Deep Reinforcement Learning for Edge Service Placement in Softwarized Industrial Cyber-Physical System

Yixue Hao, Min Chen, *Fellow, IEEE*, Hamid Gharavi, Yin Zhang, and Kai Hwang

***Abstract*—Future industrial cyber-physical system (CPS) devices are expected to request a large amount of delay-sensitive services that need to be processed at the edge of a network. Due to limited resources, service placement at the edge of the cloud has attracted significant attention. Although there are many methods of design schemes, the service placement problem in industrial CPS has not been well studied. Furthermore, none of existing schemes can optimize service placement, workload scheduling, and resource allocation under uncertain service demands. To address these issues, we first formulate a joint optimization problem of service placement, workload scheduling, and resource allocation in order to minimize service response delay. We then propose an improved deep Q-network (DQN)-based service placement algorithm. The proposed algorithm can achieve an optimal resource allocation by means of convex optimization where the service placement and workload scheduling decisions are assisted by means of DQN technology. The experimental results verify that the proposed algorithm, compared with existing algorithms, can reduce the average service response time by 8–10%.**

***Index Terms*—Deep reinforcement learning, edge cloud, industrial cyber-physical system (CPS), service placement.**

Manuscript received June 19, 2020; revised September 3, 2020 and November 2, 2020; accepted November 18, 2020. Date of publication December 2, 2020; date of current version May 3, 2021. This work was supported by the National Key R&D Program of China under Grant 2018YFC1314600, Nature Science Foundation of China under Grant 61802138, Shenzhen Institute of Artificial Intelligence and Robotics for Society, and in collaboration with the Advanced Network Technology Division (ANTD) of the National Institute of Standards and Technology (NIST), USA. Paper no. TII-20-2987. *(Corresponding author: Min Chen.)*

Yixue Hao is with the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430 074, China (e-mail: yixuehao@hust.edu.cn).

Min Chen is with the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430 074, China, and also with the Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen 518 172, China (e-mail: minchen2012@hust.edu.cn).

Hamid Gharavi is with the National Institute of Standards and Technology (NIST), Gaithersburg, MD 20 899-8920 USA (e-mail: hamid.gharavi@nist.gov).

Yin Zhang is with the School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu 610054, China (e-mail: zhangyin123@uestc.edu.cn).

Kai Hwang is with the Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen 518 172, China, and also with the School of Data Science (SDS), The Chinese University of Hong Kong, Shenzhen (CUHK-SZ), Shenzhen 518 172, China (e-mail: hwangkai@cuhk.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TII.2020.3041713.

Digital Object Identifier 10.1109/TII.2020.3041713

I. Introduction

THE industrial cyber-physical system (CPS) technology promises to ensure seamless connectivity of industrial physical and cyber worlds [1]. With the growing number of wireless sensor devices, CPS will continue to play a crucial role in the integration of complex sensing, including delay sensitive and computationally intensive services such as real-time communication monitoring and control. As data generated by wireless sensor devices continue to expand, offloading them to the cloud for processing may not only causes a long delay but can also lead to network congestion [2].

To address this, it is essential to enhance the computing capability at the edge of network, where servers are deployed close to sources. Using edge cloud, not only rapid processing and analyzing for sensor data can be realized, but the load of the backhaul link can also be reduced [3], [4]. Furthermore, security and privacy requirements of the industrial CPS are also important that need to be addressed. For example, data generated by wireless devices are private and should be handled with a high degree of confidentiality at the edge of the network [5].

In practice, however, due to limited computing and storage resources that would require processing a massive amount of data, the entire operation of the industrial CPS at the edge cloud requires the adoption of an efficient service placement [6], [7]. To address these challenges, Chen *et al.* [8] proposed fog configuration to minimize delay and energy consumption in the industrial Internet of Things. Wang *et al.* [9] designed a deployment of service entity with the lowest energy consumption. Zhang *et al.* [10] proposed an efficient service placement scheduling using distributed clouds. However, none of these works take into account the impact caused by heterogeneity in service requests among multiple edge clouds (i.e., service requests on edge clouds are unbalanced in spatial and temporal).

In view of the imbalances in service requests and to reduce service delay, a joint optimization of service placement and workload scheduling problems has been investigated by a number of researchers. For instance, Poularakis *et al.* [11] designed a service placement and request scheduling scheme in cases where there are insufficient communication, computing, and storage resources on the edge cloud. With regards to delay-sensitive tasks, Ma *et al.* [12] proposed cooperative service placement and workload scheduling scheme to minimize the service response time and overall outsourcing traffic to the cloud. Farhadi

et al. [13] proposed a submodular-based optimization scheme for service placement and request scheduling for data-intensive applications in edge clouds. In all these joint optimization schemes, it is assumed that the service demand is known, i.e., uncertain service demand has not been taken into consideration in these joint optimization schemes.

In this article, we focus on the service placement and workload scheduling problem in industrial CPS. We propose a deep Q-network (DQN)-based algorithm to tackle the uncertain service demands of the edge cloud. This is indeed a challenging problem [14], because when a service placement is carried out, the service request at the time is unknown, which will lead to unreasonable placement and consequently cause greater latency in the service requests of wireless sensor devices. In addition to service placement and workload scheduling, we also need to consider the allocation of computing resources as the edge cloud needs to decide how much computing resources should be allocated to execute the service. Thus, for the first time, we analyze the joint optimization problem of service placement, workload scheduling, and resource allocation under the demand uncertainties.

To achieve this, we first design a softwarized-based industrial CPS, and realize information exchange between edge clouds through centralized software-defined network (SDN) [15] controllers. Then, we formulate the joint optimization problem of service placement, workload scheduling, and resource allocation to minimize service response time. Since this problem is a nonlinear mixture of integer optimization, solving it is a difficult task. Thus, we propose an improved DQN-based service placement (DSP) algorithm. Although we have adopted a DQN-based approach to solve this challenging problem, we should point out traditional DQN network is not suitable for the edge node due to the high computational complexity caused by too many states and action spaces. Thus, in our approach we propose to reduce the computational complexity by setting offline training of the Q-network and online decision-making for service placement and workload scheduling. In addition, we evaluate the convergence of the DSP algorithm.

In summary, the main contributions of this article include the following.

