Enhanced Online Q-Learning Scheme for Resource Allocation with Maximum Utility and Fairness in Edge-IoT Networks

Ismail AlQerm*, *Member, IEEE*, Jianli Pan*, *Member, IEEE*

***Abstract*—Internet of Things (IoT) is experiencing an explosion in the data traffic due to the increase in the number of heterogeneous applications. The existing cloud computing models will not be capable to support the IoT applications that are delay-sensitive and using high bandwidth. The Edge-IoT systems represented by shared edge clouds support a wide range of IoT applications. Edge clouds provide resources closer to the IoT devices to tackle the delay sensitivity and bandwidth issues. However, the allocation of these resources with guaranteed application's utility in the context of Edge-IoT with multiple heterogeneous IoT applications, various resource demands, and limited resource availability is challenging. In this paper, we propose a novel enhanced online Q-learning scheme to allocate resources from edge clouds to IoT applications to maximize their utility and maintain allocation fairness among them. The developed online Q-learning scheme approximates its Q-value to tackle the problem of large state space, reduce the required learning computation, and expedite the system convergence. It is implemented using two settings: centralized using a dedicated controller at the edge cloud and distributed where edge servers learn cooperatively to achieve a common goal of finding joint resource allocation policy that maximizes the IoT applications' utilities. Extensive numerical results demonstrate the capability of the proposed scheme in improving applications' utilities and allocation fairness.**

***Index Terms*—Edge Computing, Internet of Things, Resource Allocation, Online Q-Learning**

I. Introduction

With the current growth of heterogeneous IoT applications such as 4K/8K UHD video, hologram, interactive mobile gaming, smart homes, etc. [1], there will be tremendous demand for network and computing resources to support these applications. It will be very difficult for the existing centralized cloud computing systems to scale with projected billions or even trillions of weak IoT devices and ubiquitous applications, due to the large amount of generated data and the relatively long distance between IoT devices and clouds [2]. Edge computing is considered as a potential approach that brings more computing, networking, storage, and intelligence resources to the edge, which would specifically benefit IoT applications that are delay-sensitive, bandwidth/data intensive, or that require closer intelligence [3], [4]. Moreover, virtualized and shared "edge clouds" and the corresponding Edge-IoT systems are enablers for the real time applications such as autonomous driving, robotics, smart homes and healthcare, which require low latency [1], [4], [5].

However, unlike cloud computing, the edge computing paradigm is still in the developing stage. Cloud computing has virtually unlimited computing and networking resources in large data centers, while edge computing has usually limited resources over distributed edge nodes that have different configurations and capabilities. In addition, heterogeneous IoT applications have different resource, quality, and priorities requirements, which makes the resource allocation problem more challenging. The resource allocation problem in Edge-IoT with these characteristics in addition to consideration of IoT applications' utilities and fairness is not well-explored.

In this paper, we propose a resource allocation model that aims to maximize the IoT applications' utilities with consideration of allocation fairness among the heterogeneous IoT applications. The application utility is defined as the gain received by these applications as a result of the resource allocation and it is evaluated in terms of the number of applications' requests successfully served by the edge cloud. The utility is tied with the number of resource requests initiated by the applications and time consumed to process these requests. To achieve the maximization goal, we develop a novel online Q-learning scheme to allocate resources to the IoT applications. The proposed scheme has the following merits. 1) It is aware of contextual information from the IoT application side including delay requirements and priority, and from the edge cloud side including the edge server capacity and server load. 2) The scheme exploits such awareness information to allocate edge resources such that the utility of the applications is maximized and the resource are fairly allocated. For example, applications that have urgent delay requirements will have the priority in resource allocation over the ones that are not delay sensitive. 3) The scheme tackles the dimensionality issue of online Q-learning due to the large state/action space in Edge-IoT systems by developing approximation mechanism for the Q-value. The Q-value is approximated as a function of much smaller set of variables. This reduces the scheme complexity as smaller state/action space is exploited and expedites the scheme convergence. 4) The proposed online Q-learning scheme uses the historical resource allocation actions of the edge servers to achieve distributed and cooperative resource allocation which enhances the scalability of the system and avoids the bottleneck processing of the typical centralized allocation. To the best of our knowledge, the proposed scheme is the first to use online Q-learning to

*Ismail AlQerm and Jianli Pan are with Dept. of Mathematics and Computer Science, University of Missouri, St. Louis, USA. (Email: alqermi@umsl.edu, pan@umsl.edu)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2020.3015689, IEEE Transactions on Network Science and Engineering

tackle the utility based resource allocation with the above merits for multiple heterogeneous applications at the Edge-IoT networks.

We choose online Q-learning [6] for such resource allocation problem as it is model free and the system can learn from its past experience. It is challenging to determine the exact state transition model for such dynamic Edge-IoT environment by applying a model-based method such as dynamic programming algorithms. Moreover, it is not trivial to list all the state and action pairs to migrate from one state to another, and it is not practical to pre-define the state transition model in problem solving. The paper has the following unique contributions:

- We propose a novel resource allocation model that aims to maximize the IoT applications' utilities with consideration of multiple applications' priorities and various delay requirements, and with guaranteed fairness in resource allocation. In the proposed model, the resources are allocated from the edge servers to the IoT applications with awareness of the following factors: the edge server load, the capacity of each server, and the IoT application's requirements that are determined by the application type.
- We develop an enhanced centralized online Q-learning based resource allocation scheme that aims to maximize IoT applications utilities. This scheme employs a controller at the network edge which exploits awareness information of edge cloud and IoT applications to achieve resource allocation.
- We propose a novel distributed and cooperative online Q-learning based resource allocation scheme that exploits the capability of edge servers to learn cooperatively through sharing their historical resource allocation actions and the fact that allocation actions do not alter for similar network states. Thus, edge servers exploit these historical actions for allocation decisions if they encounter similar network scenarios.
- The proposed online Q-learning model accounts for the dimensionality problem in typical Q-leanring due to the large state/action space. The proposed learning model narrows the state space by approximating Q-value as function of smaller set of variables using a brief representation feature. This reduces the required computation and expedites the convergence significantly in comparison to standard online Q-learning.

The rest of the paper is organized as follows. The related work is presented in Section II. Section III describes the system model, fairness characterization, and problem formulation. The centralized and distributed online Q-learning schemes for resource allocation are presented in Section IV. The numerical results are discussed in Section V and the paper concludes in Section VI.

II. Related Work

The recent work in the literature addressed the potential benefits and technical aspects of edge computing. Many of the recent work focus on the workload offloading problem [7]. The traffic offloading facilitates access for IoT applications to resource-rich servers at the edge to handle their power exhaustive computation. In [8], a workload offloading optimization problem was proposed to minimize energy consumption under the latency constraint. Another scheme was proposed in [9] to tackle the workload offloading problem considering latency to minimize the average response time. In [10], the authors proposed different profit maximization frameworks for cloud providers. [11] addressed the utility based pairing problem between the fog nodes and IoT devices with the Irving's matching algorithm. All these solutions focus on the offloading problem from the IoT devices to the edge servers without considering solid models for resource allocation.

Other work considered the resource allocation problem in Edge-IoT context focusing on serving data service subscribers (DSSs). For example, the work in [12] considered a three-tier edge network, where the data service operators can obtain computing resources from different edge servers to serve their DSSs using stackelberg game based joint optimization. From the DSS's side, the edge cloud was able to surrogate the requirements and simplify the management of the network. From the servers' side, the edge cloud can exploit content and support service delivery in an efficient way. The work in [13] proposed an edge computing framework that can provide rich flexibility in meeting different mobile users' demands. The authors in [14] proposed an Edge-IoT architecture to enhance the system performance and computational resource management. It focuses on admission control, computational resource allocation, and power control. The authors in [15] exploits reinforcement learning at the IoT devices to decide whether to offload data for processing to the edge or perform it locally with energy optimization objective. Deep reinforcement learning based resource allocation scheme, which aims to allocate computing and network resources to reduce the average service time and balance the use of resources, was proposed in [16]. Auction theory was also adopted in [17] to allocate resources for IoT devices with various prices. Auction schemes do not take the IoT users budget into consideration for acquiring resources as they only focus on the servers profit from providing the computation resources.

None of the proposed work in the literature that tackles the resource allocation in Edge-IoT networks considering multiple applications with heterogeneous delay requirements and fairness guarantee at the edge, which is essential in the context of IoT networks. In addition, none of them considered cooperation between edge servers through exploitation of historical allocations. In addition, it is vital to balance between the efficiency and fairness in resource allocation. Conventional schemes such as social welfare maximization [18] and auction models [19] may not be suitable as they either give the resources to the applications with high utility as the case in the welfare maximization, or only to the auction winners as in the auction based models.

III. System Descriptions

This section describes the system model, fairness characteristics, and the resource allocation problem formulation.

