

Adversarial Bandit Learning Assisted Online Optimization for Digital Twin Placement and Update in End-Edge-Cloud Collaboration

Yuye Yang¹, Changyan Yi¹, Senior Member, IEEE, Haifeng Zhu, Shimin Gong², Member, IEEE, Jiawen Kang³, Senior Member, IEEE, and Dusit Niyato⁴, Fellow, IEEE

Abstract—Digital twin (DT) is envisioned not only to perform the high-fidelity virtual representation of its corresponding physical entity (PE), but also to serve as an active agent delivering diverse types of sophisticated services. This paper studies an end-edge-cloud collaborative DT placement and update framework. Specifically, we consider that DTs are dynamically placed across edge servers (ESs) via migration following their paired PEs' potential mobility, while being supported by real-time data fetched from the cloud center and user ends. On top of this, we emphasize a unique feature that DTs should also be continually updated capturing the uncertain evolutions for both personalized service ability improvement and versatile service ability maintenance, where the personalization is improved by utilizing the experiential knowledge from the cloud center and their corresponding PEs, and the versatility is maintained by integrating pre-stored profiles. To maximize the long-term system-wide average weighted quality-of-service (QoS) in handling all types of PEs' service requests under the stringent system cost constraint, we formulate an online problem to jointly optimize DT migrations, service priorities towards various request types, and all related DT updating strategies. To address underlying difficulties, we propose a novel adversarial bandit learning assisted online optimization approach, called ARBOK. We first leverage the Lyapunov decomposition method to transform the long-term problem into multiple instant ones, each of which is further decoupled into two correlated subproblems. For solving one subproblem with a bilinear structure, we develop a McCormick envelopes based algorithm (MO-EL). Besides, we design an extended adversarial combinatorial multi-armed bandit algorithm (AC-BL) to tackle the other subproblem, which constructs a super arm set to resolve

the issue of excessively large decision space and employs a robust scheme to handle the inherent uncertainty and non-stationarity in each super arm's loss function. We integrate both algorithms seamlessly into ARBOK and alternately execute them till the convergence. Theoretical analysis and extensive simulations show the effectiveness of the introduced dynamic DT placement and continual update framework, demonstrating that ARBOK can converge to the asymptotic optimum within a polynomial-time complexity while outperforming counterparts.

Index Terms—Digital twin, end-edge-cloud collaboration, dynamic placement, continual update, adversarial bandit learning, online optimization.

I. INTRODUCTION

DIGITAL twin (DT), as an instance, refers to the high-fidelity, dedicated virtual model of a physical entity (PE), such as a vehicle in intelligent transportation, a patient in smart healthcare, which establishes and sustains precise one-to-one mapping and synchronization between physical and digital spaces [1]. Beyond merely serving as a passive data mirror, DT is also expected to become an active agent providing the delegated task-execution or autonomous acting services for its corresponding PE [2], [3]. These services commonly require sophisticated reasoning and decision-making capabilities for high quality-of-service (QoS) outcomes, and thus DT should be naturally empowered by advanced artificial intelligence (AI) techniques, e.g., deep neural networks and large language models, fulfilling PE's heterogeneous requests [4], [5].

The end-edge-cloud collaboration offers a promising paradigm for the cost-efficient and timely DT deployment, wherein DTs can be dynamically placed at different edge servers (ESs) via migration in accordance with PEs' potential mobility and access handover, while being supported by real-time collected data from both user ends and the cloud center [6], [7]. It is worth noting that, different from conventional service applications that are commonly pre-configured and remain relatively unchanged [8], [9], [10], each DT, serving as the virtual counterpart, must be continually updated whenever the status of its corresponding PE evolves due to either internal or external influences. Failing to do so will reduce the DT's effectiveness, making it useless or even misleading. For example, if a patient's DT is outdated, any diagnosis relying on it may lead to incorrect prescriptions or dangerous delays in critical care [11], [12].

Received 28 September 2025; revised 13 February 2026; accepted 18 March 2026. Date of publication 24 March 2026; date of current version 6 August 2026. This work was supported by National Natural Science Foundation of China under Grant 62372488. Recommended for acceptance by H. Tabassum. (Corresponding author: Changyan Yi.)

Yuye Yang and Changyan Yi are with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China (e-mail: mryyy@nuaa.edu.cn; changyan.yi@nuaa.edu.cn).

Haifeng Zhu is with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China, and also with the Servo Technology Institute of China Aerospace Science and Industry Corporation, Nanjing 210094, China (e-mail: caccgzhf@nuaa.edu.cn).

Shimin Gong is with the School of Intelligent Systems Engineering, Shenzhen Campus of Sun Yat-sen University, Shenzhen 518000, China, and also with Southern Marine Science and Engineering Guangdong Laboratory, Zhuhai 519082, China (e-mail: gongshm5@mail.sysu.edu.cn).

Jiawen Kang is with the School of Automation, Guangdong University of Technology, Guangzhou 510006, China (e-mail: kavinkang@gdut.edu.cn).

Dusit Niyato is with the College of Computing and Data Science, Nanyang Technological University, Singapore 639798 (e-mail: dniyato@ntu.edu.sg).

This article has supplementary downloadable material available at <https://doi.org/10.1109/TMC.2026.3677210>, provided by the authors.

Digital Object Identifier 10.1109/TMC.2026.3677210

Although several pioneering works have studied similar issues, all of them predominantly conceptualize DTs as specialized service providers with monolithic functions, focusing solely on personalized adaptations [13], [14], [15]. However, in practice, an associated DT may serve its PE with a stochastic flow of multiple types of sophisticated service requests, each of which has distinct performance criteria [1], [16], requiring continual updates not only for the personalized service ability improvement but also for the versatile service ability maintenance. Particularly, in healthcare, DT should simultaneously provide various medical services, such as remote physiological monitoring, predictive risk assessment and customized diagnostic testing, while adjusting their priorities on the fly according to its patient's conditions [17].

To achieve the above goals, it is necessary to determine the target ESs for dynamic DT placement, along with the strategies for the follow-up DT update. Specifically, for improving each DT's personalization, the experiential knowledge of dealing with a chosen type of service requests should be utilized as the input to adaptively tune its model. Such experiential knowledge includes both historical training data from the cloud center (as the offline experience), and the up-to-date status data from its corresponding PE (as the newly injected contextual information). Meanwhile, for maintaining each DT's versatility, the profile in tackling every type of service requests should be built, recorded and later selected to integrate into the DT updating process. These profiles are supposed to be cached/preserved on designated ESs, each of which encapsulates the current ability of addressing one type of service requests. Clearly, all aforementioned procedures have to be jointly optimized, yet it is significantly challenging due to the following reasons.

- 1) The introduced DT placement and update are tightly coupled, which cannot be optimized sequentially. In particular, the DT placement affects the resource availability for the DT update, and conversely, the long-term overhead of future DT updates should be taken into account for the DT placement. Moreover, since corresponding PEs may experience uncertain access handover due to the random mobility [9], and suffer from evolving status induced by unpredictable service preferences [6], all included decisions must be made dynamically over time. In general, these time-varying decisions concern the placement of DTs across ESs via migration, and their updates through knowledge selection and profile management assisted by the end-edge-cloud collaboration to uphold both personalized and versatile service capabilities. Obviously, an online optimization approach is imperative for supporting such a dynamic DT placement and continual update framework.
- 2) The DT update further involves two interrelated aspects, i.e., personalized service ability improvement and versatile service ability maintenance. Naturally, owing to model limitations, improving a DT's personalization inevitably degrades its versatility, while overemphasizing a DT's versatility in turn restricts its personalization [18]. Therefore, we have to flexibly balance them by carefully regulating the service priorities of DTs in tackling different types of

service requests, driving DTs to be continually updated on-demand. However, these service priorities depend not only on the uncertainly-changing subjective preferences of PEs but also on the decision-related objective QoS performances produced by their DTs, lacking a closed-form relationship [19]. Even by knowing the above two factors, these service priorities can hardly be obtained because they are in fact unobservable environment parameter, which cannot be revealed as explicit observations by traditional learning algorithms [20], [21]. This calls for an advanced online learning algorithm that can operate under the implicit decision feedback while being compatible with the online optimization solution.

In this paper, we investigate the dynamic DT placement and continual update framework over an end-edge-cloud collaborative networking system. We aim to maximize the long-term system-wide average weighted QoS in handling all types of PEs' service requests, while constraining the resulted system cost. We formulate a complicated online joint optimization problem to determine the placement/migration of DTs on ESs, the selection of experiential knowledge for updating the personalized service ability, the caching/preservation of DT profiles, and the integration of profiles for updating the versatile service ability, while simultaneously learning the service priority of each request type on each DT, in every defined time slot. To this end, we develop a novel adversarial bandit learning assisted online optimization approach, called ARBOK. We first decompose the formulated long-term problem into a series of instant ones by applying the Lyapunov decomposition method. Then, we decouple each instant problem into two correlated subproblems based on the observation that DT placement/migration and profile caching/preservation are prerequisites for other decisions. For the subproblem regarding DT placement and profile caching, which exhibits a bilinear structure, we design a McCormick envelopes based algorithm (MO-EL). For the subproblem with respect to service priority inference, experiential knowledge selection and profile integration for the DT update, we devise an adversarial combinatorial bandit learning algorithm (AC-BL), that simultaneously infers the latent control variable, i.e., the service priority of each type of requests, and refines the estimated loss by adjusting all the rest of variables, thereby obtaining the near-optimal outcome. By solving one subproblem with the given solution of the other, MO-EL and AC-BL are integrated into ARBOK and executed alternately till convergence.

The key contributions of this paper are outlined below.

- To the best of our knowledge, we are the first to study the DT deployment in end-edge-cloud collaboration by dynamic DT placement and continual update. We formulate an online optimization problem that jointly determines DT placement via migration and DT update with both personalized service ability improvement and versatile service ability maintenance, with the goal of maximizing the long-term system-wide average weighted QoS in tackling all types of PEs' service requests.
- We propose ARBOK as the solution approach, which first conducts the Lyapunov decomposition, and then alternately leverages MO-EL and AC-BL algorithms till the

TABLE I
COMPARISON BETWEEN OUR WORK AND SOME EXISTING REPRESENTATIVE WORKS

References	DT deployment framework			References	Online optimization approaches
	DT placement	DT update	DT update for both personalization and versatility		
Zhang et al. [24]	√	×	×	Chen et al. [28]	A Follow-the-Perturbed-Leader MAB algorithm that employed a conditional geometric resampling scheme using the explicit decision feedback .
Mou et al. [25]	√	×	×	Shi et al. [9]	A two-timescale Lyapunov-based online optimization algorithm , which first transformed the long-term problem into multiple instant ones, and then applied a Lagrangian-based method to optimize each decomposed convex problem .
Jiang et al. [13]	√	√	×	Chen et al. [27]	A contextual MAB algorithm , utilizing the upper confidence bound method to dynamically adjust strategies based on the explicit decision feedback .
Li et al. [15]	√	√	×	Our paper	A novel approach, called ARBOK, which incorporates an extended adversarial MAB algorithm into a Lyapunov-based online optimization solution , enabling the online optimization under implicit decision feedback .
Chen et al. [26]	√	√	×		
Our paper	√	√	√		

convergence. Aiding by the adversarial bandit learning, this approach not only addresses the formulated problem in an online manner but also well handles the implicit decision feedback stemming from unobservable environment parameters.

- We theoretically analyze the proposed ARBOK approach by rigorously showing its convergence, optimality gap and computational complexity. Furthermore, extensive simulations illustrate the effectiveness of the introduced dynamic DT placement and continual update framework, and demonstrate the superiority of the optimization solution, i.e., ARBOK, compared to counterparts.

The rest of this paper is organized as follows. Section II reviews the related work and highlights the novelties of this paper. Section III describes the considered system model, and formulates the corresponding optimization problem. In Section IV, the ARBOK approach is proposed and analyzed. Simulation results are presented in Section V, followed by the conclusion in Section VI.

II. RELATED WORK

As a key enabler of the ongoing digital transformation, the advanced AI empowered DT has shown great potential in a diverse spectrum of critical fields, such as agriculture, healthcare, and manufacturing. For example, Ghandar et al. in [22] constructed a DT of an aquaponics system upon the decision tree based machine learning algorithm to predict the growth rate of fishes and plants. Nakao et al. in [23] built a DT of the patient's thorax integrating the autoencoding generative adversarial network to achieve efficient and accurate disease diagnosis. As reported in [16], NVIDIA developed a DT platform for the production line of the BMW group, providing a smart toolkit to realize complex workshop services, such as autonomous defect detection and robotic arm trajectory planning. However, all these works considered a centralized DT deployment framework solely relying on the cloud center.

Recently, some researchers have shifted their attentions to the end-edge-cloud collaborative DT deployment. For instance, Zhang et al. in [24] proposed an actor-critic based deep reinforcement learning algorithm to dynamically migrate DTs

with the assistance of edge and cloud computing, aiming to minimize the total service cost. Mou et al. in [25] formulated an optimization problem for vehicular DTs' migration across ESs, which was then solved by an off-policy driven reinforcement learning algorithm. Nevertheless, both of them focused on the placement of DTs only, whereas ignoring that DTs should also be timely updated to synchronize with the status variations of their corresponding PEs. To this end, some preliminary works have explored the DT update on top of the DT placement. For example, Jiang et al. in [13] leveraged an iterative double auction-based federated learning approach for updating DTs over a distributed network given the known DT placement. In [26], Chen et al. designed a multi-agent reinforcement learning algorithm to jointly optimize DT placement together with the DT update, targeting to minimize the PE-DT synchronization delay. However, most of existing works treated the DT as a simple virtual mirror, overlooking its role as an active agent that can assist its PE to concurrently handle diverse types of service requests while balancing personalization and versatility on-demand.

Numerous research efforts have been devoted to studying the online optimization solutions, which may possibly be applied for the dynamic DT placement and continual update in the end-edge-cloud collaboration. For example, Chen et al. in [27] introduced a contextual bandit learning algorithm, utilizing the upper confidence bound method to dynamically adjust strategies based on the decision feedback. In [9], Shi et al. proposed a two-timescale Lyapunov-based online optimization algorithm, which first transformed the long-term problem into multiple instant ones, and then applied a Lagrangian-based method to optimize each decomposed convex problem. Moreover, a Follow-the-Perturbed-Leader algorithm was designed in [28] that employed a conditional geometric resampling scheme using the observed loss on different decisions. However, none of them can address our formulated problem, which contains an unobservable environment parameter, rendering the objective unattainable and its decision feedback implicit.

In summary, as shown in Table I, this work differs from the existing literature in the following aspects: i) we propose a tailored dynamic DT placement and continual update framework, taking into account both personalized service ability improvement

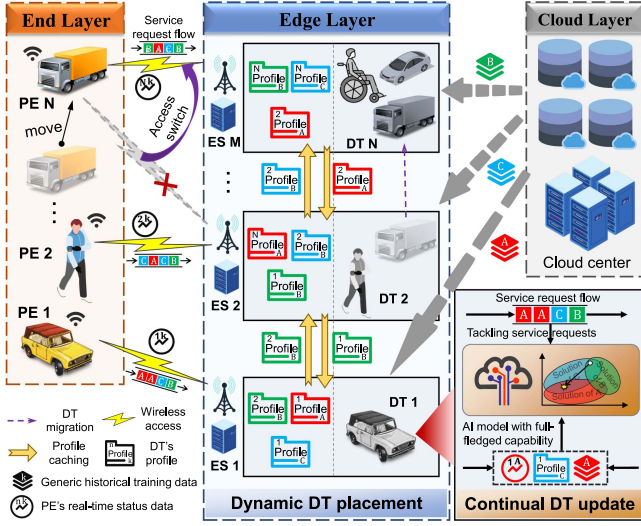


Fig. 1. The framework of dynamic DT placement and continual update in the end-edge-cloud collaboration.

and versatile service ability maintenance; and ii) we develop ARBOK, a novel adversarial bandit learning assisted online optimization approach that incorporates an extended adversarial multi-armed bandit (MAB) algorithm into a Lyapunov-based online optimization solution, enabling the stable operation under implicit decision feedback, so as to achieve the near-optimal long-term system performance.