1) *Service Placement in the Industrial CPS:* To find out which services should be placed on which edge node to minimize service response time, we propose a joint optimization approach that incorporates service placement, workload scheduling, and resource allocation under service demand uncertainty.
2) *DQN-based Service Placement Algorithm:* To solve the optimization problem, we divide it into two subproblems: a) Resource allocation of each edge cloud and b) service placement and workload scheduling. For the former, we adopt convex optimization to give the optimal allocation scheme, whereas in the latter, we adopt DQN to solve the problem through learning service requests.
3) *Performance Evaluation:* We conduct comprehensive simulation experiments to verify the effectiveness of the proposed DSP algorithm. The experiment results show that, compared with traditional edge service placement schemes, the DSP strategy can minimize latency during service acquisition. For instance, in a scenario of 30 services, compared to other algorithms, our propose algorithm can reduce the average latency by 8%–10%

The remainder of the article is organized as follows. In Section II, we give a review of related work and present the system model and problem formulation in Section III. In Section IV, we present our algorithm to solve the problem. The simulation results and discussions are given in Section V. Finally, Section VI concludes this article.

II. Related Work

The industrial CPS generally includes a large number of wireless sensor devices, which can generate and collect a large amount of data. However, offloading them to a remote cloud for further processing may lead to long delays [16]–[18]. Currently, edge computing can provide high-reliability, high-bandwidth, and low-delay computing services for sensor devices by deploying servers at network edges (including wireless access points, base stations, routers, etc.) [3], [19]. This is because the edge server is closer to the mobile device, which can be directly connected to edge cloud through a wireless network, hence greatly reducing communication latency.

Currently, most investigations have concentrated on how to offload data generated by wireless sensor devices onto the edge cloud. This not only reduces the data processing delay but also saves the energy consumption of wireless sensor devices [14], [20]. For the edge cloud, according to prior knowledge of where there is global information in the system, existing task offloading schemes can be divided into centralized task offloading [18] and the distributed task offloading algorithm [14], [16]. These works show that the delay and energy consumption can be reduced by designing a reasonable task offloading strategy. However, these works assume that all services requested by wireless sensor devices can be handled by the edge cloud.

In reality, however, the edge cloud has limited storage and computing capacity and cannot support the operation of all types of services. Thus, which services are to be placed on the edge cloud (i.e., service placement) should be taken into consideration [6]. To address this challenge, some existing works have proposed service placement algorithms. For delay-sensitive services, such as augmented reality (AR), many different service placement strategies are proposed by researchers to reduce the delay. For example, cost-aware service placement has been proposed in [19]. Since different edge clouds have different service requests, service placement and workload scheduling can be jointly optimized to further improve network performance.

In [11] and [21], a joint optimization problem for service placement and request routing is designed using submodular optimization. These joint optimization schemes are based on prior knowledge with service requests and without incorporating resource allocation in order to further improve utilization of resources. Thus, in contrast with existing works, we study the optimization of service placement, workload scheduling and resource allocation in the softwarized-based industrial CPS under demand uncertainty, aiming to minimize service response time.

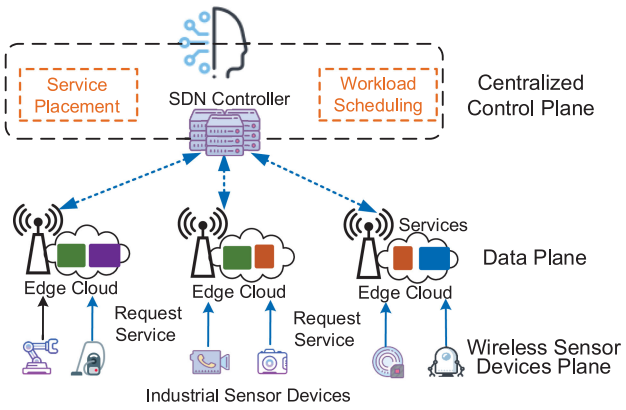

Fig. 1. System architecture of softwarized-based industrial CPS.

III. System Model and Problem Formulation

In this section, we present the system model and problem formulation. Our goal is to minimize the delay when wireless devices in industrial CPS obtain services.

A. System Architecture

In our investigation, we consider an industrial CPS ecosystem scenario, which includes a mass of wireless sensors devices, multiple edge clouds, and a remote cloud. In this scenario, wireless sensor devices communicate with edge clouds through wireless channels, while edge clouds are connected to the remote cloud through wired link. Moreover, we adopt SDN in order to realize the distributed management of the edge cloud according to the system architecture shown in Fig. 1.

Specifically, the softwarized-based industrial CPS includes the wireless sensors devices plane, the data plane, and the centralized control plane. The devices plane is composed of wireless sensor devices in industrial CPS, which is responsible for data collection. The data plane consists of access points and their corresponding edge clouds, and is responsible for processing the data services collected by the sensors. The interconnection among multiple edge clouds is based on the centralized control plane. Moreover, the centralized control plane can realize service placement and workload scheduling on edge clouds. Thus, with the SDN technology, a centralized control of industrial CPS on distributed edge clouds can be realized.

Let us consider a softwarized industrial CPS system that consists of N edge clouds where each can provide data analysis and processing services for wireless sensor devices. We denote a set of edge clouds by $\mathcal{N} = \{1, 2, \ldots, N\}$. Also, let F_i^{edge} and M_i^{edge} be the computation capacity and storage capacity of edge cloud i, respectively. Compared with limited storage and computing resources on the edge cloud, as in [19], we consider the remote cloud has sufficient computing and storage capacities and has all the service requests in industrial CPS.

Moreover, we consider the service placement operates in discrete time slots $\mathcal{T} = \{1, 2, \ldots, T\}$, where T denotes the finite time horizon and each time slot has a duration. In each duration, the SDN controller needs to place services, schedule workload, and to do resource allocation. In the following, we present our proposed service placement, workload scheduling, and resource allocation model. For the sake of clarity, the main notations used in this article are shown in Table I.

TABLE I
Summary Table of Importation Notations

Notation	Meaning
$\mathcal{N}$	set of edge clouds.
N	number of edge clouds.
$\mathcal{K}$	set of services.
K	number of services.
ω_k	computing resource for processing service k.
s_k	storage capacity required to place service k.
$x_{i,k}(t)$	indicator of whether the service k is placed on the edge cloud i or not at time slot t.
$y_{i,k}(t)$	fraction of computing capacity for service k allocated by edge cloud i at time slot t .
$p_{i,k}(t)$	workload ratio where service k is operated on edge cloud i at time slot t.
$\lambda_{i,k}^t$	the number of request for service k on edge cloud i at time slot t.
F_i^{edge}	the computing capacity of edge cloud i.
M_i^{edge}	the storage capacity of edge cloud i.
$r_{edge}(t)$	the transmission rate between edge clouds.
$r_k(t)$	the transmission rate of core network when service k is transmitted.