A. System Model

The considered system model of Edge-IoT is depicted in Fig. 1, where there are multiple distributed edge servers with

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2020.3015689, IEEE Transactions on Network Science and Engineering

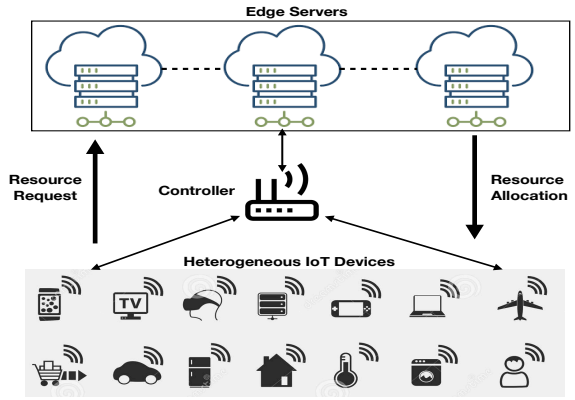

Fig. 1. Edge-IoT Network Model

various computation capabilities and multiple heterogeneous IoT applications that need to acquire available computation resources to process their data. All the IoT applications' requests for resources are processed either at a controller located at the edge for centralized allocation setting or at the edge servers for the distributed allocation. Each IoT application sends dynamic requests for resources from the edge and aims to maximize its utility. It is assumed that the resource requests arrival rate and resource capacity vary at each time instance. The edge server load and capacity vary over time based on the demand of the IoT devices and the resources allocated. Edge server's resources capacity, edge server's load, application's delay requirements, and application's priority are the main factors that impact the resource allocation. Therefore, our resource allocation scheme takes these factors into consideration to harmonize the applications' interests and support fairness in resource allocation. The mentioned environment factors related to the edge servers and the IoT devices can be very dynamic over the time. Thus, it is difficult to determine an exact model for this dynamic environment which makes model-free solutions more appealing for such environment.

Let **M**, **N**, M, and N be the set of edge servers, the set of IoT devices, the number of edge servers, and the number of IoT devices respectively. i and j are the indexes for the IoT application and the edge server respectively assuming that each IoT device runs one application. The capacity for each edge server j is denoted by c_j, which is the number of computing resources available at that server. The edge server is divided into a set of clusters each of which has homogeneous computing resources. In case the edge server incorporates various computing resources, each cluster is considered as a separate edge server. Note that distinct edge servers can have distinct capacity and one application can receive resources from multiple edge servers. Let $s_{i,j}$ be the resources of server j that are allocated to application i. Hence, the vector of resources allocated to application i from multiple edge servers is denoted by $s_i = (s_{i,1}, s_{i,2}, ..., s_{i,M})$. Our resource allocation scheme aims to find a computation resource allocation policy π for application i from server $j \in \mathbf{M}$. The application utility is denoted as $u_i(s_i)$. Different IoT applications have distinct methods to define $u_i(s_i)$. In this work, we consider delay as the main performance metric for the utility evaluation since delay sensitive applications are the ones that need maximum support of edge computing. The delay encountered at the edge server for certain resource requests initiated by the IoT applications consists of processing delay and network delay. The processing delay $d^p_{i,j}$ is the time required at the edge server to process the resources requests. The network delay $d^n_{i,j}$ includes the transmission delay and the propagation delay which is the round trip time between the IoT device and the edge server and calculated as in [9].

The maximum encountered delay for processing, transmission and propagation must be maintained below certain threshold $d^{max}_{i,j}$ as follows,

$$\omega_j d^p_{i,j} + d^n_{i,j} \leq d^{max}_{i,j} \quad , \forall i, j \tag{1}$$

where ω_j is the server load that impact the processing delay and is defined as the ratio of the received requests for resources and the server maximum capacity. The arrival rate of requests from application i to server j is denoted as $\lambda_{i,j}$. The average processing time at the server j is computed as follows,

$$d^p_{i,j} = \frac{1}{k_{i,j} - \frac{\lambda_{i,j}}{s_{i,j}}} \quad , \forall i, j \tag{2}$$

where $k_{i,j}$ is the service rate of one computing unit of server j to handle application i with constraint $\frac{\lambda_{i,j}}{s_{i,j}} < k_{i,j}$. By substituting (2) in (1),

$$\omega_j \frac{1}{k_{i,j} - \frac{\lambda_{i,j}}{s_{i,j}}} \leq d^{max}_{i,j} - d^n_{i,j}$$

$$\lambda_{i,j} \leq s_{i,j}(k_{i,j} - \frac{\omega_j}{d^{max}_{i,j} - d^n_{i,j}}) \tag{3}$$

If $d^n_{i,j} < d^{max}_{i,j}$, the maximum number of requests that server j processes for application i is,

$$\lambda^{max}_{i,j} = max\{s_{i,j}(k_{i,j} - \frac{\omega_j}{d^{max}_{i,j} - d^n_{i,j}}), 0\}$$

$$\lambda^{max}_{i,j} = s_{i,j} q_{i,j}, \quad \forall i, j \tag{4}$$

where $q_{i,j} = max\{(k_{i,j} - \frac{\omega_j}{d^{max}_i - d^n_{i,j}}), 0\}$. The service of requests of application i is considered successful if the total delay is less than or equal the maximum delay tolerance. The utility of application i with resources allocated from server j is calculated as follows,

$$u_{i,j}(s_{i,j}) = h_{i,j} q_{i,j} s_{i,j} = \rho_{i,j} s_{i,j} \quad \forall i, j \tag{5}$$

where $\rho_{i,j} = h_{i,j} q_{i,j}$ and $h_{i,j}$ is the gain of the application i resulted from computing resources allocation from server j. It is assumed that $\rho_{i,j}$ is computed beforehand. The total utility for application i with resources allocated from multiple servers is given by,

$$u_i(s_i) = \sum_{j=1}^{M} u_{i,j} = \sum_{j=1}^{M} \rho_{i,j} s_{i,j} \quad \forall i \tag{6}$$

One point to note is that the IoT devices prefer to request services from edge servers that are close and lightly loaded. Thus, the value of $\rho_{i,j}$ is dependent on the load for server j. The IoT application priority denoted by p_i is assumed to be predetermined and ranked based on the application's delay requirements. For example, the application with minimal delay requirements will be given the highest priority in resource allocation.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2020.3015689, IEEE Transactions on Network Science and Engineering

B. Fairness Characterization

We characterize the fairness of the proposed utility model using the envy-freeness feature [20], where every IoT application i feels that its resource share from certain edge server j is at least as good as other applications. Therefore, the application i will not envy other applications. In the proposed utility model, each application i aims to maximize its utility $u_i(s_i)$ with capacity constraint $\sum_{i=1}^{N} s_{i,j} \leq c_j \ \forall j \in \mathbf{M}$ which indicates that the edge servers may have different capacities c_j. The term $\zeta_i = \max_j\{\rho_{i,j}s_{i,j}\}$ is defined as the maximum achieved utility of application i over a set of edge servers. The preferred set PR_i of application i includes all the edge servers that support the achievement of ζ_i such that $PR_i = \{j : \rho_{i,j}s_{i,j} = \zeta_i\}, \forall i$. Thus, each application i will aim to obtain its resources from the server that can give ζ_i (maximum utility). When the capacities of the edge servers are the same, an envy free allocation indicates $u(s_i) \geq u(s_i')$ for all i and $i' \in \mathbf{N}$. Since the servers have different capacities, it implies that the maximum utility of application ζ_i can only be achieved through allocation from PR_i. Thus, the envy-freeness classical definition has to be extended i.e. an allocation is envy-free if:

$$\frac{u_i(s_i)}{\zeta_i} \geq \frac{u_i(s_{i'})}{\zeta_{i'}} \ \ \forall i^{'} \in \mathbf{N} \tag{7}$$

where $i^{'} \in \mathbf{N}, i^{'} \neq i$ is the index of other IoT applications. The inequality in (7) can be written as,

$$u_i(s_i)\zeta_{i'} \geq u_i(s_{i'})\zeta_i \ \ \forall i, i^{'} \in \mathbf{N} \tag{8}$$

Now, we can prove that the allocation is envy-free if the condition in (8) holds as follows,

$$u_i(s_i)\zeta_{i'} = \zeta_{i'} \sum_{j=1}^{M} \rho_{i,j}s_{i,j} = \zeta_{i'} \sum_{j=1}^{M} \rho_{i,j} \frac{\lambda_{i,j}^{max}}{q_{i,j}} \tag{9}$$

$$\zeta_{i'} \sum_{j=1}^{M} \frac{\rho_{i,j}}{q_{i,j}} \lambda_{i,j}^{max} = \zeta_{i'} W_i \sum_{j=1}^{M} \lambda_{i,j}^{max} \tag{10}$$

$$\zeta_{i'} W_i \sum_{j=1}^{M} \lambda_{i,j}^{max} = \zeta_{i'} W_i \zeta_i \tag{11}$$

In (9), we substitute $s_{i,j} = \lambda_{i,j}^{max}/q_{i,j}$ according to (4). Note that in (10), W_i refers to the total gain IoT application i receives as a result of allocation from edge servers $j \in PR_i$ and defined according to (5) as $W_i = \sum_{j}^{PR_i} \frac{\rho_{i,j}}{q_{i,j}}$. This can be inferred from the fact that each application will acquire resources from edge server in its preferred list PR_i to maximize its utility. The results in (11) confirms that application i achieves its maximum utility as the maximum number of resource requests is satisfied when application i finds the ultimate allocation at certain server j.

$$\zeta_{i'} W_i \zeta_i = W_i \zeta_i \sum_{j=1}^{M} \lambda_{i',j}^{max} = \zeta_i \sum_{j=1}^{M} \frac{\rho_{i,j}}{q_{i,j}} \lambda_{i',j}^{max}$$

$$\zeta_i \sum_{j=1}^{M} \frac{\rho_{i,j}}{q_{i,j}} \lambda_{i',j}^{max} = \zeta_i \sum_{j=1}^{M} \rho_{i,j} s_{i',j} = \zeta_i u_i(s_{i'}) \ \ \forall i, j \tag{12}$$

The fairness characteristic in (8) is proved as it is only fair if the IoT application i receives its resources from PR_i such that the utilities of all applications are maximized.