III. SYSTEM MODEL AND PROBLEM FORMULATION

In this section, we first present a system overview on how DTs are dynamically placed and continually updated over an end-edge-cloud collaborative networking system. Then, we describe the dynamic DT placement via migration and the continual DT update by personalized service ability improvement along with the versatile service ability maintenance are described. After that, aiming to maximize the overall QoS in tackling all types of different PEs' service requests by their associated DTs, we formulate an online optimization problem.

A. System Overview

Consider the DT deployment over an end-edge-cloud collaborative networking system, as shown in Fig. 1, consisting of a set of PEs denoted as $\mathcal{N}$ with $|\mathcal{N}| = N$, multiple geographically distributed ESs in set $\mathcal{M}$ with $|\mathcal{M}| = M$, and a cloud center acting as the central controller. PEs (roaming around) generate streams of heterogeneous and sophisticated service requests¹ in different types from set $\mathcal{K}$ with $|\mathcal{K}| = K$, requiring to be processed by their associated DTs [2]. Here, each DT represents the digital avatar of a certain PE empowered by a complex AI model with full-fledged service capability, which can tackle all types of its corresponding PE's service requests [16]. To

provide agile and cost-effective responses, DTs are required to be deployed on ESs, supporting by real-time collected data from both PEs and the cloud center. Since PEs and DTs are naturally one-to-one paired, we can denote the set of DTs as the same as PEs, i.e., $\mathcal{N}$. Due to the random mobility, each PE may switch its wireless access from one ES to another [6], [29]. Triggered by this, after the initial construction, DTs should be dynamically placed on various ESs via migration. Note that one ES is allowed to simultaneously host multiple DTs of different PEs. To depict the system dynamics, we define that the timeline is segmented into $T \in \mathbb{N}^+$ fine-grained time slots, and denote $t \in \mathcal{T} = \{1, 2, \dots, T\}$ as the index of the t -th time slot.

Besides the dynamic placement via migration, each DT $n \in \mathcal{N}$ is also expected to be continually updated on a slot-by-slot basis, for meeting the ever-changing personalized service demand of its corresponding PE while balancing its service capability in dealing with different types of the randomly generated service requests. This is because, in practice, each PE may have distinct service preferences/weights towards different types of its requests, denoted by $\pi_{n,k}(t), \forall n \in \mathcal{N}, \forall k \in \mathcal{K}, \forall t \in \mathcal{T}$, which is closely related to its traits and characteristics [30]. Such preferences can even be time-varying, leading to uncertain evolutions of service priorities required for its associated DT in handling different service requests. To describe this key enabler of the continual DT update, we further introduce $\omega_{n,k}(t)$ to denote the priority of service request type $k \in \mathcal{K}$ on DT $n \in \mathcal{N}$ in time slot $t \in \mathcal{T}$. It is worth noting that $\omega_{n,k}(t)$ cannot be predetermined solely based on $\pi_{n,k}(t)$ but instead has to be dynamically revealed coupled with the continual DT updating process, depending on both the subjective preferences of PEs and the objective QoS performances produced by their DTs, denoted by $Q_{n,k}(t), \forall n \in \mathcal{N}, \forall k \in \mathcal{K}, \forall t \in \mathcal{T}$. In other words, the determination of $\omega_{n,k}(t)$ should consider not only to satisfy service preferences of the PE but also to address fluctuations (particularly starvations/saturations) of the QoS provided by its associated DT to sustain the versatile capability². Moreover, since these two factors may exert intricate and subtle mutual impacts in reality, $\omega_{n,k}(t), \forall n \in \mathcal{N}, \forall k \in \mathcal{K}, \forall t \in \mathcal{T}$ cannot be given by an explicit function in advance, but instead should be learned in an online manner [33].

Motivated by the above, as illustrated in Fig. 2, the continual update of each DT can be divided into two main aspects, i.e., personalized service ability improvement and versatile service ability maintenance [34], [35]. The personalized service ability of each DT $n \in \mathcal{N}$ can be improved by utilizing the experiential knowledge with a selected service type $k \in \mathcal{K}$ of the corresponding PE. Such experiential knowledge includes the generic historical training data from the cloud center and the personalized status data from its corresponding PE, all aggregated at ES $m \in \mathcal{M}$. As a consequence of the personalized service ability improvement, the versatile service ability of DT

¹These service requests (e.g., customized surgery planning, immersive interaction with industrial robot and virtual training of autonomous driving strategy) demand the support of high-fidelity simulation platforms, which have to be solved by DT-based solutions [12].

²In fact, this setting has been widely adopted in various applications. For instance, in queueing systems with dynamic priorities, the service priority of each customer in the buffer is adjusted according to its delay sensitivity (similar to PE's preference) and queue waiting delay (similar to the QoS performance produced by the DT) [31], [32].

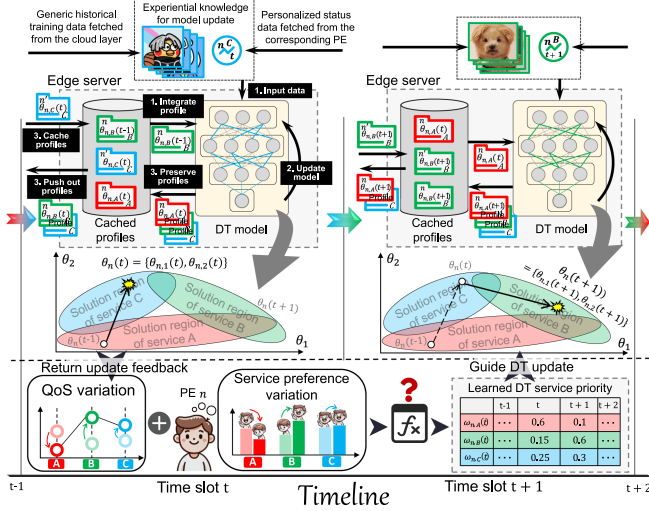


Fig. 2. An illustration of the continual DT update. Particularly, in each time slot $t \in \mathcal{T}$, we first input the newly collected experiential knowledge and integrate the previously cached profile for updating DTs. After that, we preserve the profiles of service request types and cache them on different ESs with sufficient caching resources.

$n \in \mathcal{N}$ is naturally biased over the time³, and thus has to be maintained by timely integrating the chosen profile of the DT in tackling specific type of service request in the updating process, where each profile records the current service ability for a particular type of service requests from its PE [18]. All profiles are automatically produced in accompany with the DT and cached on the designated ESs for preservation. In each time slot, DT $n \in \mathcal{N}$ can then integrate one profile selected from the cache of its deployed ES $m \in \mathcal{M}$ into the updating process for versatile service ability maintenance.

In summary, we aim to maximize the long-term overall QoS in processing all types of different PEs' service requests by their associated DTs, while reducing the system cost arising from the aforementioned dynamic DT placement and continual DT update. Particularly, in each time slot t , we aim to determine i) which DT should be placed on which ES via migration, ii) which type of experiential knowledge should be fetched for improving the personalized service ability of each DT, iii) which DT profile should be preserved and cached on which ES, and iv) which profile should be integrated for maintaining the versatile service ability of each DT. For convenience, all important notations in this paper are listed in Table II.

B. DT Placement via Migration

Since PEs are mobile, each associated DT should be timely migrated from its corresponding PE's previously accessed ES in time slot $t-1 \in \mathcal{T}$ to another ES that the PE newly switches its access in time slot $t \in \mathcal{T}$. Denote $a_{n,m}(t) \in \{0,1\}$ as the DT placement decision indicating whether DT $n \in \mathcal{N}$ is determined to be placed on ES $m \in \mathcal{M}$ in each time slot $t \in \mathcal{T}$,

³This is because each DT can only be mostly skillful in handling a few types of service requests at the same time due to model limitations, while its service ability for other types of requests (especially rare ones) will be gradually forgot after multiple rounds of updating process [18].

TABLE II
IMPORTANT NOTATIONS IN THIS PAPER

Symbol	Meaning
$a_{n,m}(t)$	placement decision of DT n
$C_n^{pl}(t)$	transmission cost of placing DT n
$C_n^{dl}(t)$, $C_n^{ul}(t)$	transmission cost of downloading generic historical training data and uploading real-time status data for updating DT n
$C_n^{ud}(t)$	computation cost of updating DT n
$C_n^{cache}(t)$	transmission cost of caching profiles of DT n
$D_n^{mdl}(t)$	model size of DT n
$D_k^{tr}(t)$	generic historical training data size of request k
$D_{n,k}^p(t)$	real-time status data size for request k of PE n
$D_{n,k}^{prf}(t)$	profile size of DT n skilled in handling request k
$h_{n,k}(t)$	profile integration decision in updating DT n
$\mathcal{N}, \mathcal{M}, \mathcal{K}$	set of DTs/PEs, ESs and service request types
$Q_{n,k}(t)$	QoS of DT n in tackling request k
$u_{n,k}(t)$	experiential knowledge fetch type for updating DT n
V	Lyapunov control parameter
$z_{n,m,k}(t)$	caching decision of DT n 's profile recording the ability of handling request k
$\omega_{n,k}(t)$	service priority of request k on DT n
$\varepsilon_{n,k}(t)$	reduced QoS of request k after updating DT n
$\sigma_{n,k}(t)$	compensate QoS of request k after updating DT n

i.e., $a_{n,m}(t) = 1$ if DT n is placed on ES m in time slot t , and $a_{n,m}(t) = 0$ otherwise. Note that DT $n \in \mathcal{N}$ can only be placed on one ES simultaneously, i.e., $\sum_{m \in \mathcal{M}} a_{n,m}(t) = 1$, because placing multiple redundant copies of each DT is always resource-exhausted [9]. For each DT $n \in \mathcal{N}$, we define $D_n^{mdl}(t)$ as its current model size in time slot $t \in \mathcal{T}$. Then, the transmission cost of placing DT $n \in \mathcal{N}$ by migrating its model from ES m' accessed in time slot $t-1$ to ES m accessed in time slot t can be expressed as

$$C_n^{pl}(t) = \sum_{m \in \mathcal{M}} \sum_{m' \in \mathcal{M}} a_{n,m'}(t-1) a_{n,m}(t) \frac{D_n^{mdl}(t)}{r_{m',m}} p_{m'}, \quad (1)$$

where $r_{m',m}$ stands for the transmission rate from ES m' to ES m via a fiber link [36], and $p_{m'}$ is the transmission cost per unit time of ES m' (e.g., power consumption [8]).

C. DT Update by Personalized Ability Improvement

The personalized service ability improvement of each DT $n \in \mathcal{N}$ can be done by updating its model in each time slot $t \in \mathcal{T}$ with the aggregated experiential knowledge of a certain service request type, including the generic historical training data downloaded from the cloud center and the real-time status data fetched from the corresponding PE. Let $u_{n,k}(t) \in \{0,1\}$ be a decision indicating whether the experiential knowledge of service request type $k \in \mathcal{K}$ is chosen to be collected for updating the personalization of DT $n \in \mathcal{N}$ in time slot $t \in \mathcal{T}$.

By defining $D_k^{tr}(t)$ as the size of generic historical training data of service request type $k \in \mathcal{K}$ at the cloud center, the transmission cost of downloading such training data can be obtained as

$$C_n^{dl}(t) = \sum_{m \in \mathcal{M}} \sum_{k \in \mathcal{K}} a_{n,m}(t) \frac{u_{n,k}(t) D_k^{tr}(t)}{r_{c,m}} p_c, \quad (2)$$

where $r_{c,m}$ and p_c stands for the downlink transmission rate and the transmission cost per unit time via a fiber link from the cloud center to ES $m \in \mathcal{M}$, respectively.

Meanwhile, let $D_{n,k}^p(t)$ be the real-time status data size related to service request in type $k \in \mathcal{K}$ of PE $n \in \mathcal{N}$ in time slot $t \in \mathcal{T}$. Then, the transmission cost of fetching this status data for updating DT $n \in \mathcal{N}$ from its corresponding PE can be written as

$$C_n^{ul}(t) = \sum_{m \in \mathcal{M}} \sum_{k \in \mathcal{K}} a_{n,m}(t) \frac{u_{n,k}(t) D_{n,k}^p(t)}{r_{n,m}(t)} p_n, \quad (3)$$

where p_n is the transmission cost per unit time of PE n , and $r_{n,m}(t)$ represents the wireless transmission rate from PE n to ES m . To measure $r_{n,m}(t)$, we define the location of PE $n \in \mathcal{N}$ as $(x_n(t), y_n(t))$, and let (x_m, y_m) be the fixed location of each ES $m \in \mathcal{M}$. The distance between any PE $n \in \mathcal{N}$ and ES $m \in \mathcal{M}$ can then be calculated as $S_{n,m}(t) = \sqrt{(x_n(t) - x_m)^2 + (y_n(t) - y_m)^2}$, and based on the Shannon-Hartley formula [37], the transmission rate from PE $n \in \mathcal{N}$ to its accessed ES $m \in \mathcal{M}$ is expressed as

$$r_{n,m}(t) = b_{n,m}(t) \log \left(1 + \frac{(S_{n,m}(t))^{-\theta} p_n |h_{n,m}(t)|^2}{N_0 b_{n,m}(t)} \right), \quad (4)$$

where $b_{n,m}(t) \in (0, 1]$ is the communication resource allocated to PE $n \in \mathcal{N}$ in time slot $t \in \mathcal{T}$, $h_{n,m}(t)$ captures the Rayleigh fading effect and follows an exponential distribution with a unity mean, N_0 is the spectral density of the channel noise power, p_n is the transmission power of PE n , and $\theta \geq 2$ is the path loss exponent [38], [39].

Having this experiential knowledge (i.e., generic historical training data from the cloud center and real-time status data from the corresponding PE), each DT $n \in \mathcal{N}$ is updated by tuning its model in each time slot $t \in \mathcal{T}$. This results in a computation cost is calculated as [40]

$$C_n^{ud}(t) = \sum_{m \in \mathcal{M}} \sum_{k \in \mathcal{K}} a_{n,m}(t) u_{n,k}(t) e_{n,t}(D_{n,k}^{knowl}(t)) \times D_{n,k}^{knowl}(t) (F_m)^2 \rho_m c_m, \quad (5)$$

where $e_{n,t}(D_{n,k}^{knowl}(t))$ denotes the training rounds in the updating process, which is a function in terms of the total data size of experiential knowledge [35], i.e., $D_{n,k}^{knowl}(t) = D_k^{tr}(t) + D_{n,k}^p(t)$, ρ_m is the effective switched capacitance of ES m depending on its chip architecture, F_m is the CPU speed (measured by cycles/s) of ES m , and c_m is the number of CPU cycles required for ES m to process a unit of data.