B. Service Placement

Since industrial wireless node may collect wide ranging data, different data processing and analysis should be carried out for different services. We assume that there are K services in industrial CPS, indexed by $\mathcal{K} = \{1, 2, \ldots, K\}$. Moreover, the edge cloud consumes computing and storage resources to handle requests of wireless sensor devices. As in [11], we assume that ω_k (in CPU cycle) is the computing resource required for processing service k, and s_k (in bits) is the storage capacity required to place service k. Considering that an edge cloud has limited computing and storage resources, only finite services can be placed upon it. So, the issue is to find the best possible service to utilize the edge cloud resources. To address this challenge, we analyze service placement, workload scheduling, and resource allocation problem in edge clouds.

First, for service placement problem on edge cloud, let binary variable $x_{i,k}(t) \in \{0, 1\}$ denote whether service k is placed on the edge cloud i at time slot t. We set $x_{i,k}(t) = 1$ (or 0) if the service is placed on the edge cloud i (or not). Accordingly, we define $\mathcal{X}(t) = \{x_{i,k}(t) | i \in \mathcal{N}, k \in \mathcal{K}\}$ to represent the service placement decision. Furthermore, we assume that the storage capacity on the edge cloud is limited, thus, in any time slot t, the size of services placed on the edge cloud i cannot be larger than the storage capacity of the edge cloud, i.e.,

$$\sum_{k=1}^{K} x_{i,k}(t) s_k \leq M_i^{\text{edge}}, \quad \forall t. \tag{1}$$

As for the resource allocation problem of the edge cloud and in consideration of heterogeneity in computing resources on it,

we know that the computing capacity of edge cloud i is F_i^{edge}. Let denote $y_{i,k}(t) \in [0,1]$ as the fraction of computing capacity F_i^{edge} allocated to the service k. When $y_{i,k}(t)$ is equal to 0, it means that the service k is not placed on the edge cloud, so the computing resources allocated at this time are 0. Accordingly, we define $\mathcal{Y}(t) = \{y_{i,k}(t) | i \in \mathcal{N}, k \in \mathcal{K}\}$ as all resource allocation decisions.

Since the computing capacity on the edge cloud is limited, the size of computing resources allocated for service k cannot exceed its maximum computing resources, i.e.,

$$\sum_{k=1}^{K} y_{i,k}(t) \leq 1, \quad \forall t. \tag{2}$$

In addition to edge cloud, the computing capacity of remote cloud is much larger than that of the edge cloud.

C. Service Response Time

To analyze service response time, we assume that at time slot t, the number of requests for service k on edge cloud i by wireless sensor devices are $\lambda_{i,k}(t)$. Due to the heterogeneity of data collected by wireless sensor devices and the dynamic nature of the service requested, $\lambda_{i,k}(t)$ changes dynamically. It should be noted that when the requested service is not placed on this edge cloud, the service can be operated on a cloud or on another edge cloud if the service is placed through scheduling. Therefore, we give the workload scheduling scheme for services. Let $p_{i,k}(t) \in [0,1]$ denote the workload ratio, where service k is operated on edge cloud i. Accordingly, we can define $\mathcal{P}(t) = \{p_{i,k}(t) | i \in \mathcal{N} \cup \{o\}, k \in \mathcal{K}\}$ as all service scheduling decisions, and $p_{o,k}(t)$ represents the proportion where service k is operated on the cloud. We assume that each requested service needs to be executed, i.e.,

$$\sum_{i \in \mathcal{N} \cup \{o\}} p_{i,k}(t) = 1, \quad \forall t. \tag{3}$$

Moreover, the total number of requests for service k in industrial CPS at time slot t can be obtained as: $\lambda_k = \sum_{i=1}^{N} \lambda_{i,k}(t)$. Thus, we can obtain the overall computation workload $W_{i,k}(t)$ and data size $S_{i,k}(t)$ required when service k is operated on edge cloud i at time slot t as

$$W_{i,k}(t) = p_{i,k} \omega_k \lambda_k(t)$$
$$S_{i,k}(t) = p_{i,k} s_k \lambda_k(t).$$

We note if $W_{i,k}(t) \leq \omega_k \lambda_{i,k}(t)$, the computation workload required by services k is from edge cloud i; otherwise, the excessive workload $(W_{i,k}(t) - \omega_k \lambda_{i,k}(t))$ corresponds to the nearby edge clouds.

Next, we analyze the specific service response time and reduce the outsourcing service to remote cloud. For instance, at the point where $x_{i,k}(t) = 1$, the edge cloud i should be able to handle service request k. Under these conditions, we first obtain the computing delay when service k is handled on edge cloud i as

$$D_{i,k}^{\text{comp}}(t) = x_{i,k}(t) \frac{W_{i,k}(t)}{y_{i,k}(t) F_i^{\text{edge}}}. \tag{4}$$

In terms of transmission delay, since wireless sensor devices are often very close to the access point, therefore, as in [11], we ignore the transmission delay between these devices and edge cloud. In addition, considering that neighboring edge clouds are not far away from each other (similar to [13]), by assuming that the transmission rate is $r_{\text{edge}}(t)$, the transmission delay of the edge cloud can be obtained as

$$D_{i,k}^{\text{tran}}(t) = x_{i,k}(t) \frac{\max\{(p_{i,k} s_k \lambda_k(t) - s_k \lambda_{i,k}(t)), 0\}}{r_{\text{edge}}(t)}. \tag{5}$$

In the case where $x_{i,k}(t) = 0$, outsourcing some services to a remote cloud would be required. Based on [12], the service response time is mainly from the transmission delay in the core network. So, we assume that the rate of the core network when service k is transmitted at time slot t is $r_k(t)$. Under these conditions, the processing delay of service k on the cloud can be obtained as

$$D_{c,k}(t) = \sum_{i=1}^{N} \left[(1 - x_{i,k}(t)) \frac{p_{0,k}(t) \lambda_k(t) s_k}{r_k(t)} \right]. \tag{6}$$

Thus, we can obtain the average response time of service k as the weighting of all edge clouds to computation delay and transmission delay, as follows:

$$d_k(t) = \sum_{i=1}^{N} (D_{i,k}^{\text{comp}}(t) + D_{i,k}^{\text{trans}}(t)) + D_{c,k}(t). \tag{7}$$

Our goal is to reduce the outsourcing service to remote cloud while minimizing the service response time, as follows: $D_k(t) = d_k(t) + p_{0,k}(t) \lambda_k(t) s_k$.