C. Problem Formulation

The allocation problem is formulated to maximize the $u_i(s_i)$ for all IoT applications subjected to the IoT application delay requirements and the edge server resources capacity constraints. The resource allocation problem to achieve the ultimate allocation of computing resources s_i^* is defined as follows,

$$\max_{(s_i \in S)} \ \sum_i u_i(s_i^*) \ \ s.t. \tag{13}$$

$$C1: \ \ \omega_j d_{i,j}^p + d_{i,j}^n \leq d_{i,j}^{max} \ \ , \forall i$$

$$C2: \ \ s_{i,j} \geq 0, \ \ \forall i, j$$

$$C3: \ \ \sum_{i=1}^{N} s_{i,j} \leq c_j, \ \ \forall j \in \mathbf{M}$$

$C1$ is the constraint that maintains the delay of serving certain application less than the maximum tolerable delay. Moreover, it implicitly accounts for the priority of the IoT applications as the applications that are delay sensitive are given higher priority in resource allocation. $C2$ confirms that the number of resources allocated is positive. The capacity constraint in $C3$ guarantees that the allocated resources from certain edge server $j \in \mathbf{M}$ never exceed its capacity.

IV. Online Q-Learning Scheme for Resource Allocation in Edge-IoT

In this section, we describe the online Q-learning scheme employed to achieve resource allocation using both the centralized and the distributed online Q-learning settings.

A. Online Q-Learning Scheme Overview

Online Q-learning [6] is a model-free reinforcement learning technique which can be used to find an action policy for any given Markov decision process (MDP). It works by learning a Q-value function that ultimately gives the expected utility of selecting certain action in a certain state and following the policy thereafter. MDP includes a discrete set of environment states A and a discrete set of actions B. At each epoch t, the learning agent obtains network state information a and selects certain action b. A reward R will be received and used to define certain Q-value which evaluates the action selected. The process continues iteratively until it converges to an action policy that maximizes the Q-value.

We formulate the MDP in this paper such that factors include server load, IoT devices, edge servers, server capacity, arrival rate of resources requests from IoT devices, and total delay threshold are taken into consideration in the resource allocation action selection. The state of the developed scheme evolves as a discrete-time Markov decision process (DTMDP) in which the resource allocation is defined as the action. The scheme aims to maximize the IoT applications utilities while

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2020.3015689, IEEE Transactions on Network Science and Engineering

satisfying the constraints defined in $C1$-$C3$. Such maximization problem lies within the domain of DTMDP. However, the large number of Edge-IoT states represents a significant impediment on developing a model-based scheme. Therefore, we choose model-free online Q-learning to solve the resource allocation problem with varying resource requests and edge resource capacity over time. The state transitions and actions occur at discrete time epochs. The DTMDP is formulated as (A, B, T, R) where $T : A \times B \times A \rightarrow [0, 1]$ is the state transition probability function. The actions available at each epoch depends on the Edge-IoT state space. For instance, if there is no resource requested from the edge, then the edge servers are switched off and no resource allocation will be necessary.

We consider the following DTMDP for resource allocation: $\aleph_t = (\mathbf{N}, \mathbf{M}, d_{i,j}^{max}, u_i(s_i), s_{i,j})$. The MDP components are defined as follows,

- **State:** the environment state at epoch t is defined as, a_t=(i, j, ω_j, c_j, $d_{i,j}^{max}$, $\lambda_{i,j}, p_i$). $a_t \in A$ describes the evolution of the network state in time epoch $t = 1, 2...$ The state information is acquired by the controller in the centralized setting or the corresponding edge server in the distributed one. p_i refers to the IoT application priority which is predetermined according to the application delay requirements.
- **Action:** $b_t = (s_{i,j})$ is defined as the allocation of computing resource $s_{i,j}$ for all IoT applications. It is selected for each state a_t at epoch t.
- **Reward:** the reward function is the utility achieved by all the IoT applications $\forall i$ and is defined as, $R_t(a, b) = \sum_i u_i(s_i)$.

The allocation policy $\pi : A \rightarrow B$ is defined as the probability of selecting certain action b_t at state a_t that maximizes the achieved utility subjected to the optimization constraints $C1$-$C3$. The value function which is defined as the total expected discounted utility function over infinite time horizon conditioned on initial network state a_1 for allocation policy $\pi \in \Pi$ is found as,

$$V_\pi(a_1) = E_\pi\Big\{\sum_{t=1}^{\infty} \beta^{t-1} R(a_t, \pi(b_t))|a_1\Big\} \tag{14}$$

where the expectation E_π is over distinct actions in distinct states for $t = 1, 2, 3...$ and $\beta \in [0, 1)$ is the discount factor. The DTMDP evolution for certain resource allocation policy $\pi \in \Pi$ is Markovian with the following state transition probability:

$$T(a, b, a') = Pr(a_{t+1} = a'|a_t = a, b_t = b) \tag{15}$$

where $a'_t \in A$ and $t = 1, 2, ...$ An exact model for the transition probability is impractical because: 1) the resource request processing is constrained by dynamic factors including the server load, server capacity, and IoT application delay requirements; 2) it is infeasible to list all the (a, b, a') pairs; 3) it is not desirable to predefine the state transition model for problems in which the real state information deviates from the model; 4) the DTMDP state space for such Edge-IoT system is large which makes it extremely difficult to compute the resource allocation policy. Therefore, we develop an online Q-learning based resource allocation scheme which gradually adapts to the dynamic Edge-IoT environment according to the received reward.

B. Centralized Resource Allocation Using Enhanced Online Q-learning

In this setting, we assume that a controller is located between the edge servers and the IoT applications. This controller follows the DTMDP defined in Section IV.A and manages the resource allocation process as it receives the state information from both sides (edge servers and IoT applications) and employs the online Q-learning technique to achieve the ultimate resource allocation policy. Fig. 2 presents the centralized resource allocation setting. When the Edge-IoT system is in the state $a_t \in A$ in epoch t, a finite set of possible actions can be selected by the controller from the action space B. The action b_t is selected by the controller in epoch t. The controller learns the optimal resource allocation policy which is defined based on the optimal Q-values $Q^*(a, b)$ as follows,

$$\pi^*(a) = arg \max_{b \in B} Q^*(a, b) \tag{16}$$

The optimal Q-value of the online Q-learning is defined as the current expected reward added to a future discounted reward as follows,

$$Q^*(a, b) = \mathbb{E}[R_t(a, b) + \beta \sum_{a' \in A} \max_{b' \in B} Q^*(a', b')] \tag{17}$$

where β is the discount factor and $b' \in B$ is the future action. The optimal Q-value $Q^*(a, b)$ is learned by updating the Q-value under the action b in epoch t as follows,

$$\begin{aligned} Q^{t+1}(a, b) = (1 - \alpha^t) Q^t(a, b) + \alpha^t [R_t(a, b) \\ + \beta max_{b' \in B} Q^t(a', b') - Q^t(a, b)] \end{aligned} \tag{18}$$

where $\alpha^t \in (0, 1]$ is the learning rate. The initialization of $Q^t(a, b)$ for all $(a, b) \in A \times B$ is arbitrary. The considered online Q-learning scheme is a stochastic approximation method established to solve the Bellman's optimality equation in (18) associated with the discrete time DTMDP. Online Q-learning does not need explicit state transition probability model and it converges with probability one to an optimal solution if $\sum_{t=1}^{\infty} \alpha^t$ is infinite, $\sum_{t=1}^{\infty} (\alpha^t)^2$ is finite, and all state/action pairs are visited infinitely [21]. These conditions are satisfied if the probability of the action selection in any state is not zero during the action exploration step. We utilize ϵ-greedy strategy [22] to balance exploration and exploitation in online Q-learning. ϵ is the percent of the time that the controller takes a randomly selected action (exploration) rather than taking the action that will maximize its reward (exploitation). Given the considered DTMDP above in Edge-IoT environment which can be very dynamic and has a large number of state/action pairs, it makes the representation of $Q^t(a, b)$ impossible. This curse of dimensionality in the proposed DTMDP increases the computational complexity. Thus, it is necessary to develop a form of brief representation in which the Q-values are approximated as a function of smaller set of variables with a reduced and countable state space A^+. The compact