If $u_{n,k}(t) = 1$, i.e., DT $n \in \mathcal{N}$ is updated by improving the personalized service ability of request type $k \in \mathcal{K}$ in time slot $t \in \mathcal{T}$, the QoS for tackling such type of service requests on DT n increases. Motivated by the saturation effect [41], [42] in existing continual learning studies, the corresponding QoS can be expressed as

$$Q_{n,k}(t) = 1 - (1 - Q_{n,k}(t-1)) \lambda^{e_{n,t}(D_{n,k}^{knowl}(t))}, \quad (6)$$

where $Q_{n,k}(t-1)$ is the basic QoS in updating DT $n \in \mathcal{N}$ with the service request of type $k \in \mathcal{K}$ in the previous time slot $t -$

$1 \in \mathcal{T}$, $\lambda = 1 - (2 - \zeta\chi)\chi\kappa$ stands for the effective contraction factor, ζ , χ and κ are hyperparameters.

Remark: The evolution of $Q_{n,k}(t)$ in (6) can be interpreted as the achieved QoS from updating a DT through the personalized service ability improvement based on the QoS $Q_{n,k}(t-1)$ in the previous time slot, capturing the diminishing marginal gains in QoS improvement as the previously achieved QoS increases. Therefore, $Q_{n,k}(t)$ is a dimensionless metric that is normalized to the interval $Q_{n,k}(t) \in [0, 1]$, and it is monotonically nondecreasing in t , bounded above by 1. The detailed derivation of its specific mathematical formulation is provided in Appendix A, available online.

D. DT Update by Versatile Ability Maintenance

As an inevitable consequence of the personalized service ability improvement, DTs' service abilities may be gradually biased along with the updating process, i.e., when the QoS of tackling a specific service request type $k \in \mathcal{K}$ is improved, the QoS of tackling the other types $k' \in \mathcal{K} \setminus \{k\}$ can be significantly reduced due to the change of some key model parameters [18]. However, PEs' service requests are generated randomly in-stream and all required to be well addressed. Therefore, to maintain the versatile service ability of each associated DT $n \in \mathcal{N}$, it is necessary to preserve and integrate profiles in handling various types of service requests into its updating process in each time slot $t \in \mathcal{T}$, so as to compensate/balance the QoS by a certain extent. Let $h_{n,k'}(t) \in [0, 1]$ be the profile integration decision indicating whether DT $n \in \mathcal{N}$ selects to integrate the profile of service request of type $k' \in \mathcal{K} \setminus \{k\}$ in the updating process in time slot $t \in \mathcal{T}$. Then, the QoS for tackling the service request of type k' on DT n becomes

$$Q_{n,k'}(t) = Q_{n,k'}(t-1) - \varepsilon_{n,k'}(t) + \sum_{m \in \mathcal{M}} a_{n,m}(t) z_{n,m,k}(t-1) h_{n,k'}(t) \sigma_{n,k'}(t), \quad (7)$$

where $\varepsilon_{n,k'}(t)$ stands for the QoS reduction in the time slot t , and $\sigma_{n,k'}(t)$ represents the restored QoS if DT n chooses to integrate the profile of service requests type k' in the updating process (i.e., $h_{n,k'}(t) = 1$). Note that each DT $n \in \mathcal{N}$ can only integrate the profiles that were preserved and cached in the previous time slot $t-1 \in \mathcal{T}$ on ES $m \in \mathcal{M}$, of which the associated DT n newly migrates to in time slot $t \in \mathcal{T}$. Motivated by the QoS variation behavior observed in practical DT models [43], [44], together with the weight regularization method in existing continual learning studies [45], [46], $\varepsilon_{n,k'}(t)$ and $\sigma_{n,k'}(t)$ can be measured by the quadratic form of the biases, i.e.,

$$\varepsilon_{n,k'}(t) = \sum_{j \in \mathcal{J}} (\theta_{n,j}(t) - \theta_{n,k',j}(t))^2, \quad (8)$$

$$\sigma_{n,k'}(t) = \sum_{j \in \mathcal{J}} \Omega_{n,k',j}(t) (\theta_{n,j}(t) - \theta_{n,k',j}(t))^2, \quad (9)$$

where $\theta_{n,j}(t) \in [0, 1]$ stands for the current value of DT n 's each key model parameter $j \in \mathcal{J}$, $\theta_{n,k',j}(t)$ represents the value of key model parameter j of DT $n \in \mathcal{N}$ that is skillful in handling

service request of type $k' \in \mathcal{K} \setminus \{k\}$, and $\Omega_{n,k',j}(t) \in [0, 1]$ is the weight of parameter j in performing the service request of type k' [47].

After performing the versatile service ability maintenance in the current time slot, each DT $n \in \mathcal{N}$ preserves profiles (which are automatically produced and refreshed) for future use, each of which contains the up-to-date status information of its corresponding PE and the newly-updated service ability of a service request type. These profiles can be cached on different ESs with sufficient caching resources. Let $D_{n,k}^{prf}(t)$ be the data size of DT n 's profile which records the service ability of handling service request of type $k \in \mathcal{K}$, and $z_{n,m,k}(t) \in \{0, 1\}$ denote as the profile caching decision indicating whether this profile is selected to be preserved and cached on ES $m \in \mathcal{M}$ in time slot $t \in \mathcal{T}$. Then, the transmission cost of caching/preserving profiles of DT $n \in \mathcal{N}$ in each time slot $t \in \mathcal{T}$ is written as

$$C_n^{cache}(t) = \sum_{k \in \mathcal{K}} \sum_{m \in \mathcal{M}} \sum_{m' \in \mathcal{M}} a_{n,m'}(t) z_{n,m,k}(t) \frac{D_{n,k}^{prf}(t)}{r_{m',m}} p_{m'}. \quad (10)$$

E. Problem Formulation

Considering the continual DT update by both personalized service ability improvement and versatile service ability maintenance, the achieved QoS for tackling each service request of type $k \in \mathcal{K}$ on DT $n \in \mathcal{N}$ in each time slot $t \in \mathcal{T}$ can be expressed by combining (6) and (7) as

$$Q_{n,k}(t) = \begin{cases} Q_{n,k}(t-1) - \varepsilon_{n,k}(t), & \text{if } u_{n,k}(t) = 0, h_{n,k}(t) = 0, \\ Q_{n,k}(t-1) + \eta_{n,k}(t), & \text{if } u_{n,k}(t) = 1, h_{n,k}(t) = 0, \\ Q_{n,k}(t-1) - \varepsilon_{n,k}(t) + \sum_{m \in \mathcal{M}} a_{n,m}(t) z_{n,m,k}(t-1) \\ \quad \times \sigma_{n,k}(t), & \text{if } u_{n,k}(t) = 0, h_{n,k}(t) = 1, \end{cases}$$

meaning that if the service ability of handling service request type k is neither chosen for improvement nor selected for maintenance in updating DT n , the QoS is calculated by solely subtracting $\varepsilon_{n,k}(t)$; besides, if DT n is updated by improving the personalized service ability of handling service request type k (i.e., $u_{n,k}(t) = 1$ and $h_{n,k}(t) = 0$), the QoS is calculated according to (6); otherwise, if the service ability of handling service request type k is selected to be maintained in updating DT n (i.e., $u_{n,k}(t) = 0$ and $h_{n,k}(t) = 1$), the QoS is calculated according to (7). Note that the same service request type k should not be simultaneously selected for both personalized ability improvement and versatile ability maintenance when updating DT n , i.e., $u_{n,k}(t) + h_{n,k}(t) \leq 1, \forall n \in \mathcal{N}, k \in \mathcal{K}, t \in \mathcal{T}$. To evaluate the overall performance of DT deployment in end-edge-cloud collaboration, we take the long-term system-wide average weighted QoS in tackling all types of PEs' service requests as the performance measurement, i.e.,

$$Q^{sys} = \lim_{T \rightarrow \infty} \frac{1}{TN} \sum_{t \in \mathcal{T}} \sum_{n \in \mathcal{N}} \sum_{k \in \mathcal{K}} \omega_{n,k}(t) Q_{n,k}(t). \quad (11)$$

Recall that $\omega_{n,k}(t)$ is defined as the priority of service request type $k \in \mathcal{K}$ on DT $n \in \mathcal{N}$ in time slot $t \in \mathcal{T}$, implicitly reflecting

both PE's subjective preference and DT's objective service ability.

Moreover, to be cost-effective, we impose a long-term system cost constraint as

$$\lim_{T \rightarrow \infty} \frac{1}{T} \sum_{t \in \mathcal{T}} C_n^{tol}(t) \leq C^{\max}, \quad (12)$$

where $C_n^{tol}(t) = C_n^{pl}(t) + C_n^{dl}(t) + C_n^{ul}(t) + C_n^{cache}(t) + C_n^{ud}(t)$ stands for the total cost introduced by dynamic placement and continual update of DT $n \in \mathcal{N}$ in each time slot $t \in \mathcal{T}$, and $C^{\max}$ represents the pre-determined cost budget.

Then, with the objective of maximizing the overall performance Q^{sys} while ensuring that the incurred system cost does not exceed a certain budget, we formulate a long-term optimization problem to jointly determine DT migration decision $a_{n,m}(t)$, fetched experiential knowledge type $u_{n,k}(t)$ for DTs' personalized ability improvement, and profile integration $h_{n,k}(t)$ and caching $z_{n,m,k}(t)$ for DTs' versatile ability maintenance, denoted as $\mathcal{J}^{sys} = \{a_{n,m}(t), u_{n,k}(t), h_{n,k}(t), z_{n,m,k}(t)\}_{\forall n,m,k,t}$, i.e.,

$$\mathcal{P}_1: \max_{\mathcal{J}^{sys}} Q^{sys} = \lim_{T \rightarrow \infty} \frac{1}{TN} \sum_{t \in \mathcal{T}} \sum_{n \in \mathcal{N}} \sum_{k \in \mathcal{K}} \omega_{n,k}(t) Q_{n,k}(t)$$

$$\text{s.t.} \quad \sum_{m \in \mathcal{M}} a_{n,m}(t) = 1, \forall n, t, \quad (13a)$$

$$\sum_{k \in \mathcal{K}} u_{n,k}(t) \leq 1, \forall n, t, \quad (13b)$$

$$\sum_{k \in \mathcal{K}} h_{n,k}(t) \leq 1, \forall n, t, \quad (13c)$$

$$u_{n,k}(t) + h_{n,k}(t) \leq 1, \forall n, k, t, \quad (13d)$$

$$\sum_{m \in \mathcal{M}} z_{n,m,k}(t) \leq 1, \forall n, k, t, \quad (13e)$$

$$\sum_{n \in \mathcal{N}} \left[a_{n,m}(t) D_n^{mdl}(t) + \sum_{k \in \mathcal{K}} z_{n,m,k}(t) D_{n,k}^{prf}(t) \right] \leq z_m^{\max}, \forall m, \quad (13f)$$

$$\omega_{n,k}(t) \in f_{n,k}(\pi_{n,k}(t), Q_{n,k}(t)), \forall n, k, t, \quad (13g)$$

$$\lim_{T \rightarrow \infty} \frac{1}{T} \sum_{t \in \mathcal{T}} C_n^{tol}(t) \leq C^{\max}, \forall n, \quad (13h)$$

where constraint (13a) states that every DT has to and can only be placed on one ES in each time slot; constraints (13b) and (13c) indicate that each DT can fetch the experiential knowledge of at most one type of service requests and integrate no more than one profile during the updating process in each time slot; constraint (13d) requires that the types of service requests for personalized service ability improvement and versatile service ability maintenance cannot be the same when updating each DT in each time slot; constraint (13e) enforces that every profile of each DT is preserved and cached on at most one ES in each time slot; constraint (13f) restricts that cached DTs and all their profiles cannot exceed the caching capacity of each ES; constraint (13g) signifies that the unobserved environment

parameter $\omega_{n,k}(t)$ can be seen as an output of a complex and chaotic function $f_{n,k}(\cdot)$, in terms of $\pi_{n,k}(t)$ and $Q_{n,k}(t)$, if this function exists; and constraint (13h) is the long-term system cost constraint.

Theorem 1: The formulated long-term optimization problem $\mathcal{P}_1$ is NP-hard.

Proof: We can prove the NP-hardness of problem $\mathcal{P}_1$ by showing that a relaxed case of $\mathcal{P}_1$ is NP-hard via a polynomial-time reduction from the classical multiple knapsack problem (MKP) [48]. Given any time slot $t \in \mathcal{T}$, we construct a relaxed case of $\mathcal{P}_1$ by i) fixing service priorities towards different request types $\omega_{n,k}(t)$, and decision variables of each DT's fetched experiential knowledge type $u_{n,k}(t)$, and profile integration $h_{n,k}(t)$ and caching $z_{n,m,k}(t)$ as constants; and ii) ignoring the long-term system cost constraint by relaxing the budget $C^{\max}$ sufficiently large. Under these relaxations, the resulted problem is equivalent to an MKP because i) each DT n can be regarded as an item with weight $D_n^{ddl}(t)$, while each ES m corresponds to a knapsack whose capacity is determined by constraint (13f), i.e., $z_m^{\max} - \sum_{n \in \mathcal{N}} \sum_{k \in \mathcal{K}} z_{n,m,k}(t) D_{n,k}^{prf}(t)$; and ii) according to constraint (13a), each item n must be assigned to exactly one knapsack m , and the associated assignment profit can be expressed by the $a_{n,m}(t)$ -related term in $Q_{n,k}(t)$, i.e., $(1 - u_{n,k}(t))h_{n,k}(t)Q_{n,k}(t-1) - \varepsilon_{n,k}(t) + \sum_{m \in \mathcal{M}} a_{n,m}(t)z_{n,m,k}(t-1)\sigma_{n,k}(t)$. Therefore, since the relaxed problem is an instance of the MKP, which is well-known as NP-hard, the original $\mathcal{P}_1$ is also NP-hard. $\square$

Clearly, solving problem $\mathcal{P}_1$ directly is very challenging because i) all decision variables are discrete and coupled in the objective function $\mathcal{Q}^{sys}$ as well as constraint (13h), making $\mathcal{P}_1$ becomes a non-convex integer programming problem, and $\mathcal{P}_1$ must be NP-hard according to Theorem 1; ii) PE's mobilities and service preferences are both uncertain and time-varying, and thus $\mathcal{P}_1$ has to be optimized dynamically by designing an appropriate online optimization approach; iii) function $f_{n,k}(\cdot)$ is in fact unknown and fundamentally unattainable, meaning that $\omega_{n,k}(t)$ is an intrinsic factor, necessitating a learning algorithm compatible with the online solution.

IV. AN ADVERSARIAL BANDIT LEARNING ASSISTED ONLINE OPTIMIZATION APPROACH

In this section, we propose a novel approach, called ARBOK, for efficiently addressing the formulated problem $\mathcal{P}_1$. Specifically, we first decompose the long-term problem into a series of instant problems using the Lyapunov decomposition method. Then, for each instant problem corresponding to each time slot $t \in \mathcal{T}$, we further decouple it into two interrelated subproblems, which are alternately solved by the developed MO-EL and AC-BL algorithms, respectively, till convergence.

