. PRELIMINARIES

In this section, we introduce the system model, notions, notations, and problem definitions.

A. System Model

Consider an MEC network, modelled by an undirected graph $G = (V \cup \{v_0\}, E)$ with a set V of Access Points (APs), a set E of links connecting APs, and a remote cloud v_0 . Each AP is co-located with a cloudlet, while they are interconnected by an optical fiber cable with negligible communication delay between them [26]. We abuse notion $v \in V$ to denote an AP or its co-located cloudlet. We assume the main resource demand of a serverless function is the memory resource [39], according to serverless platforms [1], [3], [5], because the amount of data loaded to the memory is related to the amount of memory allocated to a serverless function [37], and the amount of computing resource allocated to a serverless function is proportional to the amount of memory allocated to the serverless function [8]. We thus only consider memory resource constraints on cloudlets for the implementation of digital twin replicas. Denote by M_v the amount of available memory resource in cloudlet $v \in V$. Let d_e be the communication delay for transmitting a unit of data along link $e \in E$ [39]. Each AP $v \in V$ is interconnected to the remote cloud v_0 via a gateway, and the communication delay between an AP and the remote cloud via the gateway is far longer than that between any pair of APs [19].

Suppose the monitoring time horizon of the MEC network is divided into equal *time slots*. Denote by $T = \{1, 2, \dots, |T|\}$ the set of time slots.

B. User Queries With User Mobility and Digital Twin Replicas

Let U be a set of users, which are highly mobile in the network during the time horizon T . There is a subset of users $U_t \subseteq U$ at each time slot $t \in T$, and each user $u \in U_t$ issues a query with uploading its query raw data with size $s(u, t)$ to the digital twin of the user for a query result, and the digital twin then returns the query result back to the user. The uploaded raw data will also facilitate the synchronization between the digital twin and the user, i.e., its physical object.

Suppose a digital twin of each user $u \in U$ has been deployed in the remote cloud v_0 , referred to as the *primary digital twin* of the user. Intuitively, user queries can be processed in their primary digital twins in the remote cloud, which, however, will cause long response delays. A viable solution is to deploy a

digital twin replica of the user in a cloudlet to provide delay-sensitive digital twin services to the user. Especially, instead of sending the raw data to the primary digital twin in the remote cloud, the raw data is sent to the digital twin replica in the cloudlet for analysis first, and the query result is then sent to the user. Meanwhile, the raw data is compressed with a much smaller volume at the digital twin replica, and then sent to the primary digital twin in the cloud for synchronization, thereby saving the bandwidth resource and communication costs. Notice that the query of a user can be processed in either a digital twin replica in the cloudlet or the primary digital twin in the cloud.

Following the serverless computing paradigm, we consider implementing each digital twin replica by a serverless function based on a container, which encapsulates the required packages of the digital twin replica. Denote by m_u the amount of demanded memory resource of a container for implementing the digital twin replica of user u .

Instantiating a container usually incurs a considerable delay, i.e., the cold start delay [27]. Because the required packages of containers for the same digital twin replica are considered to be identical, the container for a digital twin replica of a user at time slot t can be reused for another query of the user at the next time slot $t + 1$. E.g., suppose that a container for a digital twin replica is instantiated in a cloudlet to provide the query service to a user at time slot t . After processing the query, the service provider determines whether to keep or remove the container for the service. If the container is kept, the user issuing another query at the next time slot $t + 1$ can leverage the container for processing, i.e., warm start, and the warm start delay is negligible [27]. Note that primary digital twins in remote cloud for services are considered to be always in the warm states, because there are abundant computing and memory resources in clouds.

C. Delay Model

Because the query result usually is small, its downloading time is insignificant [19]. The service delay of a query composes (1) the communication delay from the user to the digital twin, and (2) the processing delay on the digital twin, defined as follows.

The communication delay from the user to the digital twin: Suppose a user $u \in U_t$ moves to location $v_{u,t}$ (i.e., covered by AP $v_{u,t} \in V$) at time slot t , and issues a query with the size $s(u, t)$ of the raw data. The communication delay of transmitting the raw data of user u to the digital twin replica in the cloudlet (or the primary digital twin in the remote cloud) $v \in V \cup \{v_0\}$ at time slot t is

$$d_{com}(u, v, t) = s(u, t) \cdot d(v_{u,t}, v), \quad (1)$$

where $d(v_{u,t}, v)$ is the communication delay of transmitting a unit of data through the shortest path (or the gateway) from the location $v_{u,t}$ of the user to node $v \in V \cup \{v_0\}$.

The processing delay of the digital twin: Denote by ρ_{u,v_0} the processing time of a unit of data by the primary digital twin of user u in remote cloud v_0 , and $\rho_{u,v}$ the processing time of

a unit of data by the digital twin replica of user u in cloudlet $v \in V$. We analyze the processing delay of the digital twin by distinguishing two cases as follows.

Case 1. The query of user u is processed by the digital twin replica in a cloudlet $v \in V$, and the container of the digital twin replica is newly instantiated, i.e., cold start. Denote by $d_{u,v,t}^{cold}$ the cold start delay of instantiating the container of the digital twin replica of user u in cloudlet $v \in V$ at time slot t . Then the processing delay of the digital twin replica of user u in cloudlet $v \in V$ at time slot t is

$$d_{proc}(u, v, t) = d_{u,v,t}^{cold} + s(u, t) \cdot \rho_{u,v}. \quad (2)$$

Case 2. The query of user u is processed by the primary digital twin in the remote cloud, or the digital twin replica in a cloudlet with a warm start, i.e., user u reuses the existing container of the digital twin replica. We can observe that the cold start delay is not considered, and the processing delay of the primary digital twin of user u in node $v \in V \cup \{v_0\}$ at time slot t is

$$d_{proc}(u, v, t) = s(u, t) \cdot \rho_{u,v}. \quad (3)$$

Let $I_{u,v,t}$ be a binary constant indicator, and $I_{u,v,t} = 1$ shows user $u \in U_t$ issues a query at the last time slot $t - 1$, with the query processed by a digital twin replica (or the primary digital twin) in node $v \in V \cup \{v_0\}$; otherwise, $I_{u,v,t} = 0$. We can see $I_{u,v,1} = 0, \forall u \in U_1, v \in V \cup \{v_0\}$ at the initial time slot 1, because no query is issued before the time horizon. We set $d_{u,v_0,t}^{cold} = 0$ for the remote cloud $v_0, \forall u \in U_t, \forall t \in T$. By Eq. (2) and (3), $\forall v \in V \cup \{v_0\}$, we have

$$d_{proc}(u, v, t) = (1 - I_{u,v,t}) \cdot d_{u,v,t}^{cold} + s(u, t) \cdot \rho_{u,v}. \quad (4)$$

Therefore, suppose a query is processed by a digital twin replica in the cloudlet (or the primary digital twin in the remote cloud) $v \in V \cup \{v_0\}$, the service delay of the query issued by user u at time slot t is

$$D(u, v, t) = d_{com}(u, v, t) + d_{proc}(u, v, t). \quad (5)$$

D. Utility Gain

Queries can always be processed by primary digital twins in remote cloud v_0 , which, however, usually leads to high query service delays. Through deploying digital twin replicas in cloudlets for query processing, such query service delays can be significantly reduced, and the user service satisfactions are augmented. In the following, we use the *utility gain* $H_{u,v,t}$ to capture the user service satisfaction augmentation of each user $u \in U_t$ issuing a query at time slot t when processing the query by the digital twin in node $v \in V \cup \{v_0\}$.

$$H_{u,v,t} = D(u, v_0, t) - D(u, v, t), \quad (6)$$

where $D(u, v_0, t)$ is the query service delay in the case that the query is processed by the primary digital twin of the user in remote cloud v_0 , and $D(u, v, t)$ is the query service delay when processing the query by the digital twin of the user in node $v \in V \cup \{v_0\}$.

Utility gain $H_{u,v,t}$ with $v \in V$ implies the query service delay reduction through deploying a digital twin replica for its user in cloudlet $v \in V$, and we have $H_{u,v_0,t} = 0, \forall u \in U_t, t \in T$.

TABLE I
TABLE OF SYMBOLS

Notations	Descriptions
$G = (V \cup \{v_0\}, E)$	In an MEC network G with v_0 the remote cloud, V is the set of APs (cloudlets) and E is the set of links between APs.
M_v	The available amount of memory resource in cloudlet $v \in V$.
d_e	The communication delay for transmitting a unit of data through link $e \in E$.
$T = \{1, 2, \dots, T \}$	The monitoring time horizon.
U and U_t	A set of users in the network, and a subset of users issuing queries at time slot t .
$s(u, t)$	The size of the query raw data of user t at time slot t .
m_u	The amount of demanded memory resource of a container for implementing the digital twin replica of user u .
$d(v_{u,t}, v)$	The communication delay of transmitting a unit of data through the shortest path (or the gateway) from the location $v_{u,t}$ of the user to node $v \in V \cup \{v_0\}$ at time slot t .
$d_{u,v,t}^{cold}$	The cold start delay of instantiating the container of the digital twin replica of user u in cloudlet $v \in V$ at time slot t .
$I_{u,v,t}$	A binary constant indicator showing whether user $u \in U_t$ issues a query at the last time slot $t - 1$, with the query processed by a digital twin replica (or the primary digital twin) in node $v \in V \cup \{v_0\}$.
$d_{com}(u, v, t)$	The communication delay of transmitting the raw data of the user u to the digital twin replica in the cloudlet (or the primary digital twin in the remote cloud) $v \in V \cup \{v_0\}$.
$d_{proc}(u, v, t)$	The processing delay of the digital twin replica of user u in cloudlet $v \in V$ at time slot t .
$D(u, v, t)$ and $H_{u,v,t}$	The service delay and the utility of processing the query of user u at time slot t by a digital twin replica in the cloudlet (or the primary digital twin in the remote cloud) $v \in V \cup \{v_0\}$.
$x_{u,v,t}$	A binary decision variable showing whether the query of user u is processed in node $v \in V \cup \{v_0\}$ at time slot t .
$d_s(u, v, t)$	The static delay of assigning the query of user $u \in U_t$ to node $v \in V \cup \{v_0\}$ at time slot t defined in Eq. (11).
$y_{u,v,t}$	A binary variable showing whether the query of user u is processed in cloudlet v involves the cold start at time slot t .
$\mathcal{D}_s(t)$ and $\mathcal{D}_d(t)$	The total static and dynamic delay of queries of users at time slot t .