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2020.3015689, IEEE Transactions on Network Science and Engineering

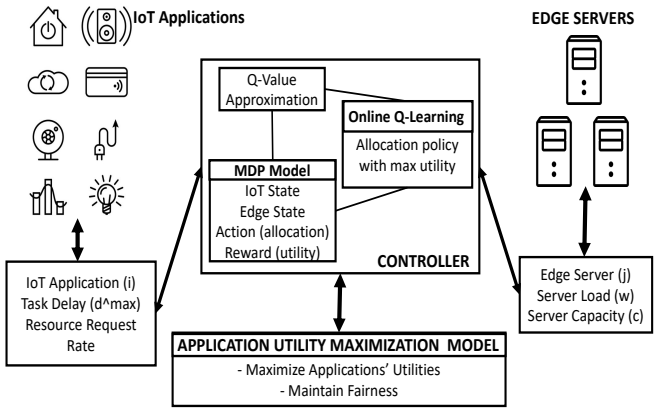

Fig. 2. Centralized resource allocation setting

representation of $Q : A^+ \times B \rightarrow \mathbb{R}$ is achieved using the function $Q^{'} : A^+ \times B \times \mathbb{R}^V \rightarrow \mathbb{R}$ which is called a function approximator.

To approximate the Q-function, we consider a group of functions $\mathbf{Q} = \{Q_\vartheta\}$ that are parameterized by a finite dimensional vector $\vartheta = [\{\vartheta_v\}_{v=1}^V] \in \mathbb{R}^V$. The iterative procedure to find Q^* is replaced by a suitable procedure to find ϑ^* instead such that Q^* is approximated by a function in $\mathbf{Q}$. Therefore, we switch from searching in an infinite dimensional function space to a finite dimensional space $(\mathbb{R}^V)$. An implication can be deduced that unless $Q^* \in \mathbf{Q}$, we cannot determine the exact Q^*. We will be able to determine the fixed point of a combined operator ϱH, where ϱ is a mapping that projects a function defined in $A^+ \times B$ to a point in $\mathbf{Q}$. The group $\mathbf{Q}$ is assumed to have a linear span of a basis function $\chi_v : A^+ \times B \rightarrow \mathbb{R}$ and any $q \in \mathbf{Q}$ can be written as a linear combination of $\chi_v(a,b)$. Therefore, the approximated $Q^{'}$ is calculated as follows:

$$Q^{'}(a,b,\vartheta) = \sum_{v=1}^{V} \vartheta_v \chi_v(a,b) = \vartheta \chi^T(a,b) \tag{19}$$

where T is the transpose operator and the vector $\chi(a,b) = [\chi_v(a,b)_{v=1}^V]$ with a scalar function $\chi_v(a,b)$ defined over $A^+ \times B$ and ϑ_v belongs to the vector $\vartheta \in \mathbb{R}^V$. $\chi_v(a,b)(v = 1,2,3....,V)$ is viewed as the basis functions and $\vartheta_v(v = 1,2,3....,V)$ are the associated weights. The basis functions $\chi_v(a,b)$ adopted for Q-value approximation are assumed to be bounded and linearly independent and $\sum |\chi_v(a,b))| \leq 1$ for all $(a,b) \in A^+ \times B$. We consider a sample based approximation model that introduce further restrictions on the set of the basis functions. This allows to derive error bounds for the given approximation Q_{ϑ^*}. The basis functions are also assumed to verify $||\chi_v||_\infty = 1$. With the basis functions assumptions hold, the basis functions are linearly independent.

The online Q-learning defined earlier is combined with the compact state representation using gradient based update. Thus, the update rule in (18) is re-defined accordingly as follows,

$$\vartheta^{t+1} = \vartheta^t + \alpha^t \big(R_t(a,b) + \beta \max_{b' \in B} \vartheta^t \chi^T(a',b') - \vartheta^t \chi^T(a,b)\big)\chi(a,b) \tag{20}$$

The new update rule in (20) is composed of $\vartheta^t = [\{\vartheta_v^t\}_{v=1}^V]$ which is the vector of parameter value over epoch t, a generic temporal difference in the epoch and the gradient. The gradient $(\chi(a,b))$ is defined as the vector of partial derivatives with respect to ϑ^t.

Since the update rule in (20) is performed in a vector basis, it will not converge. Thus, we use ordinary differential equations (ODE) to obtain the necessary conditions for convergence. To proceed, we first introduce the following definitions and assumptions.

Definition 1: we define the matrix Υ as,

$$\Upsilon = \mathbb{E}[\chi^T(a,b)\chi(a,b)] \tag{21}$$

For the parameter vector ϑ and a specific network state $a \in A^+$, we define a vector $\chi(a,\vartheta) = [\{\chi_v(a,b)\}_{v=1}^V]$ where $b \in B_a^\vartheta$. B_a^ϑ is the set of optimal resource allocation actions for state a and it is defined as $B_a^\vartheta\{b \in B_a^\vartheta | b = arg\ \max_{b' \in B} \vartheta \chi^T(a,b')\}$. Now, we define the following a ϑ-dependent matrix:

$$\Upsilon^\vartheta = \mathbb{E}[\chi^T(a,\vartheta)\chi(a,\vartheta)] \tag{22}$$

Both Υ and Υ^ϑ are positive. We introduce the following required assumptions,

Assumption 1: The basis functions $\{\chi_v(a,b)\}_{v=1}^V$ are linearly independent for all $(a,b) \in A^+ \times B$.

Assumption 2: For every $v = (1,2....V)$, $\chi_v(a,b)$ is bounded, which means $\mathbb{E}\{\chi_v^2(a,b)\} < \infty$ and the reward function satisfies $\mathbb{E}\{R^2(a,b)\} < \infty$.

Assumption 3: The learning rate $\{\alpha^t\}$ fulfill the constraint $\sum_{t=1}^\infty \alpha^t = \infty$ and $\sum_{t=1}^\infty (\alpha^t)^2 < \infty$.

Proposition 1: Under *assumptions 1-3* and *Definition 1*, the approximated online Q-learning converges with probability one, if

$$\Upsilon^\vartheta < \Upsilon \quad \forall \vartheta \in \mathbb{R}^V \tag{23}$$

Proof: The convergence of the proposed online Q-learning requires finding stable fixed points of the ODE associated with the update rule in (20), which are found as follows,

$$\vartheta^{\cdot t} = \mathbb{E}[R_t(a,b) + \beta \vartheta^t \chi^T(a',\vartheta^t) - \vartheta^t \chi^T(a,b))\chi(a,b)] \tag{24}$$

We need to find a stable points of the ODE defined in (24) to prove the convergence of the proposed online Q-learning. Thus, we define two trajectories ϑ_1^t and ϑ_2^t of the ODE with different initial conditions and $\vartheta_0^t = \vartheta_1^t - \vartheta_2^t$. Then, we have

$$\frac{\partial \|\vartheta_0^t\|^2}{\partial t} = 2(\vartheta_1^{\cdot t} - \vartheta_2^{\cdot t})(\vartheta_0^t)^T = 2\beta \mathbb{E}\big[\vartheta_1^t \chi^T(a',\vartheta_1^t)\chi(a,b)(\vartheta_0^t)^T - \vartheta_2^t \chi^T(a',\vartheta_2^t)\chi(a,b)(\vartheta_0^t)^T\big] - 2\vartheta_0^t \Upsilon (\vartheta_0^t)^T \tag{25}$$

Using the definition of $\chi(a,\vartheta)$ in *Definition 1*, we can infer the following two inequalities,

$$\vartheta_1^t \chi^T(a',\vartheta_1^t) \leq \vartheta_1^t \chi^T(a',\vartheta_2^t) \tag{26}$$

$$\vartheta_2^t \chi^T(a',\vartheta_2^t) \leq \vartheta_2^t \chi^T(a',\vartheta_1^t) \tag{27}$$

Since the expectation $\mathbb{E}$ in (25) is taken over different resource allocation actions in different states, we can define the two

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2020.3015689, IEEE Transactions on Network Science and Engineering

sets $\Gamma_+ = \{(a,b) \in A^+ \times B | \vartheta_0^t \chi^T(a,b) > 0\}$ and $\Gamma_- \in A^+ \times B - \Gamma_+$. If we substitute (26) and (27) in (25), we get,

$$\frac{\partial \|\vartheta_0^t\|^2}{\partial t} \leq 2\beta \big(\mathbb{E}\big[\vartheta_0^t \chi^T(a', \vartheta_2^t)) \chi(a,b) (\vartheta_0^t)^T \big| \Gamma_+ \big] + \mathbb{E}\big[\vartheta_0^t \chi^T(a', \vartheta_1^t)) \chi(a,b) (\vartheta_0^t)^T \big| \Gamma_- \big] \big) - 2\vartheta_0^t \Upsilon (\vartheta_0^t)^T \quad (28)$$