A. Problem Decomposition

The key to decomposing problem $\mathcal{P}_1$ lies in addressing its objective function and constraint (13h), which are both in long-term average forms. Above all, following the Lyapunov method [49], we define a virtual deployment cost queue $C_n(t)$ to describe how the total cost $C_n^{tol}(t)$ introduced by dynamic placement and continual update of each DT $n \in \mathcal{N}$ in each time slot $t \in \mathcal{T}$ may

deviate from the cost threshold $C^{\max}$, and express its dynamic evolution as

$$C_n(t+1) = \max[C_n(t) + C_n^{tol}(t) - C^{\max}, 0]. \quad (14)$$

Next, we combine deployment cost queues for all DTs by a vector as $\Theta(t) = [C_1(t), \dots, C_n(t), \dots, C_N(t)]$, and construct a quadratic Lyapunov function as

$$L(\Theta(t)) \triangleq \frac{1}{2} \sum_{n \in \mathcal{N}} C_n(t)^2, \quad (15)$$

which quantitatively reflects the congestion of all queues, and should be persistently pushed towards a minimum value to keep queue stabilities. Referring to [50], the conditional Lyapunov drift becomes

$$\Delta(\Theta(t)) = \mathbb{E}[L(\Theta(t+1)) - L(\Theta(t)) | \Theta(t)], \quad (16)$$

where $\mathbb{E}[\cdot]$ stands for the expectation, and $\Delta(\Theta(t))$ captures the value difference of the Lyapunov function between two adjacent time slots. Clearly, by minimizing the conditional Lyapunov drift in (16), we can prohibit the backlogs of deployment cost queues from the unbounded growth, and thus guarantee that the long-term system cost constraint, i.e., (13h), can be satisfied. Accordingly, the Lyapunov drift-plus-penalty function for problem $\mathcal{P}_1$ can be written as

$$\Delta(\Theta(t)) - V \mathbb{E} \left[\sum_{n \in \mathcal{N}} \sum_{k \in \mathcal{K}} \omega_{n,k}(t) Q_{n,k}(t) \mid \Theta(t) \right], \quad (17)$$

where $V > 0$ is introduced as a Lyapunov control parameter, representing the weight of significance on maximizing the QoS versus that of strictly satisfying system cost constraint.

Theorem 2: Let $V > 0$, and the drift-plus-penalty is bounded by any possible decisions in any time slot $t \in \mathcal{T}$, i.e.,

$$\begin{aligned} \Delta(\Theta(t)) - V \mathbb{E} \left[\sum_{n \in \mathcal{N}} \sum_{k \in \mathcal{K}} \omega_{n,k}(t) Q_{n,k}(t) \mid \Theta(t) \right] \\ \leq G + \mathbb{E} \left[\sum_{n \in \mathcal{N}} C_n(t) (C_n^{tol}(t) - C^{\max}) \mid \Theta(t) \right] \\ - V \mathbb{E} \left[\sum_{n \in \mathcal{N}} \sum_{k \in \mathcal{K}} \omega_{n,k}(t) Q_{n,k}(t) \mid \Theta(t) \right], \end{aligned} \quad (18)$$

where $G = \frac{1}{2} \sum_{n \in \mathcal{N}} (C_n(\max) - C^{\max})^2$ is a positive constant, representing the upper bound of the Lyapunov drift-plus-penalty function.

Proof: Please see Appendix B, available online. $\square$

Theorem 2 reveals that the Lyapunov drift-plus-penalty function is upper bounded by a positive constant G in each time slot $t \in \mathcal{T}$. Based on this, problem $\mathcal{P}_1$ can then be decomposed into multiple subproblems, each of which aims to minimize the right-hand-side of (18), reformulating as

$$\begin{aligned} \mathcal{P}_2 : \quad & \min_{\mathcal{J}^{sys}(t)} \sum_{n \in \mathcal{N}} C_n(t) C_n^{tol}(t) - V \sum_{n \in \mathcal{N}} \sum_{k \in \mathcal{K}} \omega_{n,k}(t) Q_{n,k}(t) \\ \text{s.t.} \quad & (13a)-(13g). \end{aligned}$$

Note that, although we have converted the long-term problem $\mathcal{P}_1$ into a series of instant problems $\mathcal{P}_2$, all decisions remain

unchanged, meaning that $\mathcal{P}_2$ is still a non-convex optimization problem, containing the intrinsic factor $\omega_{n,k}(t)$.

B. Problem Decoupling

As observed from $\mathcal{P}_2$, since DTs are dynamically placed before the continual update and their profiles are preserved afterward, the associated decisions $a_{n,m}(t)$ and $z_{n,m,k}(t)$ primarily resolve the DT migration and profile preservation, constituting prerequisites for the rest. In contrast, the intrinsic factor $\omega_{n,k}(t)$ serves as the key enabler of the DT update, and $u_{n,k}(t)$ and $h_{n,k}(t)$ are key decisions for the involving personalized ability improvement and versatile ability maintenance, respectively. Therefore, to facilitate problem solving, we decouple $\mathcal{P}_2$ into the following equivalent form:

$$\mathcal{P}_3 : \min_{a_{n,m}(t), z_{n,m,k}(t)} \sum_{n \in \mathcal{N}} C_n(t) C_n^{tol}(t) - V \sum_{n \in \mathcal{N}} \sum_{k \in \mathcal{K}} \omega_{n,k}(t) Q_{n,k}(t)$$

s.t. (13a), (13e), (13f).

$$\mathcal{P}_4 : \min_{u_{n,k}(t), h_{n,k}(t)} \sum_{n \in \mathcal{N}} C_n(t) (C_n^{dl}(t) + C_n^{ul}(t) + C_n^{rud}(t) + C_n^{cache}(t)) - V \sum_{n \in \mathcal{N}} \sum_{k \in \mathcal{K}} \omega_{n,k}(t) Q_{n,k}(t)$$

s.t. (13b)–(13d), (13g).

With such decoupling, we now aim to respectively solve subproblems $\mathcal{P}_3$ and $\mathcal{P}_4$, where subproblem $\mathcal{P}_3$ strives to optimize DT placement $a_{n,m}(t)$ and profile caching $z_{n,m,k}(t)$, while subproblem $\mathcal{P}_4$ targets to determine fetched experiential knowledge type $u_{n,k}(t)$ and profile integration type $h_{n,k}(t)$, and learn service priority $\omega_{n,k}(t)$ in each time slot $t \in \mathcal{T}$. It is worth noting that since these two subproblems share common terms in their objective functions, their decisions are correlated, indicating that they should be alternately tackled till the convergence.

C. McCormick Envelopes Based Algorithm Integrated in ARBOK (MO-EL)

In this subsection, we introduce the solution to subproblem $\mathcal{P}_3$, assuming that the solution to subproblem $\mathcal{P}_4$ is already given (i.e., the optimal $u_{n,k}(t)$ and $h_{n,k}(t)$ are pre-known), and that the ground-truth $\omega_{n,k}(t)$ is already known. In fact, $\mathcal{P}_3$ is still a non-convex nonlinear integer programming problem, while it can be seen as a bilinear one, meaning that $\mathcal{P}_3$ becomes linear if any one of the decision variables is fixed. This motivates us to design an MO-EL algorithm [51] integrated in the proposed ARBOK approach, which first linearizes the bilinear terms and relaxes all discrete decision variables into continuous variables, transforming the problem into linear programming.

Specifically, relaxing discrete decision variables $a_{n,m}(t)$ and $z_{n,m,k}(t)$ into continuous forms (i.e., $\tilde{a}_{n,m}(t)$, $\tilde{z}_{n,m,k}(t) \in$

Algorithm 1: MO-EL Algorithm for Solving $\mathcal{P}_3$.

Input: Given the optimal $u_{n,k}(t)$, $h_{n,k}(t)$, $\omega_{n,k}(t)$, and current backlogs of deployment cost queues $C_n(t)$.

- 1 Linear relaxation: $a_{n,m}(t) \in \{0, 1\} \rightarrow \tilde{a}_{n,m}(t) \in [0, 1]$ and $z_{n,m,k}(t) \in \{0, 1\} \rightarrow \tilde{z}_{n,m,k}(t) \in [0, 1]$;
- 2 Denote $\tilde{\phi}_{n,m,m',k}(t) = \tilde{a}_{n,m'}(t) \tilde{z}_{n,m,k}(t)$;
- 3 Reformulate problem $\mathcal{P}_3$ as $\mathcal{P}'_3$;
- 4 Solve problem $\mathcal{P}'_3$ to get the fractional solution $\{\tilde{a}_{n,m'}(t), \tilde{z}_{n,m,k}(t), \tilde{\phi}_{n,m,m',k}(t)\}$;
- 5 Round $\tilde{a}_{n,m'}(t) \in [0, 1]$ to $a_{n,m'}(t) \in \{0, 1\}$ under (13a), (13f), (19b) and (19c) with decided $\{\tilde{z}_{n,m,k}(t), \tilde{\phi}_{n,m,m',k}(t)\}$;
- 6 Solve problem $\mathcal{P}'_3$ with fixed $a_{n,m'}(t)$ to get the fractional solution $\{\tilde{z}_{n,m,k}(t), \tilde{\phi}_{n,m,m',k}(t)\}$;
- 7 Round $\tilde{z}_{n,m,k}(t) \in [0, 1]$ to $z_{n,m,k}(t) \in \{0, 1\}$ under (13e), (13f), (19b) and (19d) with fixed $\{a_{n,m'}(t), \tilde{\phi}_{n,m,m',k}(t)\}$;

Output: Solution to $\mathcal{P}_3$:

$$\{a_{n,m}(t), z_{n,m,k}(t)\}, \forall n \in \mathcal{N}, m \in \mathcal{M}, k \in \mathcal{K}.$$

$[0, 1]$), and letting $\tilde{\phi}_{n,m,m',k}(t) = \tilde{a}_{n,m'}(t) \tilde{z}_{n,m,k}(t)$ be the auxiliary variable of bivariate $\tilde{a}_{n,m'}(t) \tilde{z}_{n,m,k}(t)$, $\mathcal{P}_3$ can be transformed as

$$\mathcal{P}'_3 : \min_{a_{n,m}(t), z_{n,m,k}(t), \tilde{\phi}_{n,m,m',k}(t)} \sum_{n \in \mathcal{N}} C_n(t) C_n^{tol}(t) - V \sum_{n \in \mathcal{N}} \sum_{k \in \mathcal{K}} \omega_{n,k}(t) Q_{n,k}(t)$$

s.t. (13a), (13e), (13f),

$$\tilde{\phi}_{n,m,m',k}(t) \geq 0, \quad (19a)$$

$$\tilde{\phi}_{n,m,m',k}(t) \geq \tilde{a}_{n,m'}(t) + \tilde{z}_{n,m,k}(t) - 1, \quad (19b)$$

$$\tilde{\phi}_{n,m,m',k}(t) \leq \tilde{a}_{n,m'}(t), \quad (19c)$$

$$\tilde{\phi}_{n,m,m',k}(t) \leq \tilde{z}_{n,m,k}(t), \quad (19d)$$

where constraints (19a)–(19d) are corresponding envelopes of the auxiliary variable $\tilde{\phi}_{n,m,m',k}(t)$.

Theorem 3: $\mathcal{P}'_3$ is a convex optimization problem.

Proof: Referring to [52], we can prove that problem $\mathcal{P}'_3$ is convex by verifying the convexity of its objective function and all constraints. On the one hand, the objective function is linear and all included decision variables are continuous. On the other hand, constraints (13a) and (13e) form hyperplanes, while the other constraints form halfspaces in $\mathbb{R}^{NM^2-K}$. All these together indicate that $\mathcal{P}'_3$ is convex. $\square$

Thanks to Theorem 3, we can easily address problem $\mathcal{P}'_3$ by leveraging the existing software-based optimization solvers (e.g., CVX [53]), and obtain the optimal fractional solution, i.e., $\{\tilde{a}_{n,m'}(t), \tilde{z}_{n,m,k}(t)\}$. Lastly, to prevent decision variables from violating constraints (13a), (13e), (13f) and (19a)–(19d), we further round the fractional solution into the integer form by employing a decision-dependent rounding strategy. Particularly, we first round the fractional solution $\tilde{a}_{n,m'}(t)$ into the integer one $a_{n,m'}(t)$ under associated constraints (13a), (13f) and (19b)

and (19c) with the fractional solution $\{\tilde{z}_{n,m,k}(t), \tilde{\phi}_{n,m,m',k}(t)\}$. Next, we substitute $a_{n,m'}(t)$ into $\mathcal{P}'_3$, and re-solve this problem with the solvers to obtain a new fractional solution $\{\tilde{z}'_{n,m,k}(t), \tilde{\phi}'_{n,m,m',k}(t)\}$. Finally, we round the fractional solution $\tilde{z}'_{n,m,k}(t)$ into the integer one $z_{n,m,k}(t)$ under associated constraints (13e), (13f), (19b) and (19d) with the solution $\{a_{n,m'}(t), \tilde{\phi}'_{n,m,m',k}(t)\}$. Note that since $\phi_{n,m,m',k}(t)$ is an auxiliary variable, it is unnecessary to explicitly deal with it, and its associated constraints are automatically satisfied in the above steps. Algorithm 1 illustrates the details of the overall MO-EL algorithm, which can eventually produce the solution $\{a_{n,m}(t), z_{n,m,k}(t)\}_{\forall n \in \mathcal{N}, m \in \mathcal{M}, k \in \mathcal{K}}$ to subproblem $\mathcal{P}_3$.

D. Adversarial Combinatorial Bandit Learning Algorithm Integrated in ARBOK (AC-BL)

Having the solution to $\mathcal{P}_3$, we now present the solution to subproblem $\mathcal{P}_4$. Evidently, solving $\mathcal{P}_4$ is even more challenging. This is because $\mathcal{P}_4$ contains an intrinsic factor $\omega_{n,k}(t)$ necessitating a learning process based on historical trials, while decisions $u_{n,k}(t)$ and $h_{n,k}(t)$ are also tightly coupled with $\omega_{n,k}(t)$, which results in a sequential decision-making process. To this end, we first transform $\mathcal{P}_4$ into an adversarial-combinatorial MAB problem, and then design the AC-BL algorithm for providing an effective outcome.

1) *Transformation to Adversarial-Combinatorial MAB Problem With Implicit Decision Feedback*: We recognize that subproblem $\mathcal{P}_4$ can be regarded as an adversarial-combinatorial MAB problem with implicit decision feedback. The reason is threefold: i) the discrete decision variables $u_{n,k}(t)$ and $h_{n,k}(t)$ can be seen as selectable arms, meanwhile the objective function of $\mathcal{P}_4$ can be viewed as the loss function associated with each selected arm; ii) the intrinsic factor $\omega_{n,k}(t)$ can be treated as a latent parameter in the loss function, hindering the direct observation of the true loss for each arm [54]; and iii) the loss revealed for selecting each arm may induce the change of $\omega_{n,k}(t)$, which in turn triggers the variation of the expected loss of each arm, and this can be interpreted as the presence of an adversary that strategically manipulates the average loss associated with each arm [55]. Furthermore, it is observed from the formulation of $\mathcal{P}_4$ that decision variables $u_{n,k}(t)$ and $h_{n,k}(t)$, and the latent parameter $\omega_{n,k}(t)$ for different DTs are actually independent with each other, indicating that optimizing $\mathcal{P}_4$ is equivalent to concurrently solving multiple distinct adversarial-combinatorial MAB problems, each of which corresponds to one specific DT $n \in \mathcal{N}$, as described below.

Adversarial-combinatorial MAB problem $\mathcal{P}_4^{(n)}, \forall n \in \mathcal{N}$: Formally, $\mathcal{P}_4^{(n)}$ for each DT $n \in \mathcal{N}$ consists of a combinatorial arm set and a loss function.