E. Problem Definitions

We first consider the *digital twin replica placement problem* for a single time slot within the defined MEC network G as follows. Given the placed digital twin replicas and their containers in G in the last time slot $t - 1$, we determine the placement of digital twin replicas for issued queries by users in U_t at time slot t to enhance user satisfactions, through determining whether to reuse the deployed containers at time slot $t - 1$, considering memory resource constraints on cloudlets. Note that no digital twin replica is placed in cloudlets before the initial time slot 1. Therefore, the deployment of digital twin replicas in cloudlets at time slot 1 suffers from cold starts.

Definition 1: Given an MEC network $G = (V \cup \{v_0\}, E)$, a single time slot t , a set U_t of users issuing queries at time slot t with each query uploading the size $s(u, t)$ of raw data for analyzing, and a set V of cloudlets, the digital twin replica placement problem is to maximize the total utility gain of issued queries by users in U_t at time slot t , through determining the placement of digital twin replicas enabled by containers in G at time slot t , subject to memory capacities of cloudlets in V .

We then consider the *dynamic digital twin replica placement problem*. Given time horizon T , a set U of users move within the MEC network during the time horizon, and there is a set $U_t \subseteq U$ of users issuing queries at each time slot $t \in T$ without any knowledge on future user queries, i.e., the set U_t of users and their locations are revealed at the beginning of each time slot t . The dynamic digital twin replica placement problem is to dynamically determine whether to reuse existing containers or instantiate new containers for implementing digital twin replicas to maximize the accumulative user satisfactions during time horizon T without any future knowledge.

Definition 2: Given an MEC network $G = (V \cup \{v_0\}, E)$, a finite time horizon T and a set U of users moving in MEC network G within the time horizon T , a set U_t of user issue queries at each time slot $t \in T$ with each query uploading the size $s(u, t)$ of raw data for analyzing. Without any future knowledge, the *dynamic digital twin replica placement problem* is to maximize the accumulated utility gain of user queries over time horizon T , through dynamically determining the placement of digital twin replicas enabled by containers at each time slot $t \in T$, subject to memory capacities of cloudlets in V .

All symbols adopted in this paper are listed in Table I.

F. NP-Hardness of the Defined Problems

Theorem 1: The digital twin replica placement problem in an MEC network $G = (V \cup \{v_0\}, E)$ is NP-hard.

Proof: The NP-hardness of the digital twin replica placement problem is demonstrated via reducing from the NP-hard Generalized Assignment Problem (GAP) [6]. Given a set of bins with each possessing a capacity, there is a set of items with each possessing a weight, and we can collect a profit via packing an item into a bin. The GAP is to maximize the collected profit, subject to capacities of bins.

The detailed problem reduction is as follows. We treat remote cloud v_0 as a bin with an unlimited capacity, while each cloudlet $v \in V$ is treated as a bin possessing a capacity of M_v , i.e., its available memory resource. We also treat each user $u \in U_t$ as an item possessing a weight m_u , i.e., the consumed memory resource of a container to instantiate the digital twin replica of user u . If user u is allocated to cloudlet/remote cloud $v \in V \cup \{v_0\}$ for processing, a profit $H_{u,v,t}$ is obtained, and $H_{u,v,t}$ is its utility gain calculated by Eq. (6). We observe that the

digital twin replica placement problem is equivalent to a GAP [6], which is NP-hard. Thus, the digital twin replica placement problem is NP-hard. ■

Corollary 1: The dynamic digital twin replica placement problem in an MEC network $G = (V \cup \{v_0\}, E)$ is NP-hard.

Proof: We show that the digital twin replica placement problem is NP-hard by Theorem 1. Because the digital twin replica placement problem is a special case of the dynamic digital twin replica placement problem with a single time slot, the dynamic digital twin replica placement problem is also NP-hard. ■

IV. APPROXIMATION ALGORITHM FOR THE DIGITAL TWIN REPLICA PLACEMENT PROBLEM

In this section, we provide an Integer Linear Program (ILP) solution to the digital twin replica placement problem if the problem size is small. Otherwise, we develop an approximation algorithm for it, via reducing it to a Generalized Assignment Problem (GAP).

A. ILP Formulation

Let $x_{u,v,t}$ be a binary decision variable, where $x_{u,v,t} = 1$ means the query of user u is processed in a cloudlet or remote cloud $v \in V \cup \{v_0\}$ at time slot t , and $x_{u,v,t} = 0$ otherwise. Especially, if the query of user u is processed in a cloudlet $v \in V$ with $x_{u,v,t} = 1$, the query is assigned to a digital twin replica enabled by a container in cloudlet v . Otherwise (the query of user u is processed in the remote cloud v_0 with $x_{u,v_0,t} = 1$), the query is processed by the primary digital twin in the remote cloud.

The ILP for the digital twin replica placement problem at time slot t is formulated as follows.

$$\text{Maximize } \sum_{u \in U_t} \sum_{v \in V \cup \{v_0\}} H_{u,v,t} \cdot x_{u,v,t}, \quad (7)$$

subject to:

$$\begin{aligned} &\text{Eq. (1), Eq. (4), Eq. (5), Eq. (6),} \\ &\sum_{u \in U_t} m_u \cdot x_{u,v,t} \leq M_v, \quad \forall v \in V \end{aligned} \quad (8)$$

$$\sum_{v \in V \cup \{v_0\}} x_{u,v,t} = 1, \quad \forall u \in U_t \quad (9)$$

$$x_{u,v,t} \in \{0, 1\}, \quad \forall u \in U_t, \forall v \in V \cup \{v_0\}. \quad (10)$$

Objective (7) is the total utility gain collected by deploying digital twin replicas for users U_t in cloudlets, and the utility gain $H_{u,v,t}$ is defined by Eq. (6). Constraint (8) indicates the memory resource constraints of cloudlets. Constraint (9) means the query of each user in U_t is processed in either a cloudlet or a remote cloud. Constraint (10) means $x_{u,v,t}$ is a binary decision variable showing whether the query of user u is processed in a cloudlet (or remote cloud).

Algorithm 1 Approximation algorithm for the digital twin replica placement problem

Input: An MEC network $G = (V \cup \{v_0\}, E)$, a single time slot t , a set U_t of users issuing queries at time slot t with each query uploading the size $s(u, t)$ of raw data for analyzing.

Output: Maximize the total utility gain of users by deploying digital twin replicas in cloudlets V at time slot t .

- 1: Identify the communication delay between each pair of APs by finding the shortest paths between them;
- 2: Reduce the problem instance to a GAP instance, by generating an item i_u with the weight of m_u for each user in U_t , and a bin b_v for each cloudlet with the capacity of M_v , as well as a bin b_0 for the remote cloud v_0 with unlimited capacity. Assigning item i_u to a bin b_v with $v \in V \cup \{v_0\}$ brings a profit $H_{u,v,t}$ defined by Eq. (6);
- 3: Obtain a solution \mathcal{A}_t to the GAP instance via applying the approximation algorithm in [6];
- 4: **return** Solution \mathcal{A}_t to the digital twin replica placement problem at time slot t .

B. Approximation Algorithm

We devise an approximation algorithm for the digital twin replica placement problem, via reducing the problem to a GAP, and there is an approximation algorithm for it [6].

In the following, we show the detailed problem reduction. With regard to each cloudlet $v \in V$, there is a bin b_v with the capacity M_v , i.e., the available memory resource of cloudlet v . Meanwhile, remote cloud v_0 has a corresponding bin b_0 , and its capacity is unlimited. For each user $u \in U_t$, there is an item i_u with the weight of m_u , i.e., the consumed memory resource of a container to enable the digital twin replica of user u . Assigning item i_u to a bin b_v with $v \in V \cup \{v_0\}$ brings a profit $H_{u,v,t}$, i.e., the utility gain of assigning the query of user u to cloudlet/remote cloud v by Eq. (6).

The proposed algorithm for the digital twin replica placement problem is detailed in Algorithm 1.

C. Algorithm Analysis

We now analyze the approximation ratio and time complexity of Algorithm 1.

Theorem 2: Given an MEC network $G = (V \cup \{v_0\}, E)$, a single time slot t , and a set U_t of users issuing queries at time slot t , there is an approximation algorithm, Algorithm 1, for the digital twin replica placement problem, which delivers an approximate solution with a $\frac{1}{2+\epsilon}$ approximation ratio. The time complexity of Algorithm 1 is $O(|V|^3 + |V| \cdot |U_t| \cdot \log \frac{1}{\epsilon} + \frac{|V|}{\epsilon^4})$, where ϵ is a constant with $0 < \epsilon \leq 1$.