Holder's inequality [23] is applied to the expectation in (28) as follows,

$$\frac{\partial \|\vartheta_0^t\|^2}{\partial t} \leq 2\beta \Bigg(\sqrt{\mathbb{E}\big[\big(\vartheta_0^t \chi^T(a', \vartheta_2^t)\big)^2 \big| \Gamma_+\big]} \times \sqrt{\mathbb{E}\big[\big(\chi(a,b)(\vartheta_0^t)^T\big)^2 \big| \Gamma_+\big]} + \sqrt{\mathbb{E}\big[\big(\vartheta_0^t \chi^T(a', \vartheta_1^t)\big)^2 \big| \Gamma_-\big]} \times \sqrt{\mathbb{E}\big[\big(\chi(a,b)(\vartheta_0^t)^T\big)^2 \big| \Gamma_-\big]} \Bigg) - 2\vartheta_0^t \Upsilon (\vartheta_0^t)^T$$

$$\leq 2\beta \Bigg(\sqrt{\mathbb{E}\big[\big(\vartheta_0^t \chi^T(a', \vartheta_2^t)\big)^2\big]} \times \sqrt{\mathbb{E}\big[\big(\chi(a,b)(\vartheta_0^t)^T\big)^2 \big| \Gamma_+\big]} + \sqrt{\mathbb{E}\big[\big(\vartheta_0^t \chi^T(a', \vartheta_1^t)\big)^2\big]} \times \sqrt{\mathbb{E}\big[\big(\chi(a,b)(\vartheta_0^t)^T\big)^2 \big| \Gamma_-\big]} \Bigg) - 2\vartheta_0^t \Upsilon (\vartheta_0^t)^T$$

With application of the definition of Υ^{ϑ} in (22), we get,

$$\frac{\partial \|\vartheta_0^t\|^2}{\partial t} \leq 2\beta \sqrt{max\big[\vartheta_0^t \Upsilon^{\vartheta_1} (\vartheta_0^t)^T, \vartheta_0^t \Upsilon^{\vartheta_2} (\vartheta_0^t)^T\big]} \times \sqrt{\mathbb{E}\big[\big(\chi(a,b)(\vartheta_0^t)^T\big)^2\big]} + -2\vartheta_0^t \Upsilon (\vartheta_0^t)^T \quad (29)$$

If the condition in (23) is met, we can indicate that,

$$\frac{\partial \|\vartheta_0^t\|^2}{\partial t} \leq 2\beta \vartheta_0^t \Upsilon (\vartheta_0^t)^T - 2\vartheta_0^t \Upsilon (\vartheta_0^t)^T = (2\beta - 2)\vartheta_0^t \Upsilon (\vartheta_0^t)^T < 0 \quad (30)$$

which means that ϑ_0^t converges to the origin. Therefore, there exists a stable point of the ODE in (24). Thus, the proposed online Q-learning with Q-value approximation converges with probability one. ■

Since there is a stable point ϑ^* of the ODE in (24), we can indicate the following,

$$0 = \mathbb{E}[R(a,b) + \beta \vartheta^t \chi^T(a', \vartheta^t) - \vartheta^t \chi^T(a,b)) \chi(x,y)] \quad (31)$$

Then,

$$\vartheta^* = \mathbb{E}[R_t(a,b) + \beta \vartheta^* \chi^T(a', \vartheta^*)) \chi(a,b)\big] \Upsilon^{-1} \quad (32)$$

As a result, the optimal approximated Q-function confirms that,

$$Q'(a,b,\vartheta^*) = \mathbb{E}[R_t(a,b) + \beta \vartheta^* \chi^T(a', \vartheta^*)) \chi(a,b)\big] \times \Upsilon^{-1} \chi(a,b) \quad (33)$$

for all $(a,b) \in A^+ \times B$.

C. Distributed and Cooperative Approximated Online Q-Learning Resource Allocation

Even with the brief representation of the Q-value in the centralized setting, the number of allocation actions will grow exponentially as the number of IoT devices increases in Edge-IoT. This increase will eventually create a practical scalability challenge as the controller will not be able to cope with such Edge-IoT system with large number of IoT devices with heterogeneous applications. The edge servers can cooperate with each other in learning to enhance the resource allocation. This cooperation facilitates a distributed scheme for resource allocation. In the distributed scheme, edge servers learn in a cooperative multi-agent learning fashion how to make local decisions for resource allocation of IoT devices in their proximities. In this setting, the IoT devices send the resource requests to the edge server in their proximity. The edge servers shares their state information with each others through the controller and learn cooperatively to achieve a common goal of finding joint resource allocation policy that maximizes the IoT applications' utilities and fulfill the constraints in $C1$ to $C3$. The controller acts as a repository in this setting from which the learning agents (edge servers) obtain the online Q-learning MDP (state/action/rewards) of each other. Thus, these servers can exploit them to achieve their actions policy in the distributed learning. This cooperation solves the scalability problem and enhances the decision making for resource allocation. In addition, it optimizes the resource allocation action as it is generated based on the shared information between all the edge servers.

The edge servers take the role of the learning agent in a team Markov game. The game is defined as $\Delta = \{\mathbf{M}, A^+, B, R\}$ with the common objective of finding a resource allocation policy π that maximizes the total expected applications utilities given in (13), where $\mathbf{M}$ is the set of the edge servers. The optimal Q-value $Q^*(s,b)$ for all $(a,b) \in A^+ \times B$, is related to the optimal joint resource allocation policy and captures the team Markov game structure. For each environment state $a \in A^+$, the edge servers play the game $\Delta_a = \{\mathbf{M}, B, Q^*(a,.)\}$ in which $Q^*(a,.)$ is independent. The actions in the team Markov game is generated jointly by the $\mathbf{M}$ independent edge servers in a distributed fashion. The joint resource allocation action b at state a is evaluated as optimal if $Q^*(a,b) \geq Q^*(a,b')$ for all $b' \in B$. The distributed and cooperative resource allocation setting is presented in Fig. 3. Since it is impossible to have a particular state visited infinitely often when the state space is large which is the case in Edge-IoT, we exploit the compact representation model explained in section IV.B. Consequently, we conclude the following proposition.

Proposition 2: For the Markovian game Δ, the distributed multi-agent online Q-learning algorithm converges with probability one if the constraint in *Proposition 1* holds.

Proof: If we consider that each edge server follows an independent resource allocation strategy as the controller in Section IV.B, then, the team Markov game is a discrete time MDP. As a result, the proof of *Proposition 2* is the same as the proof of *Proposition 1* in the centralized setting. ■

To coordinate the multi-agent learning, the following assump-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2020.3015689, IEEE Transactions on Network Science and Engineering

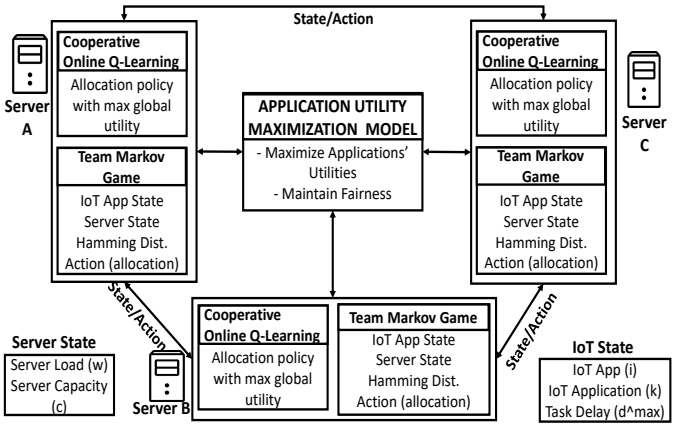

Fig. 3. Distributed resource allocation setting

tions are made,

Assumption 4: The resource allocation policy of different edge servers do not change significantly in similar network states.