- *Combinatorial arm set*: Within each time slot $t \in \mathcal{T}$, the combinatorial arm set of $\mathcal{P}_4^{(n)}$ is defined as the set of decision variables $\{u_{n,k}(t)\}_{\forall k \in \mathcal{K}}$ and $\{h_{n,k}(t)\}_{\forall k \in \mathcal{K}}$, which can be expressed as

$$\mathcal{S}_n(t) = \{u_{n,k}(t), h_{n,k}(t) | \forall k \in \mathcal{K}, (13b)-(13d)\}. \quad (20)$$

- *Loss function*: Within each time slot $t \in \mathcal{T}$, the loss function of $\mathcal{P}_4^{(n)}$ is defined as the objective function of $\mathcal{P}_4$ with respect to DT n only, i.e.,

$$\begin{aligned} \mathcal{L}_n(t) = & C_n(t)(C_n^{dl}(t) + C_n^{ul}(t) + C_n^{rd}(t) \\ & + C_n^{cache}(t)) - V \sum_{k \in \mathcal{K}} \omega_{n,k}(t) Q_{n,k}(t), \end{aligned} \quad (21)$$

which includes a latent parameter $\omega_{n,k}(t)$.

2) *Solution to Adversarial-Combinatorial MAB Problem*: Next, for any DT $n \in \mathcal{N}$, we aim to solve its corresponding adversarial-combinatorial MAB problem $\mathcal{P}_4^{(n)}$. Note that, different from traditional MAB problems with explicit decision feedback [28], [54], the true loss of each arm in $\mathcal{P}_4^{(n)}$ is fundamentally unobtainable due to the latent parameter $\omega_{n,k}(t)$. Thus, we extend conventional MAB algorithms to solve the formulated adversarial-combinatorial MAB problem with implicit decision feedback by first constructing a combinatorial arm set, and then learning the latent parameter while selecting the optimal arm.

a) *Combinatorial arm set construction*: The complex inter-dependencies among different arms within the combinatorial arm set $\mathcal{S}_n(t)$ yield a vast number of candidate arms, which complicates the identification of the optimal arm. To address such an issue, we convert the combinatorial arm set into a super arm set by aggregating each pair of combinatorial arms (i.e., $u_{n,k}(t)$ and $h_{n,k}(t)$) into a single super arm. Specifically, let $\mathcal{S}_n^{supr}(t) = \{(i_{n,s}^u(t), j_{n,s}^h(t))_s | i_{n,s}^u(t), j_{n,s}^h(t) \in \{0, 1, \dots, K\}, i_{n,s}^u(t) \neq j_{n,s}^h(t) \vee i_{n,s}^u(t) = j_{n,s}^h(t) = 0, s \in \mathcal{S} = [1, K^2 + K + 1]\}$ be the converted super arm set of the original adversarial-combinatorial MAB problem $\mathcal{P}_4^{(n)}$, where each element $(i_{n,s}^u(t), j_{n,s}^h(t))_s$ stands for one selectable super arm. If super arm $(i_{n,s}^u(t) = k_1, j_{n,s}^h(t) = k_2)_s, k_1 \neq 0, k_2 \neq 0$ is selected, it indicates that the experiential knowledge of type $k_1 \in \mathcal{K}$ and the profile of type $k_2 \in \mathcal{K}$ are chosen to update DT $n \in \mathcal{N}$ in time slot t , i.e., $u_{n,k_1}(t) = 1$ and $h_{n,k_2}(t) = 1$. Otherwise, if $k_1 = 0$ or $k_2 = 0$, it means that we do not select any types of experiential knowledge or profile to update DT $n \in \mathcal{N}$ in time slot t , i.e., $u_{n,k}(t) = 0, \forall k \in \mathcal{K}$ or $h_{n,k}(t) = 0, \forall k \in \mathcal{K}$.

b) *Latent parameter learning and optimal arm selection*: Based on the above, to further deal with the implicit loss feedback, we design an exploration-exploitation scheme to simultaneously learn the latent parameter and select the optimal super arm from $\mathcal{S}_n^{supr}(t)$, for determining decision variables $u_{n,k}(t)$ and $h_{n,k}(t)$ and inferring the latent parameter $\omega_{n,k}(t)$. Recall that $\mathcal{P}_3$ and $\mathcal{P}_4$ must be alternately solved in each time slot $t \in \mathcal{T}$, and each super arm's loss should be estimated by every alternation rather than every slot once. Denote τ as the index of alternating round. Then, in each alternating round τ of time slot $t \in \mathcal{T}$, we can substitute decisions $u_{n,k}(t, \tau - 1)$, $h_{n,k}(t, \tau - 1)$ together with the on-going learning parameter $\omega_{n,k}(t, \tau - 1)$ from the previous alternating round $\tau - 1$ into the loss function $\mathcal{L}_n(t)$ in (21) to compute the corresponding loss value $\mathcal{L}_n(t, \tau - 1)$, and obtain the cumulative loss for each type $k \in \mathcal{K}$ as

$$\mathcal{L}_{n,k}(t, \tau) = \mathcal{L}_{n,k}(t, \tau - 1) + \mathbb{I}\{u_{n,k}(t, \tau - 1) = 1$$

$$\forall h_{n,k}(t, \tau - 1) = 1\} \mathcal{L}_n(t, \tau - 1), \quad (22)$$

where $\mathbb{I}\{u_{n,k}(t, \tau - 1) = 1 \vee h_{n,k}(t, \tau - 1) = 1\}$ denotes an indicator function signifying whether $u_{n,k}(t, \tau - 1)$ or $h_{n,k}(t, \tau - 1)$ is set to 1 in the previous alternating round $\tau - 1$ (i.e., $\mathbb{I}\{u_{n,k}(t, \tau - 1) = 1 \vee h_{n,k}(t, \tau - 1) = 1\} = 1$ if $u_{n,k}(t, \tau - 1)$ or $h_{n,k}(t, \tau - 1)$ is set to 1, and $\mathbb{I}\{u_{n,k}(t, \tau - 1) = 1 \vee h_{n,k}(t, \tau - 1) = 1\} = 0$ otherwise). Next, by incorporating the cumulative loss and PE's service preferences, which are correlated with the service priority, we design an adaptive UCB-based learning mechanism to infer $\omega_{n,k}(t)$ as [54]

$$\tilde{\omega}_{n,k}(t, \tau) = \frac{\mathcal{L}_{n,k}(t, \tau)}{\gamma_{n,k}(t, \tau)} + \sqrt{\frac{\pi_{n,k}(t) \log t}{\gamma_{n,k}(t, \tau) + 1}}, \quad (23)$$

$$\omega_{n,k}(t, \tau) = \frac{\tilde{\omega}_{n,k}(t, \tau)}{\sum_{j \in \mathcal{K}} \tilde{\omega}_{n,j}(t, \tau)}, \quad (24)$$

where $\gamma_{n,k}(t, \tau)$ represents the cumulative number of times that service request type k has been selected up to the alternating round τ in time slot $t \in \mathcal{T}$.

Since the variation of latent parameter $\omega_{n,k}(t, \tau)$, defined in (23), in turn alters the expected loss of each arm, we recalculate the value of the loss function $\mathcal{L}_n(t, \tau)$ by incorporating the newly learnt $\omega_{n,k}(t, \tau)$, along with decisions $u_{n,k}(t, \tau - 1)$, $h_{n,k}(t, \tau - 1)$ from the previous alternating round $\tau - 1$ of time slot $t \in \mathcal{T}$. Similar to [55], to handle the dynamically changing expected loss of each super arm, we further construct an estimated loss for each super arm $s \in \mathcal{S}$ in the alternating round τ of time slot $t \in \mathcal{T}$, using the recalculated loss value $\mathcal{L}_n(t, \tau)$, which can be written as

$$\hat{l}_{n,s}(t, \tau) = \frac{2\mathbb{I}\{\mathcal{S}_n^{supr}(t, \tau - 1) = s\} \mathcal{L}_n(t, \tau)}{2P_{n,s}(t, \tau - 1) + \mu_{n,s}(t, \tau)}, \quad (25)$$

where $\mu_{n,s}(t, \tau) = \sqrt{\frac{\log S}{S(t+\tau)}}$ denotes the learning rate of each super arm s ; $P_{n,s}(t, \tau - 1)$ represents the sampling probability of each super arm s , which will be expanded later; and $\mathbb{I}\{\mathcal{S}_n^{supr}(t, \tau - 1) = s\}$ is an indicator function signifying whether the s -th super arm is selected from the super arm set $\mathcal{S}_n^{supr}(t)$ in the previous alternating round $\tau - 1$ (i.e., $\mathbb{I}\{\mathcal{S}_n^{supr}(t, \tau - 1) = s\} = 1$ if the s -th arm is selected, and $\mathbb{I}\{\mathcal{S}_n^{supr}(t, \tau - 1) = s\} = 0$ otherwise). Next, we can express the cumulative estimated loss of each super arm $s \in \mathcal{S}$ till the alternating round τ in time slot $t \in \mathcal{T}$ as

$$\hat{\mathcal{L}}_{n,s}(t, \tau) = \hat{\mathcal{L}}_{n,s}(t, \tau - 1) + \hat{l}_{n,s}(t, \tau). \quad (26)$$

Then, referring to [56], we can acquire the sampling probability of each super arm $s \in \mathcal{S}$ in each alternating round τ of time slot $t \in \mathcal{T}$ as

$$P_{n,s}(t, \tau) = \frac{\exp(-\mu_{n,s}(t, \tau) \hat{\mathcal{L}}_{n,s}(t, \tau))}{\sum_{s' \in \mathcal{S}} \exp(-\mu_{n,s'}(t, \tau) \hat{\mathcal{L}}_{n,s'}(t, \tau))}. \quad (27)$$

Lastly, let $P_n(t, \tau) = \{P_{n,s}(t, \tau) | s \in \mathcal{S}\}$ be the joint sampling probability distribution over the super arm set $\mathcal{S}_n^{supr}(t)$. We sample one super arm $(i_{n,s}^{u,*}(t, \tau), j_{n,s}^{h,*}(t, \tau))_s$ from $\mathcal{S}_n^{supr}(t)$ under $P_n(t, \tau)$, yielding the optimal $u_{n,k}(t, \tau)$ and $h_{n,k}(t, \tau)$

Algorithm 2: AC-BL Algorithm for Solving $\mathcal{P}_4$.

Input: The optimal $a_{n,m}(t)$ and $z_{n,m,k}(t)$ in solving $\mathcal{P}_3$, current backlogs of deployment cost queues $C_n(t)$.

- 1 Equivalently transform solving $\mathcal{P}_4$ into concurrently solving multiple adversarial-combinatorial MAB problems $\mathcal{P}_4^{(n)}$, each of which is associated with a specific DT $n \in \mathcal{N}$;
- 2 **for any adversarial-combinatorial MAB subproblem $\mathcal{P}_4^{(n)}$ do**
- 3 Set the alternation round $\tau = 1$;
- 4 **while alternating between $\mathcal{P}_3$ and $\mathcal{P}_4$ do not converge do**
- 5 Convert combinatorial arm set $\mathcal{S}_n(t)$ into a super set $\mathcal{S}_n^{supr}(t) = \{(i_{n,s}^u(t), j_{n,s}^h(t))_s | i_{n,s}^u(t), j_{n,s}^h(t) \in \{0, 1, \dots, K\}, i_{n,s}^u(t) \neq j_{n,s}^h(t) \vee i_{n,s}^u(t) = j_{n,s}^h(t) = 0, s \in [1, K^2 + K + 1]\}$;
- 6 Substitute $u_{n,k}(t, \tau - 1)$, $h_{n,k}(t, \tau - 1)$ and $\omega_{n,k}(t, \tau - 1)$ from the alternating round $\tau - 1$ into loss function $\mathcal{L}_n(t)$ to compute $\mathcal{L}_n(t, \tau - 1)$;
- 7 Adjust the cumulative loss for each type $k \in \mathcal{K}$ as (22);
- 8 Learn the latent parameter $\omega_{n,k}(t, \tau)$ by using the normalized adaptive confidence upper bound of each type $k \in \mathcal{K}$ as (23) and (24);
- 9 Recalculate the value of loss function $\mathcal{L}_n(t, \tau)$ by incorporating latent parameter $\omega_{n,k}(t, \tau)$ along with decisions $u_{n,k}(t, \tau - 1)$, $h_{n,k}(t, \tau - 1)$ from the previous alternating round $\tau - 1$;
- 10 Calculate the estimated loss $\hat{l}_{n,s}(t, \tau)$ of each super arm $s \in \mathcal{S}$ by (25);
- 11 Update the cumulative estimated loss $\hat{\mathcal{L}}_{n,s}(t, \tau)$ of each super arm $s \in \mathcal{S}$ by (26);
- 12 Calculate the sampling probability $P_{n,s}(t, \tau)$ of each super arm $s \in \mathcal{S}$ as (27);
- 13 Construct the joint sampling probability distribution of the super arm set $\mathcal{S}_n^{supr}(t)$ as $P_n(t, \tau) = \{P_{n,s}(t, \tau) | s \in \mathcal{S}\}$;
- 14 Sample one super arm $(i_{n,s}^{u,*}(t, \tau), j_{n,s}^{h,*}(t, \tau))_s$ from the super arm set $\mathcal{S}_n^{supr}(t)$ under probability distribution $P_n(t, \tau)$;
- 15 **if $i_{n,s}^{u,*}(t, \tau) = 0$ or $j_{n,s}^{h,*}(t, \tau) = 0$ then**
- 16 Set $u_{n,k}(t, \tau) = 0$ or $h_{n,k}(t, \tau) = 0$;
- 17 **else**
- 18 Set $u_{n,i_{n,s}^{u,*}(t, \tau)}(t, \tau) = 1$, $h_{n,j_{n,s}^{h,*}(t, \tau)}(t, \tau) = 1$, and other decisions to zero, i.e., $u_{n,k'}(t, \tau) = 0, \forall k' \in \mathcal{K} \setminus \{i_{n,s}^{u,*}(t, \tau)\}$ and $h_{n,k''}(t, \tau) = 0, \forall k'' \in \mathcal{K} \setminus \{j_{n,s}^{h,*}(t, \tau)\}$;
- 19 Update the alternation round as $\tau = \tau + 1$;
- 20 Obtain the learned latent parameter $\omega_{n,k}(t)$, and the optimal selected arms $u_{n,k}(t)$ and $h_{n,k}(t)$ as solution to $\mathcal{P}_4^{(n)}$ in each time slot $t \in \mathcal{T}$;
- 21 Aggregate solutions $\{u_{n,k}(t), h_{n,k}(t)\}$ and learned intrinsic factors $\omega_{n,k}(t)$ from all parallel subproblems $\mathcal{P}_4^{(n)}$, $\forall n \in \mathcal{N}$ in each time slot $t \in \mathcal{T}$;

Output: Solution $\{u_{n,k}(t), h_{n,k}(t)\}$ and learned intrinsic factors $\omega_{n,k}(t)$ of $\mathcal{P}_4$.

in each alternating round τ of time slot $t \in \mathcal{T}$. The alternating process between problems $\mathcal{P}_3$ and $\mathcal{P}_4$ continues until the convergence, outputting the learned latent parameters $\omega_{n,k}(t)$, and the final selected arms $u_{n,k}(t)$ and $h_{n,k}(t)$ as the solution to each subproblem $\mathcal{P}_4^{(n)}$ in each time slot $t \in \mathcal{T}$. Finally, by aggregating solutions $\{u_{n,k}(t), h_{n,k}(t)\}$ and intrinsic factors $\omega_{n,k}(t)$ from all $\mathcal{P}_4^{(n)}$, $\forall n \in \mathcal{N}$ that are solved in parallel, the solution to subproblem $\mathcal{P}_4$ can be obtained. The detailed steps of the overall AC-BL algorithm for solving subproblem $\mathcal{P}_4$ is illustrated in Algorithm 2.