D. Uncertain Service Demand

Existing service placement strategies are based on the assumption that service requests on the edge cloud are known or subject to a specific distribution (e.g., Poisson distribution). More specifically, the service provider is aware of requested services when making a service placement. However, in practice, for industrial CPS each edge cloud covers different regions where system environment changes dynamically. This situation causes uncertainty in requesting services in each region. Under these conditions, we need to design the algorithm that can deal with time changing environments (such as the arrival of a new service demand) and find an optimal solution in which the service placement, resource allocation, and workload scheduling under uncertain service demands can be efficiently handled. Thus, we first use the idea of software definition to realize the centralized control of distributed edge cloud. Then, we utilize deep reinforcement learning to learn the service request.

E. Problem Formulation

In softwarized-based industrial CPS, our goal is to minimize $D_k(t)$ through service placement, resource allocation, and workload scheduling under uncertain service demands. Thus, the

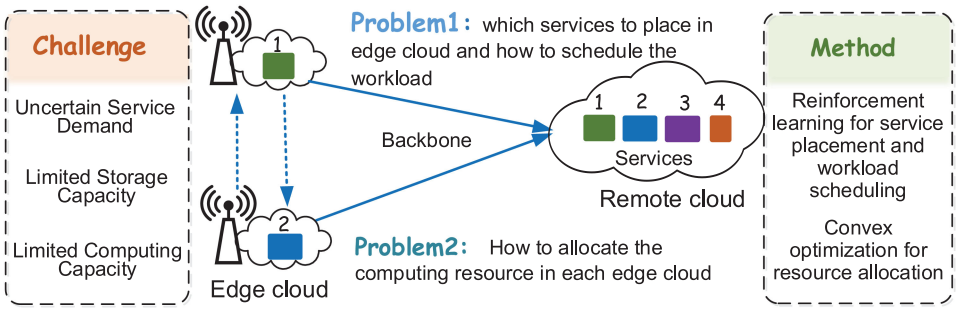

Fig. 2. Illustration of DQN-based service placement method.

problem can be expressed as

$$\mathbf{P1}: \min_{\mathcal{X},\mathcal{Y},\mathcal{P}} \frac{1}{T}\sum_{t=1}^{T}\sum_{k=1}^{K} D_k(t) \tag{8}$$

$$\text{s.t.} \quad C1: \sum_{k=1}^{K} x_{i,k}(t)s_k \le M_i^{edge}, \quad \forall t \tag{9}$$

$$C2: \sum_{k=1}^{K} y_{i,k}(t) \le 1, \quad \forall t \tag{10}$$

$$C3: \sum_{i\in\mathcal{N}\cup\{o\}} p_{i,k}(t) = 1, \quad \forall t \tag{11}$$

$$C4: x_{i,k}(t) \in \{0,1\}, k \in \mathcal{K}, i \in \mathcal{N} \tag{12}$$

$$C5: y_{i,k}(t) \in [0,1], k \in \mathcal{K}, i \in \mathcal{N} \tag{13}$$

$$C6: p_{i,k}(t) \in [0,1], k \in \mathcal{K}, i \in \mathcal{N}. \tag{14}$$

In the above, the objective function (12) aims to compute the minimal service response time and reduce the outsourcing service to remote cloud. The first constraint condition (C1) indicates that placed services cannot exceed the storage capacity of the edge cloud. Constraint (C2) ensures that resource allocation on the edge cloud cannot exceed the computing capacity of the edge cloud. Finally, the value of variables are given in constraint (C3), (C4), and (C5).

For the optimization problem $\mathbf{P1}$, we note that $\mathcal{X}$ is an integer variable, $\mathcal{Y}$ and $\mathcal{P}$ are continuous variables, and the objective function is nonlinear. Thus, the optimization problem $\mathbf{P1}$ is a mixed integer nonlinear programming, which is NP-hard problem. In order to address this, we adopt DQN for solving the problem where an optimal scheme will be can be developed through a learning service demand.

IV. DQN-Based Service Placement Algorithm

Fig. 2 illustrates our proposed DQN-based service placement method. Specifically, we adopt convex optimization and DQN to solve the optimization problem $\mathbf{P1}$. Considering that resource allocation is processed on the edge node, whereas the SDN controller handles the service placement and workload scheduling, problem $\mathbf{P1}$ can be divided into two subproblems: 1) resource allocation and 2) service placement and workload scheduling.

A. Resource Allocation Problem

First, we assume that the optimal service placement $\mathcal{X}^*(t)$ and workload scheduling $\mathcal{P}^*(t)$ at time slot t have been handled by a centralized SDN controller. Then the optimization $\mathbf{P1}$ is converted into the following optimization problem $\mathbf{P2}$ with respect to $y_{i,k}$:

$$\mathbf{P2}: \min_{\mathcal{Y}} \frac{1}{T}\sum_{t=1}^{T}\sum_{k=1}^{K} D_k(t) \tag{15}$$

$$\text{s.t.} \quad C1: \sum_{k=1}^{K} y_{i,k}(t) \le 1, \quad \forall t \tag{16}$$

$$C2: y_{i,k}(t) \in [0,1], k \in \mathcal{K}, i \in \mathcal{N}. \tag{17}$$

By analyzing $D_k(t)$, we conclude that given the optimal $x_{i,k}^*(t)$ and $p_{i,k}^*(t)$, there is only $D_{i,k}^{\text{comp}}(t)$ for the service response time that relates to edge cloud i. As resource allocation should be conducted at each edge cloud i within a time slot t and in a distributed manner, the objective function can be converted into

$$f(\mathcal{Y}_i(t)) = \sum_{k=1}^{K} D_{i,k}^{\text{comp}}(t). \tag{18}$$

Therefore, we can obtain that the optimal resource allocation scheme as shown in the following lemma:

Lemma 1: When the optimal service placement $x_{i,k}^*(t)$ and workload scheduling $p_{i,k}^*(t)$ are given, the optimal resource allocation $y_{i,k}^*(t)$ can be obtained as

$$y_{i,k}^*(t) = \frac{\sqrt{\mu_{i,k}(t)}}{\sum_{i=1}^{N}\sqrt{\mu_{i,k}(t)}} \tag{19}$$

where $\mu_{i,k}^t = (x_{i,k}^*(t)W_{i,k}(t))/F_i^{\text{edge}}$.