Assumption 5: The initial network state $a(t)$ evolves according to irreducible and Harris recurrent Markov chain [24]. Based on *Assumption 4*, it is necessary to check the state similarities in order to exploit the historical allocation actions of each edge server. Therefore, we exploit Hamming distance [25] to measure the similarity between two network states a and a' denoted by $D_H(a, a')$. Each edge server can employ the resource allocation action used by other edge servers for the current network state by using the past actions of these servers. The historical actions up to epoch t can be obtained by the σ-algebra as follows,

$$F(t) = \sigma\big(\{a(n), b(n)\}_{n=1}^{t}, \{R(a(n), b(n))\}_{n=1}^{t-1}\big) \qquad (34)$$

where the information of each experienced state $a(n)$, each performed action $b(n)$ and application utility $R(a(n), b(n))$ can be extracted from the controller. At each epoch t, every edge server checks the Hamming distance between the current state $a(t)$ and state $a(n)$ in $F(t)$. Then, it obtains a sample set $A_F^+(a(t), F(t))$, which is composed of F most recent states from $F(t)$ that minimize the distance $\sum_{f=1}^{F} D_H(a(t), a(n_f)))$. Then, a virtual game is created as $\Gamma_{a(t)} = \{\mathbf{M}, B, E(a(t), .)\}$ for state $a(t)$ at t, where $E(a(t), .)$ is the common reward (utility) that all the servers gain after they select resource allocation action $b \in B$ and is set to 1 if $b = argmax_{b' \in B} Q'(a(t), b', \vartheta^*)$ and 0 otherwise. In addition, $B_j^{"}(a(t))$ is created for each edge server as the set of joint actions that achieve reward of value 1 in state $a(t)$.

The multi-agent approximated online Q-learning is implemented through the following procedure, when $t \leq Z$, all the edge servers select resource allocation actions randomly. from $t = Z + 1$, each edge server j picks l allocation records $B_{j,l}^+(A_F^+(a(t), F(t)))$ from the historical actions that corresponds to $A_F^+(a(t), F(t))$. Note that Z and l are two integers that satisfy $1 \leq l \leq F \leq Z$. Considering the following conditions: (i) there exist a resource allocation action $b = (b_j, b_{-j}) \in B_j^{"}(a(t))$ such that $b'_{-j} = b_{-j}$ for all $b' = (b'_j, b'_{-j}) \in B_{j,l}^+(A_F^+(a(t), F(t)))$; (ii) there exists al least one joint action b that is $b \in B_{j,l}^+(A_F^+(a(t), F(t))) \sqcap B_j^{"}(a(t))$.

Algorithm 1 Distributed online Q-learning algorithm for resource allocation

Require: i, j, c_j, $d_{i,j}^{max}$, $\lambda_{i,j}$, ω_j, p_i
Ensure: s_i for IoT App

```
Initialization of Learning
set t = 1, ϑ_v^V ← 0
evaluate the state a(t)
if (t < Z + 1) then
    Select action b randomly;
    if (C1 to C3 are satisfied ) then
        R_t(a, b) is achieved
    else
        R_t(a, b) = 0
    end if
else
    Update B_j"(a(t)) = {b|E(a(t), b) = 1} for a(t)
    for (exploitation probability 1 − ϵ) do
        Randomly select B_{j,l}^+(A_F^+(a(t), F(t))) out of F actions corresponding to A_F^+(a(t), F(t))
        Calculate E'(a(t), b_j) according to (35) and populate B'_j(a(t))
        if ((i) and (ii) hold) then
            select the most recent action from B_{j,l}^+(A_F^+(a(t), F(t))) ⊓ B_j"(a(t))
        else
            select an action from B'_j(a(t))
        end if
    end for
    for (exploration probability ϵ) do
        select allocation action randomly
    end for
end if
check a(t) → a(t + 1) and R(a(t), b(t))
Update ϑ^t according to (20)
t = t + 1
```

If (i) and (ii) are satisfied, the edge server j selects the allocation action $B_j^{"}(n^*)$, where $n^* = \max_n\{n|b(n) \in Y_F(X_F(x^t, F(t))) \sqcap B_j^{"}(a(t))\}$. Otherwise, the edge server j selects a random action from $B'_j(a(t)) = \{b_j|b_j = arg\max_{b'_j} E'(a(t), b'_j)\}$, where

$$E'(a(t), b_j) = \sum_{b_{-j}} E(a(t), b) \frac{\gamma_j^t(a(t), b_{-j})}{l} \qquad (35)$$

The value in (35) is found using l records that are randomly drawn from F most recent actions, $\gamma_j^t(a(t), b_{-j})$ is the number of times the other servers perform the joint action b_{-j} in state $a(t)$. The distributed multi-agent online Q-learning for resource allocation process in Edge-IoT is illustrated in Algorithm 1. The distributed multi-agent learning for resource allocation converges with probability one to the best joint resource allocation policy with consideration of assumptions 1-5 and $l \leq F/(\varrho_{\Delta_a} + 2)$ for each $a \in A^+$, where ϱ_{Δ_a} is the best response graph of game Δ_a [26].

The convergence of $\{\vartheta^t\}$ to the optimal ϑ^* is demonstrated as a result of *Proposition 2*. Thus, the game $\Gamma_{a(t)}$ based on

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2020.3015689, IEEE Transactions on Network Science and Engineering

ϑ^t will evolve to reach the game Γ_a established based on ϑ^* for $a \in A^+$. Under *Assumptions 4* and *5*, the team stage game $\Delta_{a(t)}$ is decreased to a team stage game with states $A_F^+(a(t), F(t))$. The **M** servers manage resource allocation policy for all $a(t)$ as $l \leq F/(\varrho_{\Delta_{a(t)}} + 2)$ by applying theorem 1 in [26]. This guarantees that the distributed multi-agent resource allocation will converge with probability one.

V. Evaluation and Numerical Results

A. Simulation Setup

The evaluation environment composes of area with dimensions of 10x10 km, where the locations of edge servers and IoT devices are uniformly assigned. A total of 90 edge servers and 900 locations are generated, where each IoT device assumes to hold one of these locations. We consider three applications with various delay requirements in our simulation: gun-shot detection (GSD) (300 devices), Virtual Reality (VR) application (300 devices), and home voice assistant (VA) (300 devices), which are all delay sensitive but with different priorities. Highest priority goes to GSD, then VR application and finally VA and this is reflected on the number of resource requests for each application. The link bandwidth between the IoT devices and edge servers is assumed to have the capacity of 54 Mbps as the traffic of the considered applications is heavy while the link capacity between the edge servers is 100 Mbps. The network delay including transmission and propagation delays are estimated according to the round trip time and link capacity and it is assumed to be between [1,2] ms for the link between IoT devices and edge servers and between [0.5, 1.2] ms for the link between the edge servers. The resource requests length are exponentially distributed with an average length of 80 KB. The processing delay at the edge server is 40 ms on average. The maximum tolerable delay of the application follows a uniform distribution. The service rate $k_{i,j}$ is generated randomly with maximum rate of 200 requests per unit time. The evaluation results consist of applications utilities, fairness in resource allocation, and delay measurement. We consider social welfare with different weights (SW) [18], the three-tier resource allocation scheme proposed in [12] (TRA), DQN scheme in [27], and the resource allocation scheme (ECF) proposed in [14] for performance comparison. In the social welfare maximization scheme, the objective is to maximize the user utility subjected to the capacity constraints of the edge servers. The scheme TRA uses stackelberg game based optimization to allocate resources from edge servers to services subscribers. In ECF framework, cross-layer dynamic stochastic network optimization is exploited to maximize the system utility, based on the Lyapunov stochastic optimization approach. The label (OQL-Cent) indicates our proposed centralized online Q-learning setting while (OQL-Dist) indicates the distributed one. In [27], the authors proposed a DQN based offloading and resource allocation scheme for Edge-IoT system. They exploit deep neural networks for Q-value approximation to account for the dimensionality problem in large scale Edge-IoT environment with a large number of IoT devices.

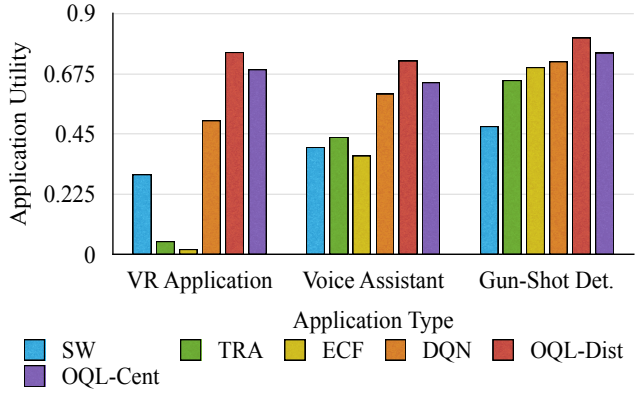

Fig. 4. Application utility for different applications

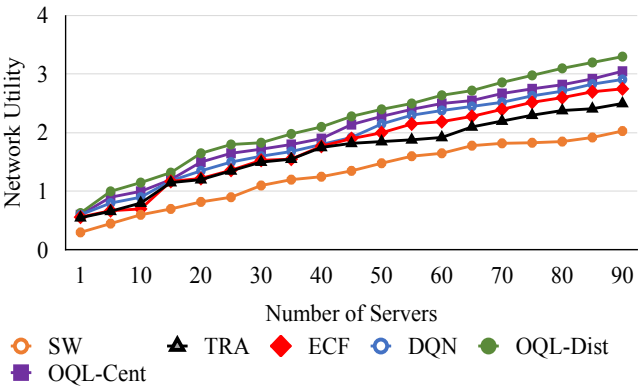

Fig. 5. Network utility against the number of servers

B. Application Utility and Fairness Evaluation

In this section, we demonstrate the capability of the proposed scheme in terms of the achieved application utility, network utility and fairness in resource allocation. Application utility is defined as the maximum gain that the application can achieve from its resource allocation. It measures the satisfaction of the application as it is a function of the number of applications requests that are successfully processed. The request is considered to be successfully processed if the encountered delay including both network and processing delay are less than maximum tolerable delay of the application. Fig. 4 presents the performance comparison among the schemes in terms of application utility achieved for the three considered applications. The network utility which is the cumulative utility for all applications against the number of servers is presented in Fig. 5. The results in Fig. 4 and Fig. 5 show that our scheme outperforms all other schemes in terms of the utility achieved. We notice that SW scheme allocates resources fairly but with low utility. In spite of the comparable utility achieved by TRA, ECF, and DQN schemes in the GSD application, they lack the fairness feature as the voice assistance application gained very low utility. The reason is that these schemes focus on the application that has the highest priority which is the GSD. Fig. 6 compares the envy-freeness (EF) among the schemes where EF =1 means that the allocation is envy-free. The general concept of EF is that every IoT application i feels that its resource share from certain edge server j is at least as good as other applications. Therefore, the application i will not envy other applications. EF is calculated