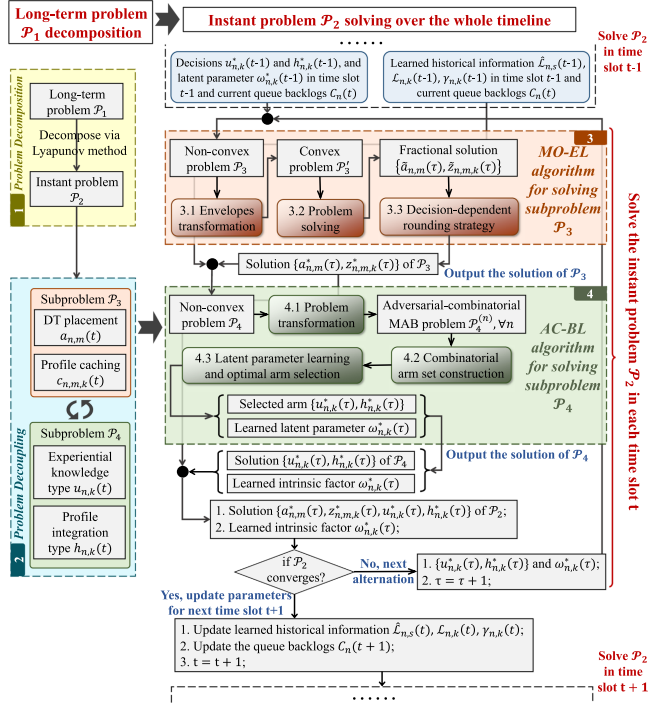


Fig. 3. The flowchart of the proposed ARBOK approach.

E. Analyses of ARBOK

In summary, our proposed approach, namely ARBOK, first decomposes the long-term problem $\mathcal{P}_1$ into a series of instant problems $\mathcal{P}_2$. Then, it further decouples problem $\mathcal{P}_2$ into two interrelated subproblems $\mathcal{P}_3$ and $\mathcal{P}_4$, which are alternately solved by the designed MO-EL and AC-BL algorithms, respectively, till the convergence. The flowchart of ARBOK is shown in Fig. 3.

In the following, we analyze the theoretical performance of the proposed ARBOK approach in terms of its convergence, optimality gap and computational complexity.

Lemma 1 (Convergence of MO-EL): The MO-EL algorithm can converge with limited rounding steps.

Proof: Please see Appendix C, available online. $\square$

Lemma 2 (Regret Analysis of AC-BL): The cumulative regret of AC-BL algorithm, denoted by $\hat{R}(T)$, is defined as the cumulative difference between the incurred cumulative loss by the AC-BL algorithm and that of the offline optimal solution with all information known in advance over T time slots. For any $\delta > 0$, the cumulative regret $\hat{R}(T)$ of AC-BL algorithm can be shown as

$$\mathbb{P} \left[\hat{R}(T) \geq \sqrt{N^2(ST+2) \log S} + \left(\sqrt{\frac{N^2ST}{\log S}} + N \right) \log \frac{S+1}{\delta} \right] \leq \delta, \quad (28)$$

which indicates that AC-BL algorithm will ultimately reach a sublinear regret.

Proof: Please see Appendix D, available online. $\square$

Theorem 4 (Convergence of ARBOK): The proposed ARBOK approach can converge with limited alternations.

Proof: With Lemmas 1 and 2, we are now ready to prove the convergence of ARBOK, which highly depends on that of the alternating process between MO-EL and AC-BL algorithms. According to the monotone convergence theorem [57], ARBOK can converge if two conditions are satisfied, i.e., the monotonicity and the boundedness of $\mathcal{P}_2$'s objective function over the alternating process.

Monotonicity of $\mathcal{P}_2$'s objective function: First, we can scale $\mathcal{P}_2$'s objective function with the alternating steps as

$$\begin{aligned} obj^{\tau-1}(\mathcal{P}_2) &= obj(\mathcal{P}_2)(a^{\tau-1}, z^{\tau-1}, u^{\tau-1}, h^{\tau-1}, \omega^{\tau-1}) \\ &\geq obj(\mathcal{P}_2)(a^{\tau-1}, z^{\tau-1}, u^{\tau}, h^{\tau}, \omega^{\tau}) \\ &\geq obj(\mathcal{P}_2)(a^{\tau}, z^{\tau}, u^{\tau}, h^{\tau}, \omega^{\tau}) = obj^{\tau}(\mathcal{P}_2), \end{aligned} \quad (29)$$

where the first inequality holds because AC-BL algorithm can gradually approach the optimal solution during the alternating process, as guaranteed by the sublinear regret proved in Lemma 2; the second inequality holds due to the convergence of MO-EL algorithm as shown in Lemma 1. These reveal that the objective function of $\mathcal{P}_2$ is monotonically decreasing along with the alternating process.

Boundedness of $\mathcal{P}_2$'s objective function: Then, we prove that the objective function of $\mathcal{P}_2$ has one lower bound, preventing it from decreasing towards negative infinity with the alternating process. By setting the queue backlogs to 0 and all $\omega_{n,k}(t)$ and $Q_{n,k}(t)$ to 1, and substituting them into the objective function of $\mathcal{P}_2$, a lower bound can be easily derived as $-V N K$.

Therefore, since the objective function of $\mathcal{P}_2$ decreases monotonically in the alternating process, and it is lower bounded by $-V N K$, we can conclude that the proposed ARBOK approach will converge with limited alternations.

Lemma 3 (Optimality Gaps of MO-EL and AC-BL): Let $\mathcal{O}(\mathcal{P}_3)$ and $\mathcal{O}^*(\mathcal{P}_3)$ be the objective of $\mathcal{P}_3$ by inputting solutions obtained from MO-EL, and the theoretical optimal one, respectively. The optimality gap of MO-EL algorithm can be expressed as

$$\mathcal{O}(\mathcal{P}_3) - \mathcal{O}^*(\mathcal{P}_3) \leq \sum_{n \in \mathcal{N}} (\Psi_{n,1}^C + \Psi_{n,2}^C) - V \sum_{n \in \mathcal{N}} \Psi_n^Q, \quad (30)$$

where $\Psi_{n,1}^C$ denotes the gap on system cost induced by dynamic DT placement and profile preservation (i.e., $C_n^{pl}(t) + C_n^{cache}(t)$), $\Psi_{n,2}^C$ represents the gap on system cost generated from the transmission and computation of experiential knowledge for continual DT update (i.e., $C_n^{dl}(t) + C_n^{ul}(t) + C_n^{ud}(t)$), and Ψ_n^Q stands for the gap on QoS performance (i.e., $\sum_{k \in \mathcal{K}} \omega_{n,k}(t) Q_{n,k}(t)$) for each DT $n \in \mathcal{N}$.

Besides, let $\mathcal{O}(\mathcal{P}_4)$ and $\mathcal{O}^*(\mathcal{P}_4)$ be the objective of $\mathcal{P}_4$ by inputting solutions obtained from AC-BL algorithm, and the theoretical optimal one, respectively. The optimality gap of AC-BL can be derived as

$$\mathcal{O}(\mathcal{P}_4) - \mathcal{O}^*(\mathcal{P}_4) \leq \hat{R}(T)/T, \quad (31)$$

where $\hat{R}(T)$ is the cumulative regret of AC-BL algorithm, which has been obtained in Lemma 2.

Proof: Please see Appendix E, available online. $\square$

Theorem 5 (Optimality Gap of ARBOK): Given Lyapunov control parameter V , the optimality gap between the solution obtained by the proposed ARBOK approach and the theoretically optimal solution to problem $\mathcal{P}_1$ can be expressed as

$$\sum_{t \in \mathcal{T}} \mathbb{E} \left[\sum_{n \in \mathcal{N}} \sum_{k \in \mathcal{K}} \omega_{n,k}(t) Q_{n,k}(t) \mid \Theta(t) \right] / T - \mathcal{O}^* \leq G/V + \Upsilon + \Gamma, \quad (32)$$

where $\mathcal{O}^*$ stands for the theoretically optimal solution to $\mathcal{P}_1$, $\Upsilon = \sum_{n \in \mathcal{N}} (\Psi_{n,1}^C + \Psi_{n,2}^C) - V \sum_{n \in \mathcal{N}} \Psi_n^Q$ and $\Gamma = \widehat{\mathcal{R}}(T)/T$ represent the optimality gaps of MO-EL and AC-BL algorithms, respectively, which have been derived in Lemma 3.

Proof: The total optimality gap of the ARBOK algorithm arises from the Lyapunov decomposition, MO-EL algorithm and AC-BL algorithm. As optimality gaps of MO-EL and AC-BL algorithms are given in Lemma 3, respectively, our analysis herein focuses on the gap introduced by the Lyapunov decomposition. In fact, this gap originates from the scalability of the drift-plus-penalty term in inequality (18) of Theorem 2. Therefore, we expand and rewrite inequality (18) as

$$\begin{aligned} \Delta(\Theta(t)) - V \mathbb{E} \left[\sum_{n \in \mathcal{N}} \sum_{k \in \mathcal{K}} \omega_{n,k}(t) Q_{n,k}(t) \mid \Theta(t) \right] \\ \leq G + \mathbb{E} \left[\sum_{n \in \mathcal{N}} C_n(t) (C_n^{\text{tol}}(t) - C^{\text{max}}) \mid \Theta(t) \right] \\ - V \mathbb{E} \left[\sum_{n \in \mathcal{N}} \sum_{k \in \mathcal{K}} \omega_{n,k}(t) Q_{n,k}(t) \mid \Theta(t) \right] \\ \leq G - V \mathcal{O}^* + \Upsilon + \Gamma. \end{aligned} \quad (33)$$

Then, by summing up (33) over T time slots, we have

$$\begin{aligned} (G - V \mathcal{O}^* + \Upsilon + \Gamma) \cdot T \\ \geq \sum_{t \in \mathcal{T}} \left\{ \Delta(\Theta(t)) - V \mathbb{E} \left[\sum_{n \in \mathcal{N}} \sum_{k \in \mathcal{K}} \omega_{n,k}(t) Q_{n,k}(t) \mid \Theta(t) \right] \right\} \\ = \mathbb{E}[L(\Theta(T)) - L(\Theta(0))] \\ - V \sum_{t \in \mathcal{T}} \mathbb{E} \left[\sum_{n \in \mathcal{N}} \sum_{k \in \mathcal{K}} \omega_{n,k}(t) Q_{n,k}(t) \mid \Theta(t) \right] \\ = \mathbb{E}[L(\Theta(T))] - \mathbb{E}[L(\Theta(0))] \\ - V \sum_{t \in \mathcal{T}} \mathbb{E} \left[\sum_{n \in \mathcal{N}} \sum_{k \in \mathcal{K}} \omega_{n,k}(t) Q_{n,k}(t) \mid \Theta(t) \right]. \end{aligned} \quad (34)$$

Afterwards, by moving $\mathbb{E}[L(\Theta(0))]$ to the left-hand-side of (34), dividing both sides by TV , and taking Υ and Γ from Lemma 3, inequality (32) can be obtained. $\square$

Theorem 6 (Computational Complexity of ARBOK): The computational complexity of the proposed ARBOK approach is $O(TR^{\text{max}}((K+1)^2 + (NMK)^3 + 4.16(NM + NMK)))$, where N is the number of DTs, M is the number of ESs, K is the number of service request types, T is the number of

TABLE III
SIMULATION PARAMETERS

Parameter	Value	Parameter	Value
z_m^{max}	80 GB	K	60
c_m	300 cycles/bit	M	20
$D_{n,k}^p(t)$	[6.1, 12.2] MB	N_0	-174 dBm/Hz
$D_{n,k}^{\text{ddl}}(t)$	200 MB	p_c, p_m	5 W
D_k^{tr}	[54.9, 109.8] MB	p_n	500 mW
$D_{n,k}^{\text{prf}}$	150 MB	$r_{c,m}$	50 Mbps
F_m	20 GHz	θ	4
N	60	ρ_m	10^{-27}
ζ, χ, κ	{8, 0.04, 2}	C^{max}	7×10^6

time slots, and R^{max} is iteration numbers for alternately solving two involved subproblems.

Proof: The complexity of ARBOK mainly results from the alternating process between MO-EL and AC-BL algorithms, and their corresponding solutions. Specifically, for MO-EL algorithm, its complexity is determined by two primary stages: solving optimization problem $\mathcal{P}_3$ and the subsequent decision-dependent rounding process. According to [52], solving problem $\mathcal{P}_3$ with a standard interior-point solver has a complexity of $O((NMK)^3)$. The complexity of the rounding process, as detailed in [6], is $O(4.16(NM + NMK))$. For AC-BL algorithm, the computational complexity is introduced by the latent parameter learning and optimal arm selection in solving adversarial-combinatorial MAB problem $\mathcal{P}_4^{(n)}$. This process involves calculating the latent parameter $\omega_{n,k}(t)$ for all K arms and subsequently executing the operations in (25)–(27) for all $S = K^2 + K + 1$ super arms. Therefore, its complexity becomes $O(K + K^2 + K + 1) = O(K^2 + 2K + 1)$.

To sum up, if the maximum alternation between MO-EL and AC-BL algorithms is denoted by R^{max} , the complexity of ARBOK can be expressed as $O(TR^{\text{max}}((K+1)^2 + (NMK)^3 + 4.16(NM + NMK)))$.

V. SIMULATION RESULTS

In this section, extensive simulations are conducted to evaluate the performance of the proposed ARBOK approach for jointly optimizing dynamic DT placement and continual update. All simulation results are obtained by taking averages over 1000 runs under various parameter settings.

A. Simulation Settings

Consider an end-edge-cloud collaborative networking system in a 1 km $\times$ 1 km square area with $M = 20$ ESs and $N = 60$ PEs. Each PE adopts a Random-Waypoint mobility model under same settings as those in [58]. The mobilities of PEs and the deployed locations of ESs are set based on the real-world datasets adopted in [59], [60]. All key simulation parameters are listed in Table III. Particularly, parameters used in dynamic DT placement and continual update are chosen according to the medical image segmentation task in healthcare DT applications [11], [61], [62], [63], while network configurations are set similar to those in [6], [8], [9].

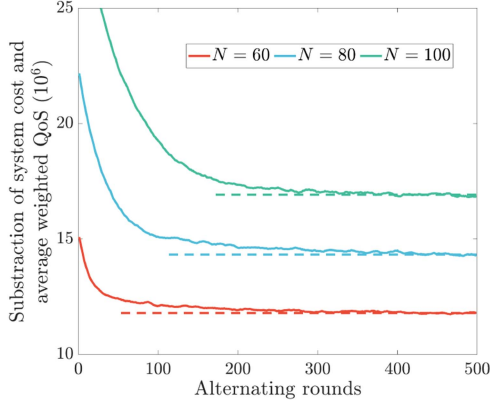


Fig. 4. Convergence of the ARBOK approach w.r.t N .