Proof: First, we can obtain the second-order derivative of $f(\mathcal{Y}_i(t))$ as

$$\frac{\partial^2 f}{\partial y_{i,k}(t)^2} = \frac{2\mu_{i,k}(t)}{y_{i,k}(t)^3} \ge 0. \tag{20}$$

Considering that if a service is not placed on the edge cloud, $y_{i,k}$ is equal to 0. So we can conclude that the Hessian matrix of $f(\mathcal{Y}_i(t))$ is positive and $f(\mathcal{Y}_i(t))$ with respect to $y_{i,k}$ is convex. Based on the Karush–Kuhn–Tucker (KKT) condition adopted

in [22], the optimal resource allocation scheme $y_{i,k}(t)$ can then be obtained. ■

B. Service Placement and Workload Scheduling Problem

According to the above analysis, we conclude that given the service placement and workload scheduling scheme, optimal resource allocation can be handled by each edge cloud i at time slot t. Next, we introduce a solution for the service placement and workload scheduling problem under uncertain service demands. We design a service placement and workload scheduling algorithm based on DQN [23], [24].

Specifically, DQN modeling is often conducted based on the Markov decision processes (MDP), which includes intelligent agent, environment, state, action, and reward elements. The basic idea is as follows: An intelligent agent observes the environment at time slot t to obtain the current state. Then, the agent makes a corresponding action as an observed state and reward will be given after the action is accepted by the environment before proceeding to the next state. An intelligent agent finally maximizes the sum of its rewards through continuous interactions with the environment.

Thus, we first need to describe the service placement and workload scheduling optimization problem as an MDP. Bear in mind that MDP consists of three components: State space $\mathcal{S}$, action space $\mathcal{A}$, and reward function $\mathcal{R}$. In order to apply this to the edge computing scenario, we need to specifically design different components of reinforcement learning as described below.

State Space: The SDN controller is responsible for observing the state of each edge cloud, which includes service demand and the information of the edge cloud. Specifically, the state of edge cloud i at time slot t can be obtained as

$$s_i(t) = \{\lambda_{i,k}(t), M_i^{\text{edge}}, F_i^{\text{edge}}, y_{i,k}^*(t)\}. \tag{21}$$

Thus, the state of edge cloud i can be obtained by the SDN controller through local observation at the beginning of each time slot t.

Action Space: We begin with service placement $x_{i,k}(t)$, by assuming that there are N edge clouds, K services, and $x_{i,k}(t) \in \{0, 1\}$. Thus, the action space of the service placement is $2^{K\times N}$. Then, we consider the workload scheduling scheme. From the optimization problem $\mathbf{P1}$, we can observe that the workload scheduling scheme $p_{i,k}(t)$ is continuous. If a modeling is conducted with usual continuous action space, the action space would be too large, which is not suitable for deployment on the edge node. To address this issue, we consider that, in practice, workload scheduling is usually done with blocks. Thus, we assumed that the minimum processing unit is $\Delta p_{i,k}(t)$. Therefore, the value of the workload scheduling scheme can be obtained as

$$p_{i,k}(t) \in \{\Delta p_{i,k}(t), \ldots, m\Delta p_{i,k}(t), \ldots, 1\}. \tag{22}$$

Thus, we can obtain the action space $\mathcal{A}(t)$ of edge cloud i at time slot t as

$$a_i(t) = \{x_{i,k}(t), m\Delta p_{i,k}(t), k \in \mathcal{K}\}. \tag{23}$$

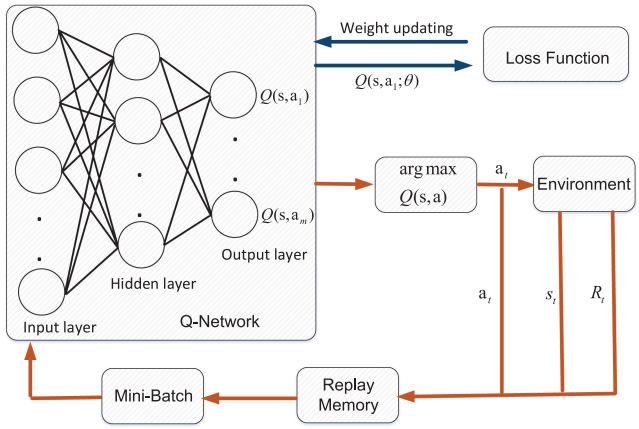

Fig. 3. Deep reinforcement learning based service placement and workload scheduling scheme.

Reward: Bear in mind that our goal is to minimize service response time through service placement, workload scheduling, and resource allocation. Thus, we can then obtain the reward function as

$$R(t) = \sum_{k=1}^{K} D_k(t). \tag{24}$$

To minimize the reward function, we first define $Q(s, a)$ as the action value. We can then show the expected total sum of future rewards for T time steps, as

$$Q(s, a) = \mathbb{E}\left(\sum_{t=1}^{T} \gamma^t R_t | s_t = s, a_t = a\right) \tag{25}$$

where $\gamma \in [0, 1]$ is the discount factor and $\mathbb{E}[]$ is the expectation with respect to the time-varying system environments. Thus, we can express the original optimization problem $P1$ as finding the optimal service placement and scheduling strategy a^* to minimize the action-value $Q(s, a)$

$$a^* = \text{argmin}_{a\in\mathcal{A}} Q(s, a) \tag{26}$$

where $\mathcal{A}$ is the action space.

C. Deep Q-Network-Based Service Placement and Workload Scheduling Algorithm

Based on the components defined above, our goal is to design the optimal service placement decision $\mathcal{X}$ and workload scheduling $\mathcal{P}$ according to the request of the service. Our algorithm architecture is shown in Fig. 3. Specifically, considering the limited computing and storage capacity of the edge cloud, we adopt the strategy, which is based on offline training of Q-network parameter θ and online service placement and workload scheduling decision a_t. The offline training refers to the Q network parameter θ, where the $\theta = (W_1, W_2)$, describes the nonlinear relationship among service request information and the reward of Q-network. The online decision-making refers to the service placement and workload scheduling strategy a_t. Moreover, the DSP algorithm is shown in Algorithm 1 with the following specifications.

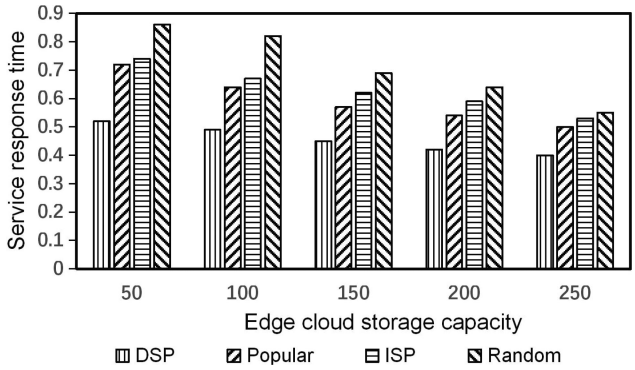

Fig. 4. Impact of the edge cloud storage capacity for different service placement schemes.