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2020.3015689, IEEE Transactions on Network Science and Engineering

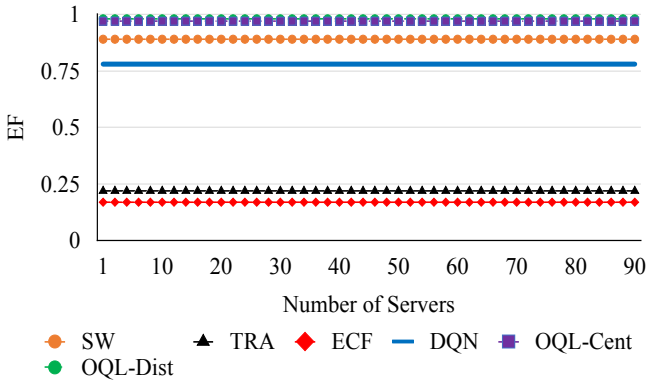

Fig. 6. EF score for resource allocation schemes

as follows,

$$EF = \min_{i,k} \frac{u_i(s_i)/\zeta_i}{u_i(s_k)/\zeta_k} \tag{36}$$

where i and k are the indices of two different IoT applications, u is the utility function, ζ is the maximum utility of the application. We can notice that our scheme in both settings outperforms the TRA, ECF and DQN schemes.

C. Delay Evaluation

In this section, we evaluate the performance of the proposed resource allocation scheme in terms of the encountered average service delays which include both processing and network delay. We evaluate the average service delay recorded by our scheme for the GSD, VR and VA applications as a function of the probability of resources requests from the edge cloud. We compare the performance of the proposed scheme against the one achieved by SW, TRA, ECF, and DQN schemes. Fig. 7, Fig. 8, and Fig. 9 present the average service delay for the GSD, VR and VA applications respectively. The figures show that the delay is reduced as the resources request probability increases because more resources will be available for the applications. In addition, we observe that our proposed scheme outperforms the other schemes as it recorded the lowest service delay for all the applications which highlights the advantages of using online Q-learning to determine the most appropriate resource allocation policy that maximizes the applications utilities. In addition, we demonstrate the capability of our proposed resource allocation scheme to maintain the average delay defined in (1) below certain delay threshold defined according to the application type as in $C1$. Fig. 10 presents the average delay of different applications compared to their corresponding threshold. It is evident that our scheme using both centralized and distributed settings is able to maintain the average applications delays below the specified threshold and consequently fulfills the constraint in $C1$.

D. Applications and Environment Analysis

In this part, the effect of the number of IoT devices and number of edge servers on the achieved network utility is evaluated. Fig. 11 and Fig. 12 show the impact of the number of the IoT devices and the number of edge servers on the achieved utilities for the three applications: GSD, VR and VA following both online Q-learning settings. Note that the

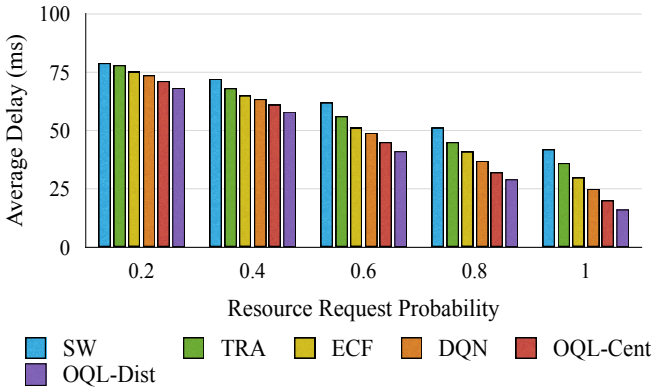

Fig. 7. Average delay for GSD application

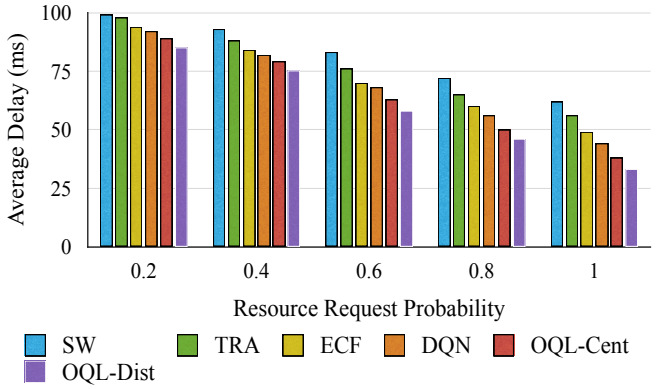

Fig. 8. Average delay for VR application

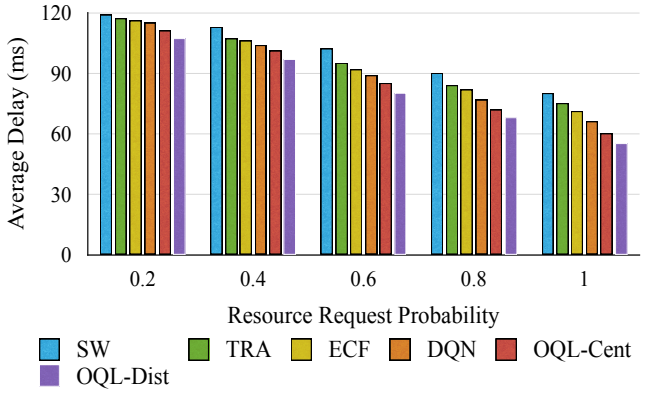

Fig. 9. Average delay for VA application

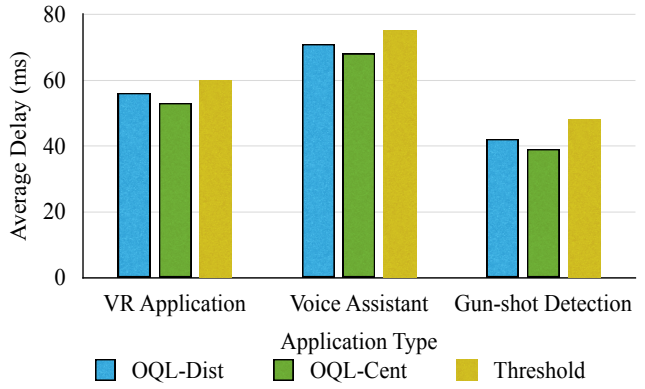

Fig. 10. Average delay for multiple application compared to threshold

ratio of each application is 1/3 of the total number of the IoT devices. We observe that as the number of devices increases, the network utility decreases as the same set of edge servers

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TNSE.2020.3015689, IEEE Transactions on Network Science and Engineering