To evaluate the effectiveness of the introduced dynamic DT placement and continual update framework, several alternative schemes are simulated for comparison.

- *MARL* [24]: *MARL* dynamically places DTs without updating their service capabilities, implying that the QoS for each type of service request remains unchanged, regardless of the variation of each service request type's service priority.
- *PUFL* [15]: *PUFL* jointly manages dynamic DT placement and continual DT update by the personalized ability improvement. However, it does not enable the versatile ability maintenance via the preservation and integration of DT profiles.

Besides, to show the superiority of the proposed ARBOK approach in optimizing the long-term average weighted QoS, the following schemes are simulated as benchmarks.

- *GST* [64]: *GST* greedily optimizes all decision variables to maximize the QoS in tackling all service request types within each single time slot, completely ignoring the long-term system cost constraint.
- *SIO* [65]: *SIO* employs a particle swarm optimization algorithm to determine all decision variables, aiming to balance the QoS in tackling all types of service requests and system cost within each single time slot.
- *FPL* [28]: *FPL* adopts a similar optimization process with the ARBOK to maximize the QoS in tackling all types of service requests under the long-term system cost constraint, except that the fetched experiential knowledge type, profile integration type and service priority are determined by utilizing a Follow-the-Perturbed-Leader algorithm, rather than a carefully designed learning process.

B. Performance Evaluations

Fig. 4 examines the convergence of the proposed ARBOK approach, which can be illustrated by showing the variation of the subtraction of the system cost and all DTs' average weighted QoS in tackling all service request types (i.e., $\sum_{n \in \mathcal{N}} C_n(t) C_n^{tol}(t) - V \sum_{n \in \mathcal{N}} \sum_{k \in \mathcal{K}} \omega_{n,k}(t) Q_{n,k}(t)$) throughout the alternating rounds within a single time slot. We can see that the value decreases at the beginning and quickly

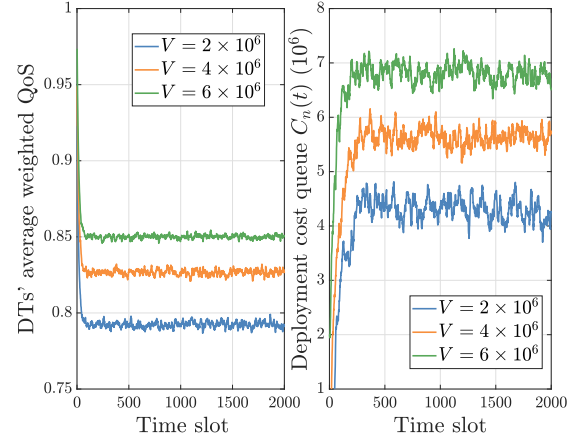


Fig. 5. Stability of DTs' average weighted QoS and deployment cost queue by varying V .

converges. This is because of not only the adoption of the decision-dependent rounding strategy in the MO-EL algorithm, which provides high-quality solution in each alternating round to accelerate the convergence (with additional experiments conducted in Appendix F, available online to quantify its approximation errors), but also the construction of combinatorial arm set in the AC-BL algorithm, which reduces the numbers of selectable arms to improve the exploration efficiency. Moreover, it can be observed that such subtraction value increases, while the convergence speed decreases with a larger N . This is because, as the number of DTs grows, more DTs need to be migrated and more experiential knowledge must be transmitted for dynamic placement and continual update, thereby increasing the resulted system cost and the decision-making complexity of the optimization problem. All these results align with the theoretical analysis proved in Theorem 4 and indicate the efficiency of ARBOK. To further validate the scalability of the proposed ARBOK approach (including the involved MO-EL and AC-BL algorithms), we also provide some experiments showing its running time under varying numbers of PEs/DTs N , DTs M and service request types K , and the obtained results are illustrated in Appendix G, available online.

Fig. 5 demonstrates the stability of the proposed ARBOK approach by presenting the long-term performance of DTs' average weighted QoS in tackling all types of service requests, i.e., $\frac{1}{N} \sum_{n \in \mathcal{N}} \sum_{k \in \mathcal{K}} \omega_{n,k}(t) Q_{n,k}(t)$, along with the deployment cost queue $C_n(t)$ brought by the Lyapunov method under different values of parameter V . We can see from Fig. 5 that the average weighted QoS decreases and the queue backlog increases initially, and both of them quickly stabilize over the time. This is because ARBOK focuses on controlling system cost to tackle the booming queue backlog at the beginning, and can eventually balance the average weighted QoS and system cost, leading to a stable outcome. Besides, this figure shows that both average weighted QoS and queue backlog stabilize on higher values with the increase of V . The reason is that Lyapunov parameter V is introduced to control the significance on maximizing the average weighted QoS versus that of controlling

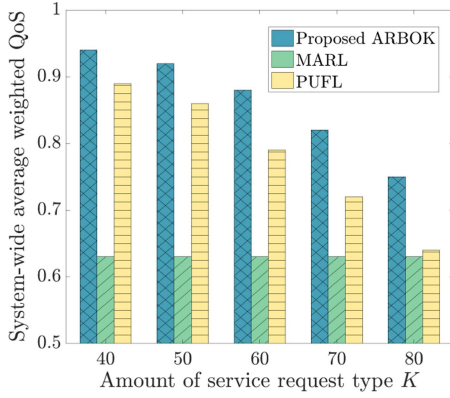
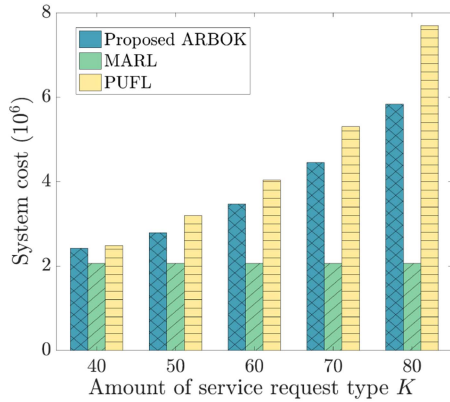
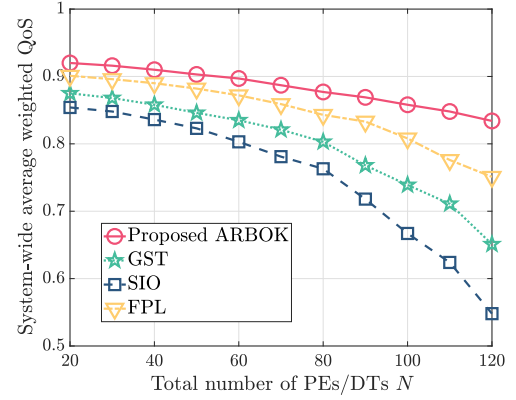
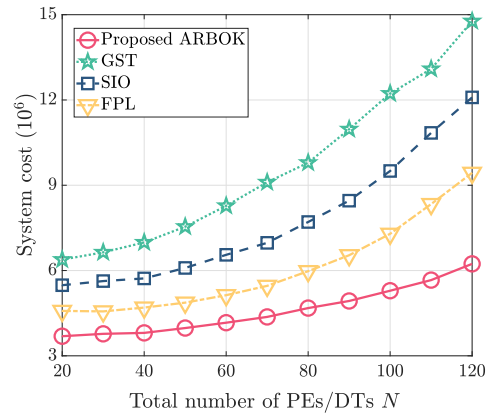
(a) System-wide average weighted QoS w.r.t. K .(b) System cost w.r.t. K .

Fig. 6. Performance comparison on different DT placement and update frameworks.

the system cost, and a larger V indicates more emphasis on the average weighted QoS.

Fig. 6 illustrates the superiority of the proposed dynamic DT placement and continual update framework compared with *MARL* and *PUFL* by varying the amount of service request type K . Since *MARL* cannot update the service capability of each DT, meaning that the experiential knowledge and profiles do not need to be computed and integrated to update DTs, the average weighted QoS and system cost remain unchanged regardless of K . In contrast, it can be observed from Fig. 6(a) and (b) that the average weighted QoS decreases and system cost increases with K for both *PUFL* and the proposed framework. This is because a larger K implies that each DT should handle more types of service requests throughout the whole process, by either transmitting experiential knowledge with larger data size or migrating to the distant ES for integrating the profile of particular service type. Moreover, we can see that the proposed framework further outperforms *PUFL* because the preservation and integration of profiles during the continual DT update helps to maintain the versatile ability of all service request types. To further quantify the system cost overheads of ARBOK, we also provide some experiments to examine the individual components of the system cost, with the corresponding results reported in Appendix H, available online.

Fig. 7 investigates both the system-wide average weighted QoS in tackling all types of service requests and the resulted

(a) System-wide average weighted QoS w.r.t. N .(b) System cost w.r.t. N .Fig. 7. Performance comparison on different optimization approaches by changing the total number of PEs/DTs N .

system cost, achieved by *GST*, *SIO*, *FPL* and the proposed ARBOK with different numbers of PEs/DTs N . We can observe from Fig. 7(a) and (b) that, the average weighted QoS decreases while the system cost increases with N for all different approaches. The main reason is that, a larger N can lead to more demands for dynamic DT placement and continual update with potentially larger amount of data, thereby prompting the reduction of the average weighted QoS due to the constraint of resource. Moreover, it can be seen that the proposed ARBOK outperforms all benchmarks in enhancing the average weighted QoS and reducing the system cost. This can be explained in threefold: i) although *GST* can offer a higher average weighted QoS compared to *SIO*, it suffers from the excessive system cost because of greedily migrating DTs to the ES storing the profile that maximizes their QoS, ignoring the potential large volume of migration cost; ii) unlike *SIO* that focuses on balancing the short-term average weighted QoS and system cost, ARBOK optimizes the long-term performance for both metrics in dynamic environments by leveraging the Lyapunov optimization; and iii) compared with *FPL*, the proposed ARBOK exhibits a much stronger exploration capability, particularly for a larger N , owing to the employment of the combinatorial arm set construction which merges two arms into a single super arm, significantly reducing the number of candidate arms.

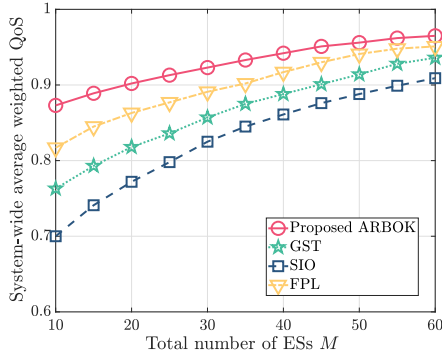
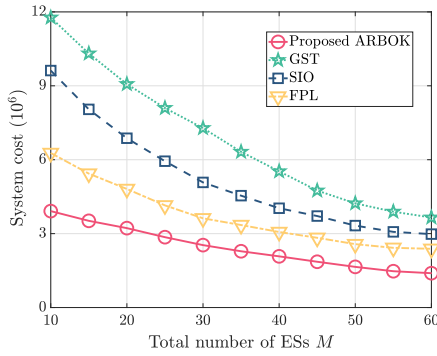
(a) System-wide average weighted QoS w.r.t. M .(b) System cost w.r.t. M .

Fig. 8. Performance comparison on different optimization approaches by changing the total number of ESs M .

Fig. 8 compares the system-wide average weighted QoS in tackling all types of service requests and the resulted system cost under *GST*, *SIO*, *FPL* and the proposed ARBOK by varying the total number of ESs M . As shown in Fig. 8(a) and (b), the average weighted QoS increases and the system cost decreases with M for all different approaches. This is because, as the total number of ESs grows, more caching resources can be provided, allowing DT profiles to be stored on closer ESs and later for effective integration in the DT update. Besides, both figures reveal that the proposed ARBOK outperforms the other benchmarks. This is due to the fact that i) different from *GST* that only maximizes the short-term QoS performance, ARBOK can well balance the long-term trade-off of average weighted QoS and system cost by dynamically adjusting the importance of these two metrics according to the deployment cost queue and the Lyapunov parameter; ii) In contrast to *SIO*, which employs a particle swarm optimization algorithm to tackle the complex non-convex optimization problem to suboptimal, the proposed ARBOK significantly reduces the overall complexity by first decoupling the problem into two subproblems, and then alternately solves them using the MO-EL and AC-BL algorithms, respectively, thereby achieving a more optimal solution; and iii) compared to *FPL* that obtains decisions by imposing a simple perturbation on the real cumulative loss, the proposed ARBOK can offer superior arm selections by calculating the sampling probability distribution of each arm based on carefully designed learning rate along with estimated cumulative loss,

with a detailed analysis of the integrated bandit learning process provided in Appendix I, available online.

Additionally, we conduct the sensitivity analysis with respect to three critical parameters in the different aspects, including the ESs' caching capacity $z_m^{\max}$ of the end-edge-cloud collaborative networking system, the hyperparameter of continual DT update ζ in the introduced DT deployment framework, and the Lyapunov control parameter V in the proposed ARBOK approach. Note that, in each experiment, we only change a single parameter while fixing all other parameters to the same settings as those in Section 5.1. The obtained results are illustrated in Appendix J, available online.

VI. CONCLUSION

In this paper, we have studied the dynamic DT placement and continual update over an end-edge-cloud collaborative networking system. Aiming to maximize the system-wide average weighted QoS in tackling all types of PEs' service requests under the system cost constraint, we have formulated an on-line optimization problem to jointly determine DT migrations, service priorities towards different service request types, and corresponding DT updating strategies for both personalized service ability improvement and versatile service ability maintenance. Considering the uncertain observations and implicit feedback with regard to various decisions, we have proposed a novel adversarial bandit learning assisted online optimization approach, namely ARBOK. Specifically, ARBOK first decomposes the long-term problem into a series of instant ones using the Lyapunov method, each of which were then decoupled into two interrelated subproblems. On top of this, ARBOK leverages the designed MO-EL and AC-BL algorithms to alternately solve different subproblems till the convergence. We have conducted the theoretical analysis and extensive simulations, not only verifying the effectiveness of the introduced framework, but also illustrating the superiority of the proposed solution over counterparts.

In our future work, we plan to extend the online learning algorithm to account for inaccurate decision feedback in the practical DT deployment process, e.g., noise, delays, or missing observations of DTs' QoS, response latency and deployment cost. Specifically, we will explore the impact of noisy decision feedback on the convergence rate and learning performance of the proposed algorithm. To mitigate such issue, we may employ the feedback filtering scheme [66] to filters out-of-range values. Besides, delayed decision feedback may degrade the algorithm's performance in dynamic environments, which necessities delay-aware arm selection mechanisms [67] to explicitly incorporate feedback delay into the exploration strategy. Furthermore, missing observations may also lead to insufficient exploration and degraded learning performance. This may require a parallel exploration strategy [68] based on the number of missing feedback associated with each arm.

REFERENCES

- [1] S. Gupta, R. S. Iyer, and S. Kumar, "Digital twin: Implementation," in *Digital Twins: Advancements in Theory, Implementation, and Applications*. Berlin, Germany: Springer, 2024, pp. 159–179.