Let us denote E as the epoch of this algorithm where in each epoch, as shown in line 4, we choose an action a_t. Specifically, in this article, we adopt the ε-greedy search strategy where an action with possible probability ε (with uniform distribution) is selected among all possible actions. For the purpose of exploitation, the known best action can be selected by the probability of $1-\varepsilon$ to be utilize. Subsequently, in line 5, we execute the service placement and workload scheduling strategy of a_t to obtain the corresponding reward R_t, and to change the state space from s_t to s_{t+1}. We store the transitions (s_t, a_t, R_t, s_{t+1}) in the reply memory $\mathcal{D}$, as shown in line 6. Then, we use the data stored in the reply memory to train the parameters θ_t in the Q-network, where we select $|\tilde{D}|$ data as a minibatch from the reply memory, and train the network based on the following loss function:

$$\mathcal{L}(\theta_t)=\frac{1}{|\tilde{D}|}\sum_{j=1}^{|\tilde{D}|}\left(R_j+\gamma\min_{a'}Q(s_{i+1},a';\bar{\theta})-Q(s_i,a_i;\theta_t)\right)^2. \tag{27}$$

The parameter θ_t can be solved by the gradient descent method as

$$\theta_{t+1}=\theta_t-\eta\nabla L(\theta_t) \tag{28}$$

where η is the learning rate. After the C step, we reset the θ. Through the above steps, we can get the parameter θ in the Q-network. Based on the above training model, we propose the online service placement and scheduling strategy as shown in lines 13–15.

D. Convergence Analysis

According to [24], when the system satisfies Markov property and the learning rate is small enough, DQN method can gradually converge to the optimal decision-making strategy. According to the above analysis, our algorithm satisfies this condition. Thus, the DSP algorithm proposed by us finally converges to the optimal strategy.

V. Performance Analysis

In this section, we evaluate the DSP algorithm to verify the effectiveness of our algorithm.

Algorithm 1: DQN-Based Service Placement and Workload Scheduling Algorithm.

Input:
State of edge system in industrial CPS;
Random initialization parameters;
Output:
Service placement decision and workload scheduling scheme.
Offline training the Q-network
1: **for** epoch=1:E **do**
2: Obtain the initial state s_t.
3: **for** time slot t=1:T **do**
4: select an action a_t based on ε-greedy policy;
5: Deploy a_t, observe the reward R_t and obtain the new state s_{t+1};
6: Store transition (s_t, a_t, R_t, s_{t+1}) into replay memory D;
7: Sample a mini-batch of transitions $\tilde{D}=(s_t,a_t,R_t,s_{t+1})_{t=1}^{|\tilde{D}|}$ from replay memory;
8: Calculate the gradient $L(\theta_t)$ according to (27);
9: Update the parameters according to (28);
10: Every C steps, reset $\bar{\theta}=\theta$.
11: **end for**
12: **end for**
Online making service placement and scheduling decision according to the service demand
13: Load the parameters θ;
14: Calculate action-value $Q(s_t,a;\theta)$;
15: Output $a_t=\text{argmin}Q(s_t,a;\theta)$

A. Experiment Setting

In this article, we assume the edge clouds are deployed near the access point. Industrial CPS devices can be connected to edge cloud via an access point. We consider the industrial CPS includes 10 edge clouds and 15 services. According to [12], we assume that the storage of each service required follows a uniform distribution with a mean of [20,80] GB, and the amount of computation required for each service follows a uniform distribution with a mean of [10,50] gigacyles. Considering the heterogeneity of storage and computing capacity of the edge cloud, we assume that the storage capacity of the edge cloud is randomly selected from $[100, 200]$ GB, and the computing capacity is randomly selected from $[50, 100]$ GHZ. For edge cloud network transmission, we assume that the transmission rate between edge clouds is [20,40] Mbps. As for service request demand, we use real datasets in [8].

For DQN algorithm, the network is realized through Tensorflow, whose network structure is a fully connected three-layer network with a single hidden layer, and the number of nodes in the hidden layer is set as 50. For the experience pool, circular queue is adopted for implementation, and the size of the experience pool is 2000. Moreover, we adopt ε-greedy strategy for the action selection strategy, where $\varepsilon=0.9$. In the early stages of network training and to encourage exploratory behavior, we set

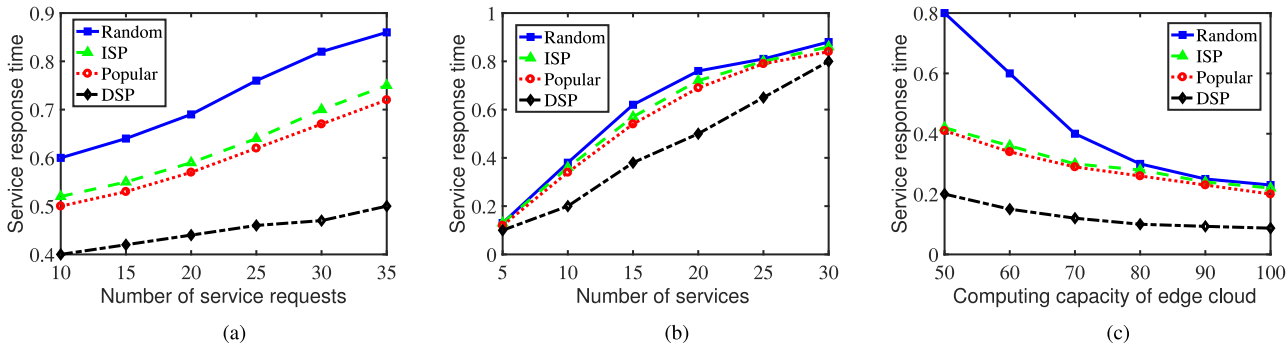

Fig. 5. Performance evaluation among random service placement algorithm, ISP algorithm, popular service placement algorithm and DSP algorithm. (a) Impact of the number of service requests for different service placement schemes. (b) Impact of the number of service for different service placement schemes. (c) Impact of the edge cloud computing capacity for different service placement schemes.

ε to start from 0, and increment as 0.0005. As for the batch-size, after many tests, the optimal batch-size is chosen as 64. The learning rate of the algorithm is set as $\eta = 0.0001$, and the discount factor is set as $\gamma = 0.9$.