Proof: The approximation ratio of Algorithm 1 is obtained referring to the detailed analysis of the approximation algorithm from [6], i.e., the solution delivered by the algorithm is no less than $\frac{1}{2+\epsilon}$ times the optimal one.

The time complexity of Algorithm 1 is analyzed as follows. Identifying the shortest paths between each pair of APs takes $O(|V|^3)$ time, while the approximation algorithm in [6] takes $O(|V| \cdot |U_t| \cdot \log \frac{1}{\epsilon} + \frac{|V|+1}{\epsilon^4})$ time. Therefore, the time complexity of Algorithm 1 is $O(|V|^3 + |V| \cdot |U_t| \cdot \log \frac{1}{\epsilon} + \frac{|V|}{\epsilon^4})$. ■

V. ONLINE ALGORITHMS FOR THE DYNAMIC DIGITAL TWIN REPLICA PLACEMENT PROBLEM

In this section, we study the dynamic digital twin replica placement problem, where the set U_t of users and their locations at time slot t are revealed at the beginning of each time slot t . We first provide an ILP solution for the offline version of the dynamic digital twin replica placement problem, serving as a benchmark to evaluate the proposed online algorithms. We then consider a special case of the problem, by assuming $U_t = U$, $\forall t \in T$, i.e., each user $u \in U$ issues a query at each time slot t . We develop an online algorithm for the special problem with a provable competitive ratio. We finally develop an efficient online algorithm for the dynamic digital twin replica placement problem.

A. ILP Formulation

We now provide an ILP formulation for the offline version of the dynamic digital twin replica placement problem. Recall that the binary decision variable $x_{t,u,v}$ indicates whether the query of user u is processed in cloudlet/remote cloud $v \in V \cup \{v_0\}$. By Eq. (4), we can adopt $x_{u,v,t-1}$, instead of the binary constant indicator $I_{u,v,t}$, to show whether user $u \in U_t$ issued a query at time $t-1$ with a digital twin replica (or the primary digital twin) in node $v \in V \cup \{v_0\}$ for processing the query.

For notation simplicity, we define the *static delay* $d_s(u, v, t)$ of assigning the query of user $u \in U_t$ to $v \in V \cup \{v_0\}$ at time slot t as follows.

$$d_s(u, v, t) = \begin{cases} d_{com}(u, v, t) + s(u, t) \cdot \rho_{u,v} + d_{u,v,t}^{cold}, & t = 1 \\ d_{com}(u, v, t) + s(u, t) \cdot \rho_{u,v}, & t \geq 2, \end{cases} \quad (11)$$

where $d_{com}(u, v, t)$ is the communication delay of transmitting the raw data of user u to node $v \in V \cup \{v_0\}$ by Eq. (1), and $s(u, t) \cdot \rho_{u,v}$ is the processing time of the query of user u in node v at time slot t . $d_{u,v,t}^{cold}$ is the cold start delay of instantiating the container of the digital twin replica of user u in cloudlet $v \in V$ at time slot t , whilst $d_{u,v_0,t}^{cold}$ is set at 0 for the remote cloud v_0 .

Note that no digital twin replica is placed before time slot 1, and each digital twin replica deployed in a cloudlet at time slot 1 inevitably experiences a cold start. Therefore, the static delay of a query is considered to include the cold start delay at time slot 1. Meanwhile, we need to determine whether to reuse existing containers of digital twin replicas to avoid cold starts at each time slot $t \geq 2$. Therefore, we identify the static delay of a query without considering the cold start delay at time slot $t \geq 2$.

We can see when the query is processed by the primary digital twin of user u in remote cloud v_0 at time slot t , its query service

delay $D(u, v_0, t)$ is its static delay, i.e., $d_s(u, v, t) = D(u, v_0, t)$ with $d_{u,v_0,t}^{cold} = 0$, because the cold start delay is not included in the processing of a primary digital twin in the remote cloud.

Based on ILP (7) for the digital twin replica placement problem at time slot t , the Integer Nonlinear Program (INP) is provided for the offline version of the dynamic digital twin replica placement problem as follows.

$$\begin{aligned} \text{Maximize } & \sum_{t \in T} \sum_{u \in U_t} d_s(u, v_0, t) \\ & - \sum_{t \in T} \sum_{u \in U_t} \sum_{v \in V \cup \{v_0\}} d_s(u, v, t) \cdot x_{u,v,t} \\ & - \sum_{t \in [2, |T|]} \sum_{u \in U_t} \sum_{v \in V} d_{u,v,t}^{cold} \cdot (1 - x_{u,v,t-1}) \cdot x_{u,v,t}, \end{aligned} \quad (12)$$

subject to:

Eq. (1), Eq. (11),

$$\sum_{u \in U_t} m_u \cdot x_{u,v,t} \leq M_v, \quad \forall t \in T, \forall v \in V \quad (13)$$

$$\sum_{v \in V \cup \{v_0\}} x_{u,v,t} = 1, \quad \forall t \in T, \forall u \in U_t \quad (14)$$

$$x_{u,v,t} \in \{0, 1\}, \quad \forall t \in T, \forall u \in U_t, \forall v \in V \cup \{v_0\}. \quad (15)$$

Referring to the definition of the static delay $d_s(u, v, t)$ by Eq. (11), function $\sum_{t \in T} \sum_{u \in U_t} d_s(u, v_0, t)$ is the accumulative query service delay when all queries are processed by their primary digital twins in the remote cloud v_0 . Function $\sum_{t \in T} \sum_{u \in U_t} \sum_{v \in V \cup \{v_0\}} d_s(u, v, t) \cdot x_{u,v,t}$ is the accumulative static delay of all queries during the time horizon, consisting of the total service delay of queries at time slot 1, and the accumulative service delay of queries without considering the cold start delay since time slot 2. Function $\sum_{t \in [2, |T|]} \sum_{u \in U_t} \sum_{v \in V} d_{u,v,t}^{cold} \cdot (1 - x_{u,v,t-1}) \cdot x_{u,v,t}$ is the accumulative cold start delay from time slot 2 to the end of the time horizon T . Therefore, objective (12) is the accumulative utility gain of queries during the time horizon by the definition function (6) of the utility gain $H_{u,v,t}$ at each time slot $t \in T$.

We observe the objective function (12) is nonlinear, and the computation complexity of the INP (12) is high. Therefore, we perform the problem linearization, by introducing a binary variable $y_{u,v,t}$, $\forall t \in [2, |T|], \forall u \in U_t, \forall v \in V$, where $y_{u,v,t} = 1$ indicates the query service of user u processed in cloudlet v involves the cold start at time slot t , and $y_{u,v,t}$ is 0 otherwise.

Through adopting $y_{u,v,t}$, the ILP is formulated for the offline version of the dynamic digital twin replica placement problem as follows.

$$\begin{aligned} \text{Maximize } & \sum_{t \in T} \sum_{u \in U_t} d_s(u, v_0, t) \\ & - \sum_{t \in T} \sum_{u \in U_t} \sum_{v \in V \cup \{v_0\}} d_s(u, v, t) \cdot x_{u,v,t} \\ & - \sum_{t \in [2, |T|]} \sum_{u \in U_t} \sum_{v \in V} d_{u,v,t}^{cold} \cdot y_{u,v,t}, \end{aligned} \quad (16)$$

subject to:

Eq. (1), Eq. (11),

$$\sum_{u \in U_t} m_u \cdot x_{u,v,t} \leq M_v, \quad \forall t \in T, \forall v \in V \quad (17)$$

$$\sum_{v \in V \cup \{v_0\}} x_{u,v,t} = 1, \quad \forall t \in T, \forall u \in U_t \quad (18)$$

$$y_{u,v,t} \geq x_{u,v,t} - x_{u,v,t-1}, \quad \forall t \in [2, |T|], \forall u \in U_t, \forall v \in V \quad (19)$$

$$x_{u,v,t} \in \{0, 1\}, \quad \forall t \in T, \forall u \in U_t, \forall v \in V \cup \{v_0\} \quad (20)$$

$$y_{u,v,t} \in \{0, 1\}, \quad \forall t \in [2, |T|], \forall u \in U_t, \forall v \in V. \quad (21)$$

Through imposing the logic constraint (19) in such a maximization problem, $y_{u,v,t}$ is functionally equivalent to $(1 - x_{u,v,t-1}) \cdot x_{u,v,t}$, $\forall t \in [2, |T|], \forall u \in U_t, \forall v \in V$, which will be later shown in Lemma 1. Therefore, the solution by ILP (16) is optimal for the offline dynamic digital twin replica placement problem.

B. Online Algorithm for a Special Case of the Dynamic Digital Twin Replica Placement Problem

Consider a special dynamic digital twin replica placement problem by assuming $U_t = U$, $\forall t \in T$, i.e., each user $u \in U$ issues a query time slot t . For such a special problem, we develop an online algorithm with a provable competitive ratio.

An intuitive online solution is to invoke Algorithm 1 for the digital twin replica placement at each time slot t in order to maximize the accumulative utility gain greedily. However, this will inevitably cause frequent replacements of digital twin replicas of users, and the accumulative cold start delay of containers experienced by users can be prohibitively large during the time horizon, downgrading user service satisfaction on digital twin services dramatically.