- [2] 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.
- [3] S. D. Okegbile, J. Cai, H. Zheng, J. Chen, and C. Yi, "Differentially private federated multi-task learning framework for enhancing human-to-virtual connectivity in human digital twin," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 11, pp. 3533–3547, Nov. 2023.
- [4] A. Nechesov, I. Dorokhov, and J. Ruponen, "Virtual cities: From digital twins to autonomous AI societies," *IEEE Access*, vol. 13, pp. 13866–13903, 2025.
- [5] J. Wen, J. Kang, D. Niyato, Y. Zhang, and S. Mao, "Sustainable diffusion-based incentive mechanism for generative AI-driven digital twins in industrial cyber-physical systems," *IEEE Trans. Ind. Cyber-Phys. Syst.*, vol. 3, pp. 139–149, 2025.
- [6] Y. Yang et al., "Dynamic human digital twin deployment at the edge for task execution: A two-timescale accuracy-aware online optimization," *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 12262–12279, Dec. 2024.
- [7] C. Yi, J. Cai, and Z. Su, "A multi-user mobile computation offloading and transmission scheduling mechanism for delay-sensitive applications," *IEEE Trans. Mobile Comput.*, vol. 19, no. 1, pp. 29–43, Jan. 2020.
- [8] Y. Shi et al., "Service migration or task rerouting: A two-timescale online resource optimization for MEC," *IEEE Trans. Wireless Commun.*, vol. 23, no. 2, pp. 1503–1519, Feb. 2024.
- [9] Y. Shi, Y. Yang, C. Yi, B. Chen, and J. Cai, "Toward online reliability-enhanced microservice deployment with layer sharing in edge computing," *IEEE Internet Things J.*, vol. 11, no. 13, pp. 23370–23383, Jul. 2024.
- [10] B. Gao, Z. Zhou, F. Liu, F. Xu, and B. Li, "An online framework for joint network selection and service placement in mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 21, no. 11, pp. 3836–3851, Nov. 2022.
- [11] J. Chen, C. Yi, S. D. Okegbile, J. Cai, and X. Shen, "Networking architecture and key supporting technologies for human digital twin in personalized healthcare: A comprehensive survey," *IEEE Commun. Surv. Tutor.*, vol. 26, no. 1, pp. 706–746, First Quarter 2024.
- [12] J. Chen et al., "A revolution of personalized healthcare: Enabling human digital twin with mobile AIGC," *IEEE Netw.*, vol. 38, no. 6, pp. 234–242, Nov. 2024.
- [13] L. Jiang, H. Zheng, H. Tian, S. Xie, and Y. Zhang, "Cooperative federated learning and model update verification in blockchain-empowered digital twin edge networks," *IEEE Internet Things J.*, vol. 9, no. 13, pp. 11154–11167, Jul. 2022.
- [14] R. Chen, C. Yi, H. Zhu, W. Wu, J. Kang, and D. Niyato, "DTAS: Adaptive model splitting for dynamic digital twin update with edge-cloud collaboration," in *Proc. IEEE Trans. Mobile Comput.*, 2026, pp. 1–10.
- [15] Y. Li et al., "Budget-constrained digital twin synchronization and its application on fidelity-aware queries in edge computing," *IEEE Trans. Mobile Comput.*, vol. 24, no. 1, pp. 165–182, Jan. 2025.
- [16] N. Kshetri, "Organizational impacts of the industrial metaverse," in *Proc. 32nd Eur. Conf. Int. Telecommun. Soc. (ITS): "Realising Digital Decade Eur. Union – Easier Said Than Done?"*, Madrid, Spain, Jun. 19–20, 2023. [Online]. Available: <https://hdl.handle.net/10419/277992>
- [17] J. Chen, Y. Shi, C. Yi, H. Du, J. Kang, and D. Niyato, "Generative-AI-driven human digital twin in IoT healthcare: A comprehensive survey," *IEEE Internet Things J.*, vol. 11, no. 21, pp. 34749–34773, Nov. 2024.
- [18] L. Wang, X. Zhang, H. Su, and J. Zhu, "A comprehensive survey of continual learning: Theory, method and application," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 8, pp. 5362–5383, Aug. 2024.
- [19] Y. Chen, P. Yu, Z. Zheng, J. Shen, and M. Guo, "Modeling feature interactions for context-aware QoS prediction of IoT services," *Future Gener. Comput. Syst.*, vol. 137, pp. 173–185, 2022.
- [20] C. Wang, Q. Hu, D. Yu, and X. Cheng, "Online learning for failure-aware edge backup of service function chains with the minimum latency," *IEEE/ACM Trans. Netw.*, vol. 31, no. 6, pp. 2730–2744, Dec. 2023.
- [21] A. Molavi Kakhki et al., "Identifying traffic differentiation in mobile networks," in *Proc. ACM Internet Meas. Conf.*, 2015, pp. 239–251.
- [22] A. Ghandar, A. Ahmed, S. Zulfikar, Z. Hua, M. Hanai, and G. Theodoropoulos, "A decision support system for urban agriculture using digital twin: A case study with aquaponics," *IEEE Access*, vol. 9, pp. 35691–35708, 2021.
- [23] T. Nakao et al., "Unsupervised deep anomaly detection in chest radiographs," *J. Digit. Imag.*, vol. 34, no. 2, pp. 418–427, 2021.
- [24] Y. Zhang, L. Wang, and W. Liang, "Deep reinforcement learning for mobility-aware digital twin migrations in edge computing," *IEEE Trans. Serv. Comput.*, vol. 18, no. 2, pp. 704–717, Mar./Apr. 2025.
- [25] F. Mou et al., "Adaptive digital twin migration in vehicular edge computing and networks," *IEEE Trans. Veh. Technol.*, vol. 74, no. 3, pp. 4839–4854, Mar. 2025.
- [26] Z. Chen, W. Yi, A. Nallanathan, and J. A. Chambers, "Distributed digital twin migration in multi-tier computing systems," *IEEE J. Sel. Top. Signal Process.*, vol. 18, no. 1, pp. 109–123, Jan. 2024.
- [27] D. Chen et al., "Caching user-generated content in distributed autonomous networks via contextual bandit," *IEEE Trans. Mobile Comput.*, vol. 23, no. 8, pp. 8355–8369, Aug. 2024.
- [28] B. Chen, J. Lee, and J. Honda, "Geometric resampling in nearly linear time for follow-the-perturbed-leader with best-of-both-worlds guarantee in bandit problems," in *Proc. Int. Conf. Mach. Learn.*, 2025, pp. 8403–8426.
- [29] H. Cao and J. Cai, "Distributed multiuser computation offloading for cloudlet-based mobile cloud computing: A game-theoretic machine learning approach," *IEEE Trans. Veh. Technol.*, vol. 67, no. 1, pp. 752–764, Jan. 2018.
- [30] F. Simon and J.-C. Usunier, "Cognitive, demographic, and situational determinants of service customer preference for personnel-in-contact over self-service technology," *Int. J. Res. Mark.*, vol. 24, no. 2, pp. 163–173, 2007.
- [31] M. Adhikari, M. Mukherjee, and S.N. Srirama, "DPTO: A deadline and priority-aware task offloading in fog computing framework leveraging multilevel feedback queueing," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 5773–5782, Jul. 2020.
- [32] A. Alnoman and A. Anpalagan, "A dynamic priority service provision scheme for delay-sensitive applications in fog computing," in *Proc. IEEE 29th Biennial Symp. Commun.*, 2018, pp. 1–5.
- [33] B. Ferwerda, M. Tkalcic, and M. Schedl, "Personality traits and music genre preferences: How music taste varies over age groups," in *Proc. ACM Temporal Reasoning Recommender Syst.*, 2017, pp. 16–20.
- [34] D. Goldfarb, I. Evron, N. Weinberger, D. Soudry, and P. Hand, "The joint effect of task similarity and overparameterization on catastrophic forgetting—an analytical model," 2024, *arXiv:2401.12617*.
- [35] J. Yoo, Y. Liu, F. Wood, and G. Pleiss, "Layerwise proximal replay: A proximal point method for online continual learning," 2024, *arXiv:2402.09542*.
- [36] 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.
- [37] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, 1948.
- [38] A. Goldsmith, *Wireless Communications*. Cambridge, U.K.: Cambridge Univ. Press, 2005.
- [39] J. Cai, X. Shen, J. W. Mark, and A. S. Alfa, "Semi-distributed user relaying algorithm for amplify-and-forward wireless relay networks," *IEEE Trans. Wireless Commun.*, vol. 7, no. 4, pp. 1348–1357, Apr. 2008.
- [40] C. Yi et al., "A queueing game based management framework for fog computing with strategic computing speed control," *IEEE Trans. Mobile Comput.*, vol. 21, no. 5, pp. 1537–1551, May 2022.
- [41] S. Dohare, J. F. Hernandez-Garcia, Q. Lan, P. Rahman, A. R. Mahmood, and R. S. Sutton, "Loss of plasticity in deep continual learning," *Nature*, vol. 632, no. 8026, pp. 768–774, 2024.
- [42] M. Li, S. Kudugunta, and L. Zettlemoyer, "(MIS) fitting: A survey of scaling laws," in *Proc. Int. Conf. Learn. Representations*, 2025.
- [43] C. Li, L. Zheng, X. Ji, X. Liao, Z. Wang, and G. Mao, "6G-DFTP: A digital-twin-enabled privacy-preserving federated user prediction framework for 6G mobile edge computing," *IEEE Commun. Mag.*, vol. 63, no. 8, pp. 100–106, Aug. 2025.
- [44] L. Yang, L. Wang, Z. Zheng, and Z. Zhang, "A continual learning-based framework for developing a single wind turbine cyber-twin adaptively serving multiple modeling tasks," *IEEE Trans. Ind. Inform.*, vol. 18, no. 7, pp. 4912–4921, Jul. 2022.
- [45] G. M. Van de Ven, T. Tuytelaars, and A. S. Tolias, "Three types of incremental learning," *Nature Mach. Intell.*, vol. 4, no. 12, pp. 1185–1197, 2022.
- [46] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 3987–3995.
- [47] J. Rissanen, "Fisher information and stochastic complexity," *IEEE Trans. Inf. Theory*, vol. 42, no. 1, pp. 40–47, Jan. 1996.
- [48] D. Pisinger, "An exact algorithm for large multiple knapsack problems," *Eur. J. Oper. Res.*, vol. 114, no. 3, pp. 528–541, 1999.
- [49] M. Neely, *Stochastic Network Optimization With Application to Communication and Queueing Systems*. San Rafael, CA, USA: Morgan & Claypool, 2010.

- [50] L. Georgiadis et al., "Resource allocation and cross-layer control in wireless networks," *Found. Trends Netw.*, vol. 1, no. 1, pp. 1–144, 2006.
- [51] P. M. Castro, "Tightening piecewise mccormick relaxations for bilinear problems," *Comput. Chem. Eng.*, vol. 72, pp. 300–311, 2015.
- [52] S. P. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
- [53] C. Bliclú, P. Bonami, and A. Lodi, "Solving mixed-integer quadratic programming problems with IBM-CPLEX: A progress report," in *Proc. 26th RAMP Symp.*, 2014, pp. 16–17.
- [54] Z. Huang et al., "AoI-guaranteed bandit: Information gathering over unreliable channels," *IEEE Trans. Mobile Comput.*, vol. 23, no. 10, pp. 9469–9486, Oct. 2024.
- [55] T. Lattimore and C. Szepesvári, *Bandit Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2020.
- [56] T. Kocák, G. Neu, M. Valko, and R. Munos, "Efficient learning by implicit exploration in bandit problems with side observations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 613–621.
- [57] W. Takahashi and M. Toyoda, "Weak convergence theorems for nonexpansive mappings and monotone mappings," *J. Optim. Theory Appl.*, vol. 118, no. 2, pp. 417–428, 2003.
- [58] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*. Berlin, Germany: Springer, 1996, pp. 153–181.
- [59] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from GPS trajectories," in *Proc. Int. Conf. World Wide Web*, 2009, pp. 791–800.
- [60] B. Xiang, J. Elias, F. Martignon, and E. Di Nitto, "A dataset for mobile edge computing network topologies," *Data Brief*, vol. 39, 2021, Art. no. 107557.
- [61] Y. Zhu et al., "Rethinking query-based transformer for continual image segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2025, pp. 4595–4606.
- [62] J. Ma, Y. He, F. Li, L. Han, C. You, and B. Wang, "Segment anything in medical images," *Nature Commun.*, vol. 15, no. 1, 2024, Art. no. 654.
- [63] A. Sellergrén et al., "Medgemma technical report," 2025, *arXiv:2507.05201*.
- [64] 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.
- [65] B. Bandyopadhyay, P. Kuila, and M. Chandra Govil, "Popularity-conscious service caching and offloading in digital twin and NOMA-aided connected autonomous vehicular systems," *IEEE Trans. Netw. Serv. Manage.*, vol. 21, no. 6, pp. 6451–6464, Dec. 2024.
- [66] S. Bubeck, N. Cesa-Bianchi, and G. Lugosi, "Bandits with heavy tail," *IEEE Trans. Inf. Theory*, vol. 59, no. 11, pp. 7711–7717, Nov. 2013.
- [67] S. Masoudian, J. Zimmert, and Y. Seldin, "A best-of-both-worlds algorithm for bandits with delayed feedback," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2022, pp. 11752–11762.
- [68] Y. Wu, A. Gyorgy, and C. Szepesvári, "On identifying good options under combinatorially structured feedback in finite noisy environments," in *Proc. ACM Int. Conf. Mach. Learn.*, PMLR, 2015, pp. 1283–1291.



Yuze Yang received the BS and MS degrees in computer science and technology from the School of Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing, China, in 2022 and 2025, respectively. He is currently working toward the PhD degree with the College of Computer Science and Technology, NUAA. His main research interests include mobile edge computing, digital twin, edge intelligence, and online optimization.



Changyan Yi (Senior Member, IEEE) received the PhD degree in electrical and computer engineering from the University of Manitoba, MB, Canada, in 2018. He is currently a professor with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing, China. His research interests include stochastic optimization, game theory, incentive mechanism, queueing scheduling and artificial intelligence, with applications in resource management and decision making for wireless communication and

mobile computing, industrial cyber-physical system and intelligent control, edge intelligence, and digital twin network.



Haifeng Zhu received the BSc degree from Chongqing University and the MS degree from Zhejiang University. He is currently working toward the PhD degree with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China. He is a senior engineer with the Servo Technology Institute of China Aerospace Science and Industry Corporation. His research interests include industrial cyber-physical system and intelligent control.



Shimin Gong (Member, IEEE) received the BE and ME degrees in electronics and information engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2008 and 2012, respectively, and the PhD degree in computer engineering from Nanyang Technological University, Singapore, in 2014. He is currently a professor with the School of Intelligent Systems Engineering, Sun Yat-sen University, Shenzhen, China. His research interests include wireless powered communications, backscatter communications, and machine learning in wireless communications. He was the co-recipient of IEEE WCNC 2019 Best Paper Award on MAC and Cross-layer Design, and 2023 IEEE Communications Society Best Survey Paper Award. He is an associate editor for *IEEE Transactions on Vehicular Technology*.



Jiawen Kang (Senior Member, IEEE) received the MS and PhD degrees from the Guangdong University of Technology, China, in 2015 and 2018, respectively. He is currently a full professor with the Guangdong University of Technology. He was a post-doctoral researcher with Nanyang Technological University, Singapore, from 2018 to 2021. His research interests mainly focus on blockchain, metaverse, and AIGC in wireless communications and networking.



Dusit Niyato (Fellow, IEEE) received the BEng degree from the King Mongkut's Institute of Technology Ladkrabang (KMUTL), Thailand, and the PhD degree in electrical and computer engineering from the University of Manitoba, Canada. He is currently a professor with the College of Computing and Data Science, Nanyang Technological University, Singapore. His research interests include mobile generative AI, edge general intelligence, quantum computing and networking, and incentive mechanism design.