B. Comparison Algorithms

The DSP algorithm proposed in this article will be compared with the following algorithms.

1) *Random Service Placement Algorithm*: In this algorithm, services are placed on the edge cloud randomly. As for workload scheduling and resource allocation, optimal scheduling and resource allocation is conducted.
2) *Independent Service Placement (ISP) Algorithm*: In this algorithm, each of the edge cloud carries out independent service placement, and the workload of each edge cloud is processed either locally or offloaded to the remote cloud. For the edge cloud computing resources, optimal resource allocation is carried out according to the placed service and workload.
3) *Popular Service Placement Algorithm*: In this algorithm, we perform service placement according to the number of requests, i.e., the services with more requests are placed on each edge cloud. According to the services placed on the edge cloud, optimal workload scheduling, and resource allocation is conducted.

C. Performance Evaluation

In order to avoid the noise data caused by exploration in the experiment results, in our experiments averaging is conducted to return values every 10 rounds. Furthermore, in the experiment, we normalize the service response time.

Edge Cloud Storage Capacity: We first analyze the impact of edge cloud storage capacity on the service response time. The edge cloud storage capacity varies from 50 to 250 GB. From Fig. 4, we can obtain that when the edge cloud storage capacity increases, the service response time decreases. This can be explained as a larger edge cloud storage capacity would allows more services to be placed in edge cloud causing a reduction in the service response time.

Number of Service Requests: Fig. 5(a) shows that as the number of service requests increases, the service response time increases. This is because a larger service request implies a heavier workload on the edge cloud. In Fig. 5(a), we find that when the number of service requests increase from 15 to 25, the service response time of the DSP algorithm increases by 12.3 %, while the other algorithms increase by at least 18.2 %. This shows that our proposed DSP algorithm is more suitable for uncertain service demands. Moreover, with the increase of service requests, the DSP algorithm keeps the lowest service response time when compared with other algorithms.

Number of Services: From Fig. 5(b), we can see that when the number of services increase, the service response time increase of each algorithm. This is because with an increase of services, the computing and storage capacity of the edge cloud is limited, so more services are processed in the cloud, thus creating more delays. Furthermore, when the number of services increases to 30, compared with other algorithms, the DSP algorithm can reduce the service response time by 8–10%.

Computing Capacity of Edge Cloud: We have discussed the impact of the computing capacity of the edge cloud on service response time. From the Fig. 5(c), we can conclude that, with the growth of the computing capacity of each edge cloud, service response times of each algorithm decrease. This is because more industrial services can be executed on the edge cloud. Furthermore, we can also see from the figure that, compared with other service placement schemes, our proposed DSP scheme has less service response time, especially when the computing capacity of the edge cloud is weak.

VI. Conclusion

In this article, we first propose softwarized-industrial CPS, and analyze the problem of service placement in this scenario. Then, in consideration of uncertain service demand, we formulate the joint optimization problem of service placement, workload scheduling and resource allocation to minimize latency in service acquisition. Furthermore, in order to solve the optimization problem, we propose a DSP algorithm using convex optimization and DRL. To the best of our knowledge, this is the first study of joint service placement, workload scheduling,

and resource allocation under uncertain demands. Simulation results show that our proposed DSP scheme is more efficient compared to the existing schemes.

REFERENCES

[1] S. Garg, K. Kaur, N. Kumar, and J. J. P. C. Rodrigues, "Hybrid deep learning-based anomaly detection scheme for suspicious flow detection in SDN: A social multimedia perspective," *IEEE Trans. Multimedia*, vol. 21, no. 3, pp. 566–578, Mar. 2019.

[2] T. Wang, P. Wang, S. Cai, Y. Ma, A. Liu, and M. Xie, "A unified trustworthy environment establishment based on edge computing in industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 16, no. 9, pp. 6083–6091, Sep. 2020.

[3] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Commun. Surv. Tut.*, vol. 19, no. 3, pp. 1628–1656, Jul.–Sep. 2017.

[4] S. Garg, S. Amritpal, B. Shalini, K. Neeraj, and Y. Laurence, "UAV-empowered edge computing environment for cyber-threat detection in smart vehicles," *IEEE Netw.*, vol. 32, no. 3 pp. 42–51, May/Jun. 2018.

[5] S. Garg, K. Kaur, G. Kaddoum, J. Rodrigues, and M. Guizani, "Secure and lightweight authentication scheme for smart metering infrastructure in smart grid," *IEEE Trans. Ind. Informat.*, vol. 16, no. 5, pp. 3548–3557, May 2020.

[6] S. Wang, R. Urgaonkar, T. He, K. Chan, M. Zafer, and K. K. Leung, "Dynamic service placement for mobile micro-clouds with predicted future costs," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 4, pp. 1002–1016, Apr. 2017.

[7] Y. Hao *et al.*, "Cognitive-caching: Cognitive wireless mobile caching by learning fine-grained caching-aware indicators," *IEEE Wireless Commun.*, vol. 27, no. 1, pp. 100–106, Feb. 2020.

[8] L. Chen, P. Zhou, L. Gao, and J. Xu, "Adaptive fog configuration for the industrial internet of things," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4656–4664, Oct. 2018.

[9] L. Wang, L. Jiao, T. He, J. Li, and M. Mühlhäuser, "Service entity placement for social virtual reality applications in edge computing," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, 2018, pp. 468–476.

[10] Q. Zhang, Q. Zhu, M. F. Zhani, R. Boutaba, and J. L. Hellerstein, "Dynamic service placement in geographically distributed clouds," *IEEE J. Sel. Areas Commun.*, vol. 31, no. 12, pp. 762–772, Dec. 2013.

[11] K. Poularakis, J. Llorca, A. M. Tulino, I. Taylor, and L. Tassiulas, "Joint service placement and request routing in multi-cell mobile edge computing networks," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, 2019, pp. 10–18.

[12] X. Ma, A. Zhou, S. Zhang, and S. Wang, "Cooperative service caching and workload scheduling in mobile edge computing," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, 2020, pp. 2076–2085.

[13] V. Farhadi *et al.*, "Service placement and request scheduling for data-intensive applications in edge clouds," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, 2019, pp. 1279–1287.

[14] N. Eshraghi and B. Liang, "Joint offloading decision and resource allocation with uncertain task computing requirement," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, 2019, pp. 1414–1422.

[15] S. Garg, K. Kaur, G. Kaddoum, S. H. Ahmed, and D.N.J. Jayakody, "SDN based secure and privacy-preserving scheme for vehicular networks: A 5G perspective," *IEEE Trans. Veh. Technol.*, vol. 68, no. 9, pp. 8421–8434, Sep. 2019.