Inspired by the work [40], we adopt an efficient policy to determine whether to replace digital twin replicas at each time slot t through reusing existing containers, thereby mitigating the accumulative cold start delay. Recall that $d_s(t, u, v)$ is the static delay of assigning the query of user u to node $v \in V \cup \{v_0\}$ at time slot t defined in Eq. (11). Let $\mathcal{D}_s(t)$ be the *total static delay* of queries of users at each time slot $t \in T$. With regard to Eq. (11), $\mathcal{D}_s(1)$ at time slot 1 is the total service delay of queries at time slot 1, because deploying a digital twin replica in a cloudlet at time slot 1 inevitably experiences a cold start for instantiating a container, while $\mathcal{D}_s(t)$ with $t \geq 2$ is the total service delay of queries without considering the cold start delay at each time slot $t \geq 2$.

Similarly, we define the *total dynamic delay* $\mathcal{D}_d(t)$ of users in U at time slot $t \in T$, which incurs by the cold starts due to the replacement of digital twin replicas. Especially, $\mathcal{D}_d(t)$ is the accumulative cold start delay experienced by users in U at time slot t with $t \geq 2$. Because the digital twin replica replacement is not considered at time slot 1, we set $\mathcal{D}_d(1) = 0$ at time slot 1.

Referring to the definition function Eq. (6) of the utility gain, we observe the total utility gain at each time slot $t \in T$ is $\sum_{u \in U} d_s(u, v_0, t) - \mathcal{D}_d(t) - \mathcal{D}_s(t)$.

Now we develop an online algorithm for the special problem, by determining the digital twin replica replacement at each time slot. Initially, we obtain set \mathcal{A}_1 of digital twin replicas for serving users in U via invoking Algorithm 1 at time slot 1, and directly place digital twin replicas in \mathcal{A}_1 in cloudlets by instantiating containers.

For each time slot $t \geq 2$, we first obtain set \mathcal{A}_t of digital twin replicas for placement by invoking Algorithm 1. However, we cannot directly adopt \mathcal{A}_t for digital twin replica replacement. In the following, we design an efficient policy to determine whether to replace digital twin replicas at each time slot $t \geq 2$.

Suppose \hat{t} is the last time slot when replacing digital twin replicas, with $\hat{t} = 1$ initially. Let $\alpha > 1$ be a tuning parameter. If the total dynamic delay of queries at current time slot t is no larger than $\frac{1}{\alpha}$ times the accumulative utility gain of queries from time slot \hat{t} to $(t-1)$ when ignoring the dynamic delay, i.e., $\mathcal{D}_d(t) \leq \frac{1}{\alpha} \cdot \sum_{t'=\hat{t}}^{t-1} (\sum_{u \in U} d_s(u, v_0, t') - \mathcal{D}_s(t'))$, we adopt \mathcal{A}_t for replacing digital twin replicas in cloudlets at time slot t ; otherwise, the digital twin replicas are not replaced, i.e., the placement of digital twin replicas follows that at the previous time slot $t-1$. The tuning parameter α helps bound the accumulative cold start delay, and a smaller α implies more tolerance on cold start delays, i.e., digital twin replicas are replaced more frequently; otherwise, a larger α indicates a less cold start delay is tolerated, i.e., less digital twin replica replacement is allowed.

The proposed online algorithm for the special dynamic digital twin replica placement problem is detailed in Algorithm 2.

C. Online Algorithm

We finally develop an efficient online algorithm for the dynamic digital twin replica placement problem.

Let U_{t-1} and U_t be the sets of users issuing queries at two sequential time slots $t-1$ and t , with $t \geq 2$. Denote by $\Delta\mathcal{U}_t \subseteq U_{t-1} \cap U_t$ the set of users with $t \geq 2$, and each user $u \in \Delta\mathcal{U}_t$ issues queries at both time slots $t-1$ and t , with a container deployed in a cloudlet at time $t-1$ for instantiating a digital twin replica for user u . It can be seen that we need to determine whether to reuse existing containers for instantiating digital twin replicas for users in $\Delta\mathcal{U}_t$ at time slot $t \geq 2$. Meanwhile, we need to consider instantiating new containers for the placement of digital twin replicas for serving users in $U_t \setminus \Delta\mathcal{U}_t$ at time slot $t \geq 2$.

An online algorithm is proposed for the dynamic digital twin replica placement problem, through deploying digital twin replicas for users in $\Delta\mathcal{U}_t$ and $U_t \setminus \Delta\mathcal{U}_t$ respectively, with each time slot as an iteration.

At the initial time slot 1, we obtain set \mathcal{A}_1 of digital twin replicas for serving users in U_1 by invoking Algorithm 1, and place digital twin replicas in \mathcal{A}_1 in cloudlets directly. For each time slot $t \geq 2$, we first invoke Algorithm 1 to obtain a solution $\mathcal{A}_t(\Delta\mathcal{U}_t)$, by considering deploying digital twin replicas in cloudlets of V for only users in $\Delta\mathcal{U}_t$.

Algorithm 2 Online algorithm for the special case of the dynamic digital twin replica placement problem

Input: An MEC network $G = (V \cup \{v_0\}, E)$, time horizon T , a set U of users issuing queries at each time slot $t \in T$ with each query uploading the size $s(u, t)$ of raw data for analyzing, without any future information.

Output: Maximize the accumulative utility of all users by dynamically deploying digital twin replicas in cloudlets during time horizon T .

- 1: Obtain set \mathcal{A}_1 of digital twin replicas by invoking Algorithm 1 and place digital twin replicas in \mathcal{A}_1 to cloudlets at time slot 1;
- 2: $t \leftarrow 2$;
- 3: $\hat{t} \leftarrow 1$;
- 4: **while** $t \leq |T|$ **do**
- 5: Obtain the set \mathcal{A}_t of digital twin replicas at time slot t by Algorithm 1;
- 6: Calculate the incurred delay $\mathcal{D}_d(t)$ by \mathcal{A}_t at time slot t ;
- 7: **if** $\mathcal{D}_d(t) \leq \frac{1}{\alpha} \cdot \sum_{t'=1}^{t-1} (\sum_{u \in U} d_s(u, v_0, t') - \mathcal{D}_s(t'))$ **then**
- 8: Adopt set \mathcal{A}_t for replacing digital twin replicas;
- 9: $\hat{t} \leftarrow t$;
- 10: **else**
- 11: Keep the placement of digital replicas in cloudlets at the previous time slot;
- 12: **end if**
- 13: $t \leftarrow t + 1$;
- 14: **end while**

Following the similar spirit as Algorithm 2, we adopt the following policy to determine whether to replace digital twin replicas for users in $\Delta\mathcal{U}_t$.

We calculate the incurred total static delay $\mathcal{D}_s(\Delta\mathcal{U}_t, t)$ and the total dynamic delay $\mathcal{D}_d(\Delta\mathcal{U}_t, t)$, i.e., the total cold start delay due to the replacement of digital twin replicas for users in $\Delta\mathcal{U}_t$. At time slot t , if the total dynamic delay $\mathcal{D}_d(\Delta\mathcal{U}_t, t)$ of queries is no larger than $\frac{1}{\beta}$ times the total utility gain of queries when ignoring the dynamic delay, i.e., $\mathcal{D}_d(\Delta\mathcal{U}_t, t) \leq \frac{1}{\beta} \cdot (\sum_{u \in \Delta\mathcal{U}_t} d_s(u, v_0, t) - \mathcal{D}_s(\Delta\mathcal{U}_t, t))$, we adopt $\mathbb{A}_t(\Delta\mathcal{U}_t)$ for replacing digital twin replicas in cloudlets for users in $\Delta\mathcal{U}_t$ at time slot t ; otherwise, the digital twin replicas are not replaced, i.e., the placement of digital twin replicas for users in $\Delta\mathcal{U}_t$ follows that at time slot $t - 1$.

Having deployed digital twin replicas for users in $\Delta\mathcal{U}_t$, we update the residual memory capacity of cloudlets of V , and invoke Algorithm 1 to obtain a solution $\mathbb{A}_t(U_t \setminus \Delta\mathcal{U}_t)$, by considering deploying digital twin replicas in cloudlets of V for users in $U_t \setminus \Delta\mathcal{U}_t$ only.

The proposed online algorithm for the dynamic digital twin replica placement problem is detailed in Algorithm 3.

D. Algorithm Analysis

Lemma 1: The $y_{u,v,t}$ is functionally equivalent with $(1 - x_{u,v,t-1}) \cdot x_{u,v,t}$, $\forall t \in [2, |T|]$, $\forall u \in U_t$, $\forall v \in V$, by imposing constraint (19).

Algorithm 3 Online algorithm for the dynamic digital twin replica placement problem

Input: An MEC network $G = (V \cup \{v_0\}, E)$, time horizon T , a set U_t of users issuing queries at each time slot $t \in T$ with each query uploading the size $s(u, t)$ of raw data for analyzing, without any future information.

Output: Maximize the accumulative utility of all users by dynamically deploying digital twin replicas in cloudlets during time horizon T .