[16] L. Chen, S. Zhou, and J. Xu, "Computation peer offloading for energy-constrained mobile edge computing in small-cell networks," *IEEE/ACM Trans. Netw.*, vol. 26, no. 4, pp. 1619–1632, Aug. 2018.

[17] M. Chen, Y. Hao, H. Gharavi, and V. Leung, "Cognitive Information Measurements: A New Perspective," *Inf. Sci.*, vol. 505, pp. 487–497, Dec. 2019.

[18] S. Garg, K. Kaur, S. H. Ahmed, A. Bradai, G. Kaddoum, and M. Atiquzzaman, "MobQoS: Mobility-aware and QoS-driven SDN framework for autonomous vehicles," *IEEE Wireless Commun. Mag.*, vol. 26, no. 4, pp. 12–20, Aug. 2019.

[19] T. X. Tran, K. Chan, and D. Pompili, "COSTA: Cost-aware service caching and task offloading assignment in mobile-edge computing," in *Proc. 16th Annu. IEEE Int. Conf. Sens., Commun., Netw.*, 2019, pp. 1–9.

[20] S. Garg, A. Gagangeet, K. Neeraj, and B. Shalini, "Tree-based attack–defense model for risk assessment in multi-UAV networks," *IEEE Consum. Electron. Mag.*, vol. 8, no. 6, pp. 35–41, Nov. 2019.

[21] T. He, H. Khamfroush, S. Wang, T. La Porta, and S. Stein, "It's hard to share: Joint service placement and request scheduling in edge clouds with sharable and non-sharable resources," in *Proc. IEEE 38th Int. Conf. Distrib. Comput. Syst.*, 2018, pp. 365–375.

[22] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge Univ. Press, 2004.

[23] S. Garg, K. Kaur, N. Kumar, G. Kaddoum, A. Zomaya, and R. Ranjan, "A hybrid deep learning based model for anomaly detection in cloud datacentre networks," *IEEE Trans. Netw. Service Manag.*, vol. 16, no. 3, pp. 924–935, Sep. 2019.

[24] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 2094–2100.

Yixue Hao received the Ph.D degree in computer science from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2017.

He is an Associate Professor with the School of Computer Science and Technology at Huazhong University of Science and Technology. His current research interests include 5G network, Internet of Things, edge computing, edge caching, and cognitive computing.

Min Chen (Fellow, IEEE) received the Ph.D. degree in computer science from the South China University of Technology, Gongdong, China, in 2004.

He is a Full Professor with the School of Computer Science and Technology at Huazhong University of Science and Technology (HUST), Wuhan, China, since Feb. 2012. He is the Director of Embedded and Pervasive Computing (EPIC) Lab at HUST. His Google Scholar Citations reached 25800+ with an h-index of 79 and i10-index of 240. His top paper was cited 2950+ times. He was selected as Highly Cited Researcher at 2018, 2019, and 2020. His research interests include cognitive computing, big data analytics, robotics, deep learning, emotion detection, and mobile edge computing, etc.

Mr. Chen is the founding Chair of IEEE Computer Society Special Technical Communities (STC) on Big Data. He was the recipient of the IEEE Communications Society Fred W. Ellersick Prize, in 2017, and the IEEE Jack Neubauer Memorial Award, in 2019.

Hamid Gharavi received the Ph.D. degree from Loughborough University, Loughborough, U.K., in 1980.

He joined the Visual Communication Research Department, AT&T Bell Laboratories, Holmdel, NJ, USA, in 1982. He was then transferred to Bell Communications Research (Bellcore) after the AT&T-Bell divestiture, where he became a Consultant on video technology and a Distinguished Member of the Research Staff. In 1993, he joined Loughborough University as a Professor and Chair of Communication Engineering. Since September 1998, he has been with the National Institute of Standards and Technology, U.S. Department of Commerce, Gaithersburg, MD, USA. His research interests include smart grid, wireless multimedia, mobile communications and wireless systems, mobile ad hoc networks, and visual communications

Dr. Gharavi was a Core Member of Study Group XV (Specialist Group on Coding for Visual Telephony) of the International Communications Standardization Body CCITT (ITU-T) and a member of the IEEE 2030 Standard Working Group.

Yin Zhang received the Ph.D. degree in computer science from the Yunnan University, Kunming, China, in 2013.

He is a Full Professor with the School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu, China. He is a Distinguished Scholar of Hubei Province, China. He has authored or coauthored more than 100 prestigious conference and journal papers, including 14 ESI highly cited papers. His research interests include mobile computing, edge intelligence, cognitive wireless communications, etc.

He is the Co-Chair of IEEE Computer Society Big Data Special Technical Communities (STC). He was the recipient of the Systems Journal Best Paper Award of the IEEE Systems Council, in 2018. He was named in Clarivate Analytics Highly Cited Researchers List, in 2019. He serves as Editor or Associate Editor for IEEE NETWORK, *Information Fusion*, IEEE ACCESS, etc. He is a Guest Editor for IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING, *Future Generation Computer Systems*, IEEE IoT JOURNAL, *ACM/Springer Mobile Networks & Applications*, *Sensors*, *Neural Computing and Applications*, *Multimedia Tools and Applications*, *Wireless Communications and Mobile Computing*, *Electronic Markets*, *Journal of Medical Systems*, *New Review of Hypermedia and Multimedia*, etc. He also served as Track Chair of IEEE CSCN 2017, TPC Co-Chair of CloudComp 2015, and TRIDENTCOM 2017, etc.

Kai Hwang received the Ph.D. degree in electrical engineering and computer science from the University of California, Berkeley, CA, USA, in 1972.

He is a Professor of Electrical Engineering and Computer Science, University of Southern California (USC), Los Angeles, CA. Prior to joining USC, in 1986, he has taught at Purdue University, West Lafayette, IN, USA, for 11 years. Dr. Hwang has authored or coauthored eight books and 250 scientific papers. According to Google Scholars, his work was cited over 15 000 times with an h-index of 52. His most cited book on Computer Architecture and Parallel Processing was cited more than 2300 times and his PowerTrust (IEEE-TPDS, April 2007) paper was cited over 540 times.

Dr. Hwang was the recipient Lifetime Achievement Award from IEEE Cloudcom-2012 for his pioneering contributions in the field of computer architecture, parallel, distributed, and cloud computing, and cyber security. He has served as the founding Editor-in-Chief for the *Journal of Parallel and Distributed Computing* from 1983 to 2011.