- 1: Obtain the set \mathcal{A}_1 of digital twin replicas by invoking Algorithm 1 and place digital twin replicas in \mathcal{A}_1 in cloudlets at time slot 1;
- 2: $t \leftarrow 2$;
- 3: **while** $t \leq |T|$ **do**
- 4: Identify the set of users $\Delta\mathcal{U}_t \subseteq U_{t-1} \cap U_t$, where each user $u \in \Delta\mathcal{U}_t$ issues queries in both time slots $t - 1$ and t , with a container deployed in a cloudlet $v \in V$ at time $t - 1$;
- 5: Obtain the set $\mathbb{A}_t(\Delta\mathcal{U}_t)$ of digital twin replicas deployed in cloudlets V for only users in $\Delta\mathcal{U}_t$ by invoking Algorithm 1;
- 6: Identify the incurred dynamic delay $\mathcal{D}_d(\Delta\mathcal{U}_t, t)$ by $\mathbb{A}_t(\Delta\mathcal{U}_t)$ at time slot t ;
- 7: **if** $\mathcal{D}_d(\Delta\mathcal{U}_t, t) \leq \frac{1}{\beta} \cdot (\sum_{u \in \Delta\mathcal{U}_t} d_s(u, v_0, t) - \mathcal{D}_s(\Delta\mathcal{U}_t, t))$ **then**
- 8: Adopt $\mathbb{A}_t(\Delta\mathcal{U}_t)$ for replacing the digital twin replicas for users in $\Delta\mathcal{U}_t$;
- 9: **else**
- 10: Keep the placement of digital replicas for users in $\Delta\mathcal{U}_t$ at the previous time slot $t - 1$;
- 11: **end if**;
- 12: Update the residual memory resource on each cloudlet in V ;
- 13: Invoke Algorithm 1 to identify the set $\mathbb{A}_t(U_t \setminus \Delta\mathcal{U}_t)$ of digital twin replicas in cloudlets V with updated capacities for users in $U_t \setminus \Delta\mathcal{U}_t$;
- 14: $t \leftarrow t + 1$;
- 15: **end while**;

Proof: We prove the lemma by enumeration.

Especially, we have $y_{u,v,t} = 1$ by constraints (19) and (21), when $x_{u,v,t-1} = 0$ and $x_{u,v,t} = 1$, i.e., a container is deployed in cloudlet v for instantiating a digital twin replica of user u at time slot t , but the container is not deployed in cloudlet v for the user at time slot $t - 1$. Similarly, we have $y_{t,u,v} \geq 0$ by constraints (19) and (21), for the value pair of $(x_{u,v,t-1}, x_{u,v,t})$ with values of $(0, 0)$, $(1, 0)$ and $(1, 1)$. We observe the objective function (16) is a maximization one, and the term $-\sum_{t \in [2, |T|]} \sum_{u \in U_t} \sum_{v \in V} d_{u,v,t}^{cold} \cdot y_{u,v,t}$ is non-positive. Therefore, to maximize the objective, it will implicitly set $y_{u,v,t} = 0$.

Hence, $y_{u,v,t}$ is functionally equivalent with $(1 - x_{u,v,t-1}) \cdot x_{u,v,t}$, $\forall t \in [2, |T|]$, $\forall u \in U_t$, $\forall v \in V$, by imposing constraint (19). ■

Lemma 2: By Algorithm 2 for the special case of the dynamic digital twin replica placement problem, the accumulative dynamic delay of queries of users in U is no larger than $\frac{1}{\alpha}$ times the accumulative utility gain of users in U over time horizon T when ignoring the dynamic delay, i.e., $\sum_{t=1}^{|T|} \mathcal{D}_d(t) \leq \frac{1}{\alpha} \cdot \sum_{t=1}^{|T|} (\sum_{u \in U} d_s(u, v_0, t) - \mathcal{D}_s(t))$.

Proof: Let \hat{t}_j be the time slot of the j th replacement of digital twin replicas, with $\hat{t}_0 = 1$ and $\mathcal{D}_d(\hat{t}_0) = 0$. Let $\hat{t}_J \leq |T|$ be the time slot when the last digital twin replacement occurs during T . By the proposed policy for determining the digital twin replica replacement, we have

$$\begin{aligned} \sum_{t=1}^{|T|} \mathcal{D}_d(t) &= \sum_j \mathcal{D}_d(\hat{t}_j) \\ &\leq \frac{1}{\alpha} \cdot \sum_{t=1}^{\hat{t}_J-1} \left(\sum_{u \in U} d_s(u, v_0, t) - \mathcal{D}_s(t) \right) \\ &\leq \frac{1}{\alpha} \cdot \sum_{t=1}^{|T|} \left(\sum_{u \in U} d_s(u, v_0, t) - \mathcal{D}_s(t) \right), \end{aligned} \quad (22)$$

because $\sum_{u \in U} d_s(u, v_0, t) - \mathcal{D}_s(t) \geq 0, \forall t \in T$. ■

Lemma 3: The optimal solution to the special case of the dynamic digital twin replica placement problem is no larger than σ times the accumulative static delay in solution by Algorithm 2, i.e., $H^* \leq \sigma(\sum_{t=1}^{|T|} \sum_{u \in U} d_s(u, v_0, t) - \sum_{t=1}^{|T|} \mathcal{D}_s(t))$, where H^* is the optimal utility gain of the problem and $\sigma = \max_{t \in T} \left\{ \frac{\max\{\sum_{u \in U} d_s(u, v_0, t) - \mathcal{D}_s(t)\}}{\min\{\sum_{u \in U} d_s(u, v_0, t) - \mathcal{D}_s(t)\}} \right\}$, i.e., the maximum ratio of the largest to the least utility gain when ignoring the dynamic delay at any time slot [40].

Proof: Because of the definition of σ , we can obtain $\sum_{u \in U} d_s(u, v_0, t) - \mathcal{D}_s^*(t) \leq \sigma \cdot (\sum_{u \in U} d_s(u, v_0, t) - \mathcal{D}_s(t))$, where $\mathcal{D}_s^*(t)$ is the static delay in the optimal solution of the problem at time slot t . Then,

$$\begin{aligned} \sum_{t=1}^{|T|} \sum_{u \in U} d_s(u, v_0, t) - \sum_{t=1}^{|T|} \mathcal{D}_s^*(t) \\ \leq \sigma \left(\sum_{t=1}^{|T|} \sum_{u \in U} d_s(u, v_0, t) - \sum_{t=1}^{|T|} \mathcal{D}_s(t) \right). \end{aligned} \quad (23)$$

Let $\mathcal{D}_d^*(t)$ be the dynamic delay in the optimal solution at time slot t . Because H^* is the optimal utility gain of the problem, then

$$\begin{aligned} H^* &= \sum_{t=1}^{|T|} \sum_{u \in U} d_s(u, v_0, t) - \sum_{t=1}^{|T|} \mathcal{D}_s^*(t) - \sum_{t=1}^{|T|} \mathcal{D}_d^*(t) \\ &\leq \sum_{t=1}^{|T|} \sum_{u \in U} d_s(u, v_0, t) - \sum_{t=1}^{|T|} \mathcal{D}_s^*(t) \\ &\leq \sigma \left(\sum_{t=1}^{|T|} \sum_{u \in U} d_s(u, v_0, t) - \sum_{t=1}^{|T|} \mathcal{D}_s(t) \right). \end{aligned} \quad (24)$$

In eq. (24) holds by in eq. (23). ■

Theorem 3: Given an MEC network $G = (V \cup \{v_0\}, E)$, time horizon T , a set U of users issuing queries at each time

slot $t \in T$, there is an online algorithm, Algorithm 2, with a competitive ratio of $\frac{1}{\sigma}(1 - \frac{1}{\alpha})$ for the special case of the dynamic digital twin replica placement problem, which takes $O(|V|^3 + |T| \cdot |V| \cdot |U| \cdot \log \frac{1}{\epsilon} + \frac{|T| \cdot |V|}{\epsilon^4})$ time over time horizon T , where $\alpha > 1$ is a control parameter and $\sigma > 1$ is defined in Lemma 3.

Proof: Suppose H^* and H are the accumulative utility gains obtained by the optimal solution and the solution delivered by Algorithm 2 for the special case of the dynamic digital twin replica placement problem, respectively. Then,

$$\begin{aligned} U &= \sum_{t=1}^{|T|} \sum_{u \in U} d_s(u, v_0, t) - \sum_{t=1}^{|T|} \mathcal{D}_s(t) - \sum_{t=1}^{|T|} \mathcal{D}_d(t) \\ &\geq \sum_{t=1}^{|T|} \sum_{u \in U} d_s(u, v_0, t) - \sum_{t=1}^{|T|} \mathcal{D}_s(t) \\ &\quad - \frac{1}{\alpha} \cdot \sum_{t=1}^{|T|} \left(\sum_{u \in U} d_s(u, v_0, t) - \mathcal{D}_s(t) \right), \text{ by Lemma 2} \\ &\geq \left(1 - \frac{1}{\alpha}\right) \left(\sum_{t=1}^{|T|} \sum_{u \in U} d_s(u, v_0, t) - \sum_{t=1}^{|T|} \mathcal{D}_s(t) \right) \\ &\geq \frac{1}{\sigma} \left(1 - \frac{1}{\alpha}\right) H^*, \text{ by Lemma 3.} \end{aligned} \quad (25)$$

The time complexity of Algorithm 2 is dominated by invoking Algorithm 1 at each time slot. Referring to Theorem 2, Algorithm 2 takes the time $O(|V|^3 + |T| \cdot |V| \cdot |U| \cdot \log \frac{1}{\epsilon} + \frac{|T| \cdot |V|}{\epsilon^4})$. ■

Theorem 4: Given an MEC network $G = (V \cup \{v_0\}, E)$, time horizon T , a set U of users in the MEC network, a set $U_t \subseteq U$ of users issuing queries at each time slot $t \in T$, there is an online algorithm, Algorithm 3, delivers a feasible online solution for the dynamic digital twin replica placement problem, which takes $O(|V|^3 + |T| \cdot |V| \cdot |U| \cdot \log \frac{1}{\epsilon} + \frac{|T| \cdot |V|}{\epsilon^4})$ time over time horizon T , where ϵ is a constant with $0 < \epsilon \leq 1$.

Proof: It can be seen that no resource violation is caused by Algorithm 3. Therefore, a feasible online solution is delivered by Algorithm 3 for the dynamic digital twin replica placement problem.

The time complexity of Algorithm 3 is dominated by invoking Algorithm 1 for sets of users $\Delta \mathcal{U}_t$ and $U_t \setminus \Delta \mathcal{U}_t$ respectively at each time slot. With $\Delta \mathcal{U}_t \subseteq U$ and $U_t \setminus \Delta \mathcal{U}_t \subseteq U$, we can observe Algorithm 3 takes the time $O(|V|^3 + |T| \cdot |V| \cdot |U| \cdot \log \frac{1}{\epsilon} + \frac{|T| \cdot |V|}{\epsilon^4})$ by Theorem 2. ■

VI. PERFORMANCE EVALUATION

In this section, we evaluated the algorithm performance for the digital twin replica placement problem per time slot and the dynamic digital twin replica placement problem over a finite time horizon.

A. Experimental Settings

Consider an MEC network, and the number of APs ranges from 50 to 250, with each AP co-located with a cloudlet. The

topology of each network is generated by the GT-ITM tool [10]. The memory capacity on each cloudlet is drawn within [5, 8] GB randomly [39]. To instantiate a digital twin replica, the memory allocation of a container is a value in the range from 256 MB to 3,008 MB [7]. The finite time horizon has 100 time slots, and there are 2,000 mobile users within the MEC network, with 1,000 queries issued at each time slot. The location of each user at the first time slot is randomly chosen, and the user then moves to another random location at the next time slot iteratively. The size of the data uploaded by a user query is set within [100, 500] MB [37]. The cold start delay range of a container is from 100 ms to 500 ms [7]. The data processing rate of queries is set within [0.5, 2] MB per ms, while the data processing rate in the remote cloud is set as 20 MB per ms [19]. The communication delay for transmitting a unit of data (one MB) along a link between each pair of APs is set within [0.2, 1] ms [38]. The communication delay for transmitting a unit of data (one MB) between the remote cloud and an AP through the gateway is set within [2, 10] ms [34]. Parameter ϵ is set as 0.5. Parameters α and β are set at 4 and 2, respectively. The running times of different algorithms for the special dynamic digital twin replica placement problem and the dynamic digital twin replica placement problem are the accumulative running time over the entire time horizon, and the value in each figure is the mean result of 30 different MEC instances with the same network size. The running time of each algorithm is obtained by a desktop with an Octa-Core Intel(R) Xeon(R) CPU @ 2.30GHz, 8G RAM. Unless otherwise specified, we adopt the above-mentioned parameters by default.

We consider the following two baselines to evaluate the proposed algorithm Algorithm 1, referred to as Alg. 1, for the digital twin replica placement problem.

GDY: Considering user queries one by one, we greedily instantiate a digital twin replica in a cloudlet with sufficient residual memory resource or utilize the primary digital twin in remote cloud in order to maximize the collected utility gain of each user query.

LP: The relaxed linear program solution (7) of the ILP for the digital twin replica placement problem, through relaxing binary decision variable $x_{u,v,t}$ into real numbers between 0 and 1. The solution will serve as an upper bound of the optimal solution for this maximization problem.

Round: A variant of the rounding algorithm in [32]. Algorithm Round first performs random rounding on the relaxed linear program solution (7) to obtain an integral solution, at the expense of memory capacity violations. It then modifies the solution to make it become a feasible solution of the problem. That is, for each cloudlet with capacity violation, the algorithm picks a digital twin replica from it randomly and removes the digital twin replica. This operation continues until no capacity violation on the cloudlet occurs.

We evaluated Algorithm 2, referred to as Alg. 2, for the special dynamic digital twin replica placement problem, against benchmark algorithms: GDY_d, LP_d and Round_d. Especially, algorithms GDY_d and Round_d invoke algorithm GDY and Round for placing digital twin replicas at each

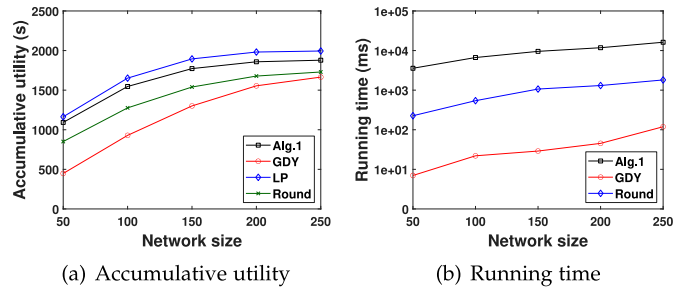


Fig. 1. Algorithm performance for the digital twin replica placement problem.

time slot, respectively. Algorithm LP_d is the relaxed linear program solution (16) for the offline dynamic digital twin replica placement problem, serving as an upper bound on its optimal solution.

We also evaluated Algorithm 3, referred to as Alg. 3, for the dynamic digital twin replica placement problem, against benchmark algorithms: GDY_d, LP_d and Round_d.

B. Algorithm Performance for the Digital Twin Replica Placement Problem

We first investigated the performance of Alg. 1 against benchmarks GDY, LP and Round for the digital twin replica placement problem by varying network size from 50 to 250. Fig. 1 demonstrates the results, i.e., the accumulative utility and running time of different algorithms. By Fig. 1(a), we observe the accumulative utility of Alg. 1 is 93.9% of that by LP with the network size of 50, while Alg. 1 outperforms GDY and Round by 12.8% and 8.6%, respectively, when the network size reaches 250. This is because Alg. 1 efficiently deploys digital twin replicas in cloudlets to enhance user satisfaction on digital twin data-based query services at a time slot. Fig. 1(b) illustrates Alg. 1 takes more running time than GDY, due to invoking the approximation algorithm from [6].

We then studied the impact of the number of queries issued at a time slot, i.e., $|U_t|$, on the performance of Alg. 1, and Fig. 2 depicts the performance curves of Alg. 1 when $|U_t| = 500, 1,000, 1,500$ and $2,000$, respectively. We can see the performance of Alg. 1 when $|U_t| = 500$ is 28.1% of that by itself when $|U_t| = 2,000$, with the network size of 250, according to Fig. 2(a). The rationale behind this is more utility gain can be obtained with more user queries. Also, more queries will lead to a longer running time of the algorithm, evidenced by Fig. 2(b).

We finally evaluated the impact of the parameter ϵ on the performance of Alg. 1 when $\epsilon = 0.1, 0.5$ and 1 , respectively. Fig. 3 plots the performance curves of Alg. 1 with different values of ϵ . It can be seen from Fig. 3(a) that the performance of Alg. 1 when $\epsilon = 1$ is 95.5% of that by itself when $\epsilon = 0.1$ if the network size is fixed at 250. Also, Fig. 3(b) indicates Alg. 1 with $\epsilon = 0.1$ takes the longest running time, as a smaller value of ϵ leads to a more accurate approximate solution by Algorithm 1.

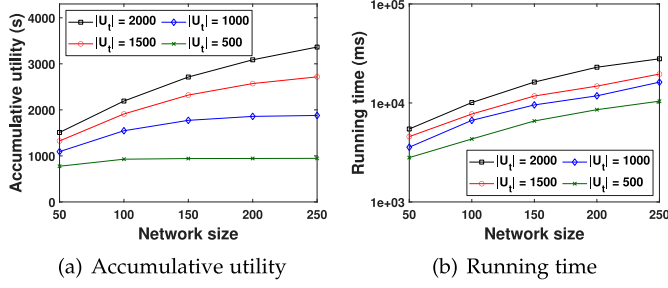


Fig. 2. Impact of the number of queries $|U_t|$ issued at a time slot on the performance of Alg. 1.

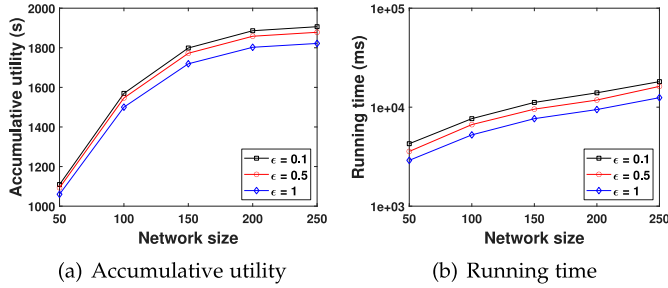


Fig. 3. Impact of the parameter ϵ on the performance of Alg. 1.

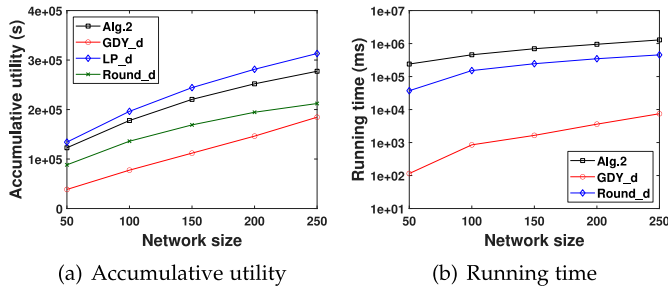


Fig. 4. Algorithm performance for the special dynamic digital twin replica placement problem.

C. Algorithm Performance for the Special Dynamic Digital Twin Replica Placement Problem

We compared the performance of Alg. 2 against benchmarks GDY_d, LP_d and Round_d for the special dynamic digital twin replica placement problem with different network sizes, and the performance of Alg. 2, GDY_d, LP_d and Round_d has been shown in Fig. 4. As observed from Fig. 4(a), when the network size is 250, the accumulative utility by Alg. 2 is 85.5% of that by LP_d, and Alg. 2 outperforms GDY_d and Round_d by 50.3% and 30.7%, respectively. The justification is that Alg. 2 adopts an efficient policy to determine whether to reuse existing containers for placing digital twin replicas at each time slot. Similar to the phenomena in Fig. 2(b), Fig. 4(b) shows Alg. 2 takes more running time than GDY_d and Round_d, because Alg. 2 invokes Alg. 1 at each time slot.

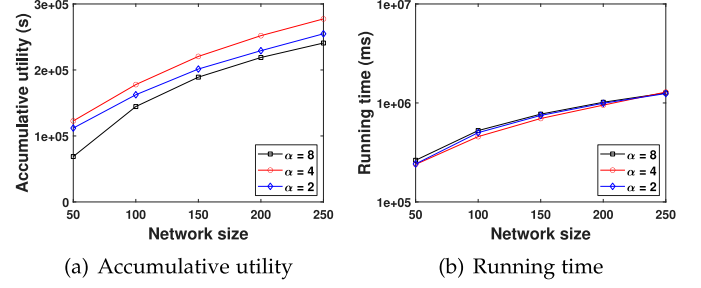


Fig. 5. Impact of parameter α on the performance of Alg. 2.

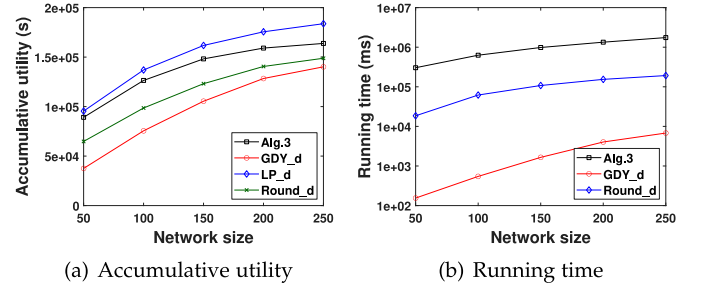


Fig. 6. Algorithm performance for the dynamic digital twin replica placement problem.

We also evaluated the impact of parameter α on the performance of Alg. 2. Fig. 5 plots the results of Alg. 2 when $\alpha = 2, 4$ and 8 , respectively. When the network size is 250, we observe from Fig. 5(a) the accumulative utility of Alg. 2 with $\alpha = 4$ is 15.2% higher than that by itself with $\alpha = 8$. With regard to the designed policy to determine whether to replace digital twin replicas at each time slot by Alg. 2, a larger α means that a less cold start delay is tolerated, which allows less digital twin replica replacement. Fig. 5(b) shows parameter α has little impact on the running time of Alg. 2.

D. Algorithm Performance for the Dynamic Digital Twin Replica Placement Problem

As shown in Fig. 6, a group of experiments was conducted to investigate the performance of Alg. 3 for the dynamic digital twin replica placement problem against benchmarks GDY_d, LP_d and Round_d. Fig. 6(a) indicates that Alg. 3 exhibits a promising performance, which outperforms GDY_d and Round_d by 16.9% and 10.1%, respectively, with the network size of 250. Moreover, the accumulative utility of Alg. 3 is 89.2% of that by LP_d with the network size of 250. This is because Alg. 3 instantiates digital twin replicas in cloudlets carefully to mitigate the cold start delay at each time slot, considering whether a user issues a query in the previous time slot or not.

We finally evaluated the impact of the number of time slots, i.e., $|T|$, on the performance of Alg. 3 by setting $|T| = 50, 100, 150$ and 200 , as demonstrated in Fig. 7. Fig. 7(a) reveals the accumulative utility of Alg. 3 with $|T| = 50$ is 25.2% of that by itself with $|T| = 200$, when the network size is 250. This is

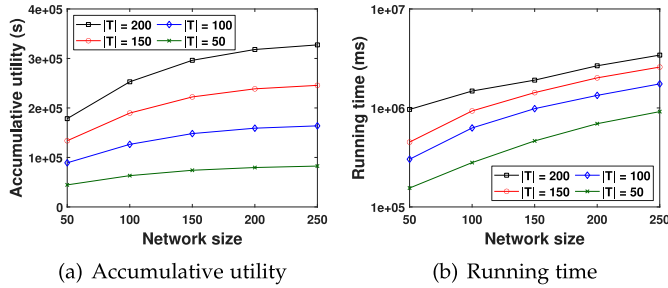


Fig. 7. Impact of the number of time slots on the performance of Alg. 3.

because Alg. 3 collects more utility with the prolonging of the time horizon, which also leads to more running time, observed from Fig. 7(b).

VII. CONCLUSION AND FUTURE WORK

In this paper, we investigated mobility-aware service provisioning in digital twin-enabled serverless edge computing. We first formulated two novel problems for service provisioning: the digital twin replica placement problem per time slot, and the dynamic digital twin replica placement problem over a finite time horizon, respectively. For the digital twin replica placement problem, we then provided an ILP solution and a performance-guaranteed approximation algorithm. We developed an online algorithm for the dynamic digital twin replica placement problem, and a performance-guaranteed online algorithm considering a special case of the problem. Finally, we evaluated the performance of the proposed algorithms for digital twin replica placements by simulations. The results showed the proposed algorithms are promising.

In our future work, we will investigate the fairness issue on mobility-aware service provisioning in digital twin-enabled serverless edge computing, and put it as one of the metrics in the design of efficient algorithms for the services. One potential topic is to consider fairness among different user queries, through assigning different weights on user queries, and developing algorithms to improve service fairness.

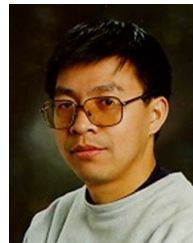
REFERENCES

- [1] "AWS Lambda." Amazon. Accessed: Apr. 2023. [Online]. Available: <https://aws.amazon.com/lambda/>
- [2] O. Ascigil, A. G. Tasiopoulos, T. K. Phan, V. Sourlas, I. Psaras, and G. Pavlou, "Resource provisioning and allocation in function-as-a-service edge-clouds," *IEEE Trans. Services Comput.*, vol. 15, no. 4, pp. 2410–2424, Jul./Aug. 2022.
- [3] "Azure functions." Microsoft. Accessed: Apr., 2023. [Online]. Available: <https://azure.microsoft.com/en-us/services/functions/>
- [4] S. Chen, Y. Wang, D. Yu, J. Ren, C. Xu, and Y. Zheng, "Privacy-enhanced decentralized federated learning at dynamic edge," *IEEE Trans. Comput.*, vol. 72, no. 8, pp. 2165–2180, Aug. 2023.
- [5] "Cloud functions." Google. Accessed: Apr. 2023. [Online]. Available: <https://cloud.google.com/functions/>
- [6] R. Cohen, L. Katzir, and D. Raz, "An efficient approximation for the generalized assignment problem," *Inf. Process. Lett.*, vol. 100, pp. 162–166, Nov. 2006.
- [7] A. Das, S. Imai, M. P. Wittie, and S. Patterson, "Performance optimization for edge-cloud serverless platforms via dynamic task placement," in *Proc. IEEE/ACM Int. Symp. Cluster Cloud Internet Comput. (CCGRID'20)*, 2020, pp. 41–50.
- [8] T. Elgamal, A. Sandur, K. Nahrstedt, and G. Agha, "Costless: Optimizing cost of serverless computing through function fusion and placement," in *Proc. IEEE/ACM Int. Symp. Cluster Cloud Int. Comput. (CCGRID)*, Piscataway, NJ, USA: IEEE Press, 2018, pp. 300–312.
- [9] 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.
- [10] "Modeling topology of large Internetworks," GT-ITM. [Online]. Available: <http://www.cc.gatech.edu/projects/gtitm/>
- [11] Y. Han et al., "A dynamic hierarchical framework for IoT-assisted digital twin synchronization in the metaverse," *IEEE Internet Things J.*, vol. 10, no. 1, pp. 268–284, Jan. 2023.
- [12] G. Jing, Y. Zou, D. Yu, C. Luo, and X. Cheng, "Efficient fault-tolerant consensus for collaborative services in edge computing," *IEEE Trans. Comput.*, vol. 72, no. 8, pp. 2139–2150, Aug. 2023.
- [13] H. Ko and S. Pack, "Function-aware resource management framework for serverless edge computing," *IEEE Internet Things J.*, vol. 10, no. 2, pp. 1310–1319, Jan. 2023.
- [14] J. Li, S. Guo, W. Liang, Q. Chen, Z. Xu, and W. Xu, "SFC-enabled reliable service provisioning in mobile edge computing via digital twins," in *Proc. IEEE 19th Int. Conf. Mobile Ad Hoc Smart Syst. (MASS'22)*, Piscataway, NJ, USA: IEEE Press, 2022, pp. 311–317.
- [15] J. Li et al., "Digital twin-assisted, SFC-enabled service provisioning in mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 23, no. 1, pp. 393–408, Jan. 2024.
- [16] J. Li et al., "AoI-aware user service satisfaction enhancement in digital twin-empowered edge computing," *IEEE/ACM Trans. Netw.*, to be published, 2023, doi: 10.1109/TNET.2023.3324704.
- [17] J. Li et al., "Digital twin-enabled service provisioning in edge computing via continual learning," *IEEE Trans. Mobile Comput.*, to be published, 2023, doi: 10.1109/TMC.2023.3332668.
- [18] J. Li et al., "Wait for fresh data? Digital twin empowered IoT services in edge computing," in *Proc. IEEE 20th Int. Conf. Mobile Ad Hoc Smart Syst. (MASS'23)*, Piscataway, NJ, USA: IEEE Press, 2023, pp. 397–405.
- [19] 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.
- [20] J. Li, W. Liang, W. Xu, Z. Xu, Y. Li, and X. Jia, "Service home identification of multiple-source IoT applications in edge computing," *IEEE Trans. Services Comput.*, vol. 16, no. 2, pp. 1417–1430, Mar./Apr. 2023.
- [21] J. Li, J. Wang, Q. Chen, Y. Li, and A. Y. Zomaya, "Digital twin-enabled service satisfaction enhancement in edge computing," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM'23)*, Piscataway, NJ, USA: IEEE Press, 2023, pp. 1–10.
- [22] J. Li et al., "Blockchain assisted decentralized federated learning (BLADE-FL): Performance analysis and resource allocation," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 10, pp. 2401–2415, Oct. 2022.
- [23] 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.
- [24] M. Liu, S. Fang, H. Dong, and C. Xu, "Review of digital twin about concepts, technologies, and industrial applications," *J. Manuf. Syst.*, vol. 58, pp. 346–361, Jan. 2021.
- [25] T. Liu, L. Tang, W. Wang, Q. Chen, and X. Zeng, "Digital-twin-assisted task offloading based on edge collaboration in the digital twin edge network," *IEEE Internet Things J.*, vol. 9, no. 2, pp. 1427–1444, Jan. 2022.
- [26] 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.
- [27] L. Pan, L. Wang, S. Chen, and F. Liu, "Retention-aware container caching for serverless edge computing," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM'22)*, Piscataway, NJ, USA: IEEE Press, 2022, pp. 1069–1078.
- [28] I. Pelle, F. Paolucci, B. Sonkoly, and F. Cugini, "Latency-sensitive edge/cloud serverless dynamic deployment over telemetry-based packet-optical network," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 9, pp. 2849–2863, Sep. 2021.

- [29] A. Rasheed, O. San, and T. Kvamsdal, "Digital twin: Values, challenges and enablers from a modeling perspective," *IEEE Access*, vol. 8, pp. 21980–22012, 2020.
- [30] 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.
- [31] Q. Tang et al., "Distributed task scheduling in serverless edge computing networks for the internet of things: A learning approach," *IEEE Internet Things J.*, vol. 9, no. 20, pp. 19634–19648, Oct. 2022.
- [32] 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.
- [33] D. Wang, B. Li, B. Song, Y. Liu, K. Muhammad, and X. Zhou, "Dual-driven resource management for sustainable computing in the blockchain-supported digital twin IoT," *IEEE Internet Things J.*, vol. 10, no. 8, pp. 6549–6560, Apr. 2023.
- [34] 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.
- [35] K. Wei et al., "User-level privacy-preserving federated learning: Analysis and performance optimization," *IEEE Trans. Mobile Comput.*, vol. 21, no. 9, pp. 3388–3401, Sep. 2022.
- [36] Y. Wu, K. Zhang, and Y. Zhang, "Digital twin networks: A survey," *IEEE Internet Things J.*, vol. 8, no. 18, pp. 13789–13804, Sep. 2021.
- [37] Z. Xu, Y. Fu, Q. Xia, and H. Li, "Enabling age-aware big data analytics in serverless edge clouds," in *Proc. IEEE Conf. Comp. Commun. (INFOCOM'23)*, Piscataway, NJ, USA: IEEE Press, 2023, pp. 1–10.
- [38] Z. Xu et al., "Energy-aware collaborative service caching in a 5G-enabled MEC with uncertain payoffs," *IEEE Trans. Commun.*, vol. 70, no. 2, pp. 1058–1071, Feb. 2022.
- [39] Z. Xu et al., "Stateful serverless application placement in MEC with function and state dependencies," *IEEE Trans. Comput.*, vol. 72, no. 9, pp. 2701–2716, Sep. 2023.
- [40] L. Zhang, C. Wu, Z. Li, C. Guo, M. Chen, and F. Lau, "Moving big data to the cloud: An online cost-minimizing approach," *IEEE J. Sel. Areas Commun.*, vol. 31, no. 12, pp. 2710–2721, Dec. 2013.
- [41] Z. Zhang, H. Zhou, L. Zhao, and V. C. M. Leung, "Digital twin assisted computation offloading and service caching in mobile edge computing," in *Proc. IEEE 42nd Int. Conf. Distrib. Comput. Syst. (ICDCS'22)*, Piscataway, NJ, USA: IEEE Press, 2022.
- [42] J. Zheng et al., "Digital twin empowered heterogeneous network selection in vehicular networks with knowledge transfer," *IEEE Trans. Veh. Technol.*, vol. 71, no. 11, pp. 12154–12168, Nov. 2022.



Song Guo (Fellow, IEEE) is a Full Professor with the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology. He also holds a Changjiang Chair Professorship awarded by the Ministry of Education of China. His research interests include big data, edge AI, mobile computing, and distributed systems. With many impactful papers published in top venues in these areas, he has been recognized as a Highly Cited Researcher (Web of Science) and received over 12 Best Paper Awards from IEEE/ACM conferences, journals and technical committees. He is the Editor-in-Chief of IEEE Open Journal of the Computer Society. He has served on IEEE Communications Society Board of Governors, IEEE Computer Society Fellow Evaluation Committee, and editorial board of a number of prestigious international journals such as IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, IEEE TRANSACTIONS ON CLOUD COMPUTING, and IEEE INTERNET OF THINGS JOURNAL. He has also served as a Chair of organizing and technical committees of many international conferences. He is an ACM Distinguished Member.



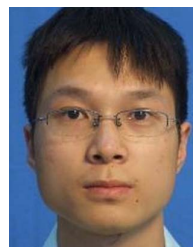
Weifa Liang (Senior Member, IEEE) received the B.Sc. degree from Wuhan University, China, in 1984, the M.E. degree from the University of Science and Technology of China, in 1989, and the Ph.D. degree from the Australian National University, in 1998, all in computer science. He is a Full Professor with the Department of Computer Science, City University of Hong Kong. Prior to that, he was a Full Professor with the Australian National University. His research interests include design and analysis of energy efficient routing protocols for wireless ad hoc and sensor networks, Mobile Edge Computing (MEC), Network Function Virtualization (NFV), Internet of Things and digital twins, design and analysis of parallel and distributed algorithms, approximation and online algorithms, combinatorial optimization and graph theory. He currently serves as an editor for IEEE TRANSACTIONS ON COMMUNICATIONS.



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



Jing Li received the B.Sc. with the first class Honours and the Ph.D. degrees from The Australian National University, in 2018 and 2022, respectively. He is currently a Postdoctoral Fellow with City University of Hong Kong. His research interests include edge computing, internet of things, digital twin, network function virtualization, and combinatorial optimization.



Quan Chen (Member, IEEE) received the B.S., Master's, and Ph.D. degrees from the School of Computer Science and Technology, Harbin Institute of Technology, China. He is currently an Associate Professor with the School of Computers, Guangdong University of Technology. He once worked as a Postdoctoral Research Fellow with the Department of Computer Science, Georgia State University. His research interests include routing and scheduling algorithms in wireless networks and sensor networks.

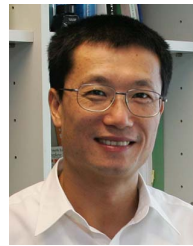


Wenchao Xu (Member, IEEE) received the B.E. and M.E. degrees from Zhejiang University, Hangzhou, China, in 2008 and 2011, respectively, and the Ph.D. degree from the University of Waterloo, Canada, in 2018. He is a Research Assistant Professor with The Hong Kong Polytechnic University. In 2011, he joined Alcatel Lucent Shanghai Bell Co. Ltd., where he was a Software Engineer for telecom virtualization. He has also been an Assistant Professor with the School of Computing and Information Sciences, Caritas Institute of Higher Education, Hong Kong. His research interests include wireless communication, internet of things, distributed computing, and AI enabled networking.



Kang Wei (Member, IEEE) received the B.S. degree in information engineering from Xidian University, Xian, China, in 2014, and the Ph.D. degree from Nanjing University of Science and Technology, supervised by Prof. Jun Li. He is currently a Postdoctoral Fellow with The Hong Kong Polytechnic University. He has won the 2022 IEEE Signal Processing Society Best Paper Award and 2022 Wiley China Open Science Author of the Year. He has served as reviewer for several prestigious journals and conferences such as IEEE

JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, IEEE TRANSACTIONS ON MOBILE COMPUTING, and ACM KDD. He mainly focuses on privacy protection and optimization techniques for edge intelligence, including federated learning, differential privacy, and network resource allocation.



Xiaohua Jia (Fellow, IEEE) received the B.Sc. and M.E. degrees from the University of Science and Technology of China, in 1984 and 1987, respectively, and the D.Sc. 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, from 2006 to 2009, *Journal of World Wide Web*, *Wireless Networks*, *Journal of Combinatorial Optimization*, and so on. He is the General Chair of ACM MobiHoc 2008, the TPC Co-Chair of IEEE MASS 2009, the Area-Chair of IEEE INFOCOM 2010, the TPC Co-Chair of IEEE GlobeCom 2010, the Panel Co-Chair of IEEE INFOCOM 2011, and the General Co-Chair of IEEE ICDCS 2023.