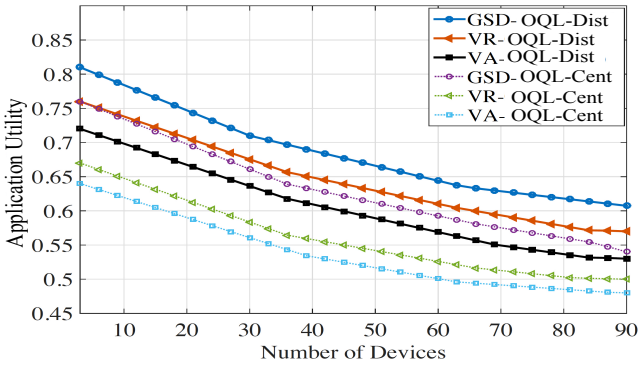

Fig. 11. Application utility with variable number of IoT devices

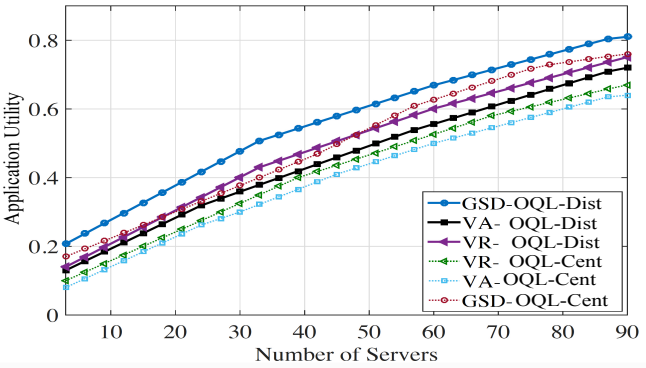

Fig. 12. Application utility with variable number of edge servers

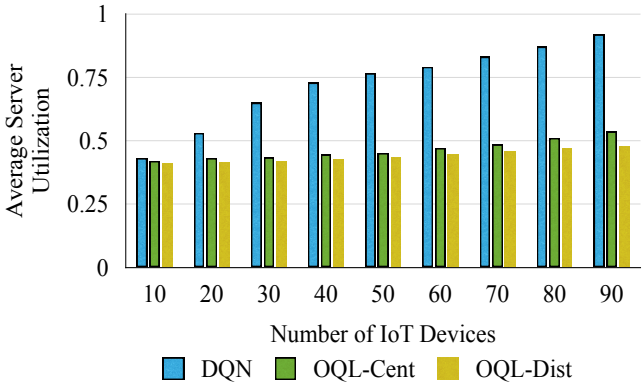

Fig. 13. Average edge server utilization with variable number of IoT devices

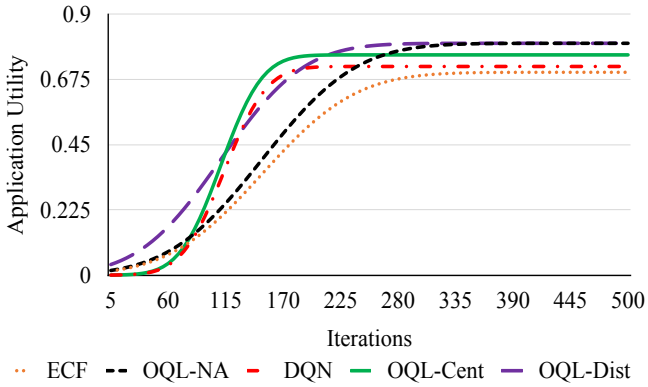

Fig. 14. Resource allocation schemes convergence

is shared among more devices. However, the utility increases significantly as the number of servers increases where more resources will be available.

In addition, we demonstrate the capability of our resource allocation scheme in maintaining the average edge server load balanced and stable regardless of the number of IoT devices that join the system. Fig. 13 shows the normalized average server utilization versus the number of IoT devices. We can notice that our scheme using both approaches keep the utilization of the server steady and balanced. The server utilization is compared with the utilization achieved by the DQN scheme. The DQN scheme measured server utilization deteriorates as the number of IoT devices increases. This evaluation demonstrates that the proposed online Q-leaning scheme maintained the server utilization i.e. the amount of allocated resources below the maximum capacity. Finally, Fig. 14 measures the speed of convergence of our scheme using both centralized and distributed settings. We compare the performance to cooperative online Q-learning with same optimization objective but without Q-value approximation (OQL-NA), DQN, and ECF schemes. It is evident that our scheme achieves a considerable improvement in speed of convergence in comparison to the typical online Q-learning thanks to the approximation of Q-value that reduces that state space. We also notice that the centralized approach achieved better convergence than the distributed one. The reason is that the distributed approach requires information exchange of the historical resource allocation actions among the edge servers which consumes more time. The evaluation in Fig. 14 is executed for the GSD application.

All the presented results demonstrate the potential of the proposed utility based resource allocation in Edge-IoT using approximated online Q-learning. Note that the distributed setting of online Q-learning records better results than the centralized setting. This is expected as the edge servers learn the ultimate resource allocation action in a cooperative manner and in the distributed setting. The cooperative learning allows the edge servers to utilize the controller as a repository to store their individual resource allocation policy. This makes edge servers able to exploit each others allocation policies in similar network scenarios. Hence, it fosters the servers' capability to have better resource allocation actions in comparison with the centralized online Q-learning. Each edge server obtains the state information from the controller and becomes able to play an optimal resource allocation action in a team Markov game by using its historical experience and other servers experience. Moreover, the computational complexity of the centralized online Q-learning increases significantly as a result of the extreme growth in the number of IoT devices in comparison with the complexity of the distributed approach. Thus, the usage of centralized setting might be less preferred especially in large scale Edge-IoT systems. However, the centralized setting has better convergence speed as the distributed setting requires more time to run the team Markov game on multiple servers to reach joint resource allocation policy.

VI. Conclusion

In this paper, we investigated the resource allocation problem in Edge-IoT network, which consists of multiple distributed edge servers and heterogeneous IoT applications. We proposed a novel application utility based model to allocate re-

sources to multiple applications that maximizes the application utility considering many factors such as servers load, capacity, and applications' requirements. The proposed model maintains fairness in resource allocation at maximum level and this was evaluated using the envy-freeness feature. We developed an enhanced centralized online Q-learning scheme for resource allocation in which a dedicated controller is exploited to gather network state information and learns how to achieve an effective resource allocation policy. The developed scheme comprises a unique method to expedite the learning convergence with less computation in comparison with the typical Q-learning through approximation of the Q-value. In addition, we proposed a distributed cooperative online Q-learning approach where each edge server acts as a learning agent. The learning agent takes advantage of other servers' resource allocation policies if it encounters the same state scenario. The distributed online Q-learning improves the resource allocation process and converges to better application utility.

ACKNOWLEDGMENT

The work is supported in part by National Science Foundation (NSF) CNS core grant No. 1909520 and by National Aeronautics and Space Administration (NASA) EPSCoR research grant under No. NNX15AK38A.

REFERENCES

[1] M. Chiang and T. Zhang, "Fog and iot: An overview of research opportunities", *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 854–864, Dec 2016.

[2] Y. Lin and H. Shen, "Cloudfog: Leveraging fog to extend cloud gaming for thin-client mmog with high quality of service", *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 2, pp. 431–445, Feb 2017.

[3] M. Satyanarayanan, "The emergence of edge computing", *Computer*, vol. 50, no. 1, pp. 30–39, Jan 2017.

[4] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges", *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, Oct 2016.

[5] L. Gu, D. Zeng, S. Guo, A. Barnawi, and Y. Xiang, "Cost efficient resource management in fog computing supported medical cyber-physical system", *IEEE Transactions on Emerging Topics in Computing*, vol. 5, no. 1, pp. 108–119, Jan 2017.

[6] Richard S. Sutton and Andrew G. Barto, *Introduction to Reinforcement Learning*, MIT Press, Cambridge, MA, USA, 1st edition, 1998.

[7] X. Lyu, H. Tian, C. Sengul, and P. Zhang, "Multiuser joint task offloading and resource optimization in proximate clouds", *IEEE Transactions on Vehicular Technology*, vol. 66, no. 4, pp. 3435–3447, April 2017.

[8] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption", *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 1171–1181, Dec 2016.

[9] X. Sun and N. Ansari, "Latency aware workload offloading in the cloudlet network", *IEEE Communications Letters*, vol. 21, no. 7, pp. 1481–1484, July 2017.

[10] A. N. Toosi, K. Vanmechelen, K. Ramamohanarao, and R. Buyya, "Revenue maximization with optimal capacity control in infrastructure as a service cloud markets", *IEEE Transactions on Cloud Computing*, vol. 3, no. 3, pp. 261–274, July 2015.

[11] S. F. Abedin, M. G. R. Alam, N. H. Tran, and C. S. Hong, "A fog based system model for cooperative iot node pairing using matching theory", in *2015 17th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, Aug 2015, pp. 309–314.

[12] H. Zhang, Y. Xiao, S. Bu, D. Niyato, F. R. Yu, and Z. Han, "Computing resource allocation in three-tier iot fog networks: A joint optimization approach combining stackelberg game and matching", *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1204–1215, Oct 2017.

[13] S. Li, N. Zhang, S. Lin, L. Kong, A. Katangur, M. K. Khan, M. Ni, and G. Zhu, "Joint admission control and resource allocation in edge computing for internet of things", *IEEE Network*, vol. 32, no. 1, pp. 72–79, Jan 2018.

[14] X. Chen, W. Li, S. Lu, Z. Zhou, and X. Fu, "Efficient resource allocation for on-demand mobile-edge cloud computing", *IEEE Transactions on Vehicular Technology*, vol. 67, no. 9, pp. 8769–8780, Sep. 2018.

[15] Xiaolan Liu, Zhijin Qin, and Yue Gao, "Resource allocation for edge computing in iot networks via reinforcement learning", 03 2019.

[16] J. Wang, L. Zhao, J. Liu, and N. Kato, "Smart resource allocation for mobile edge computing: A deep reinforcement learning approach", *IEEE Transactions on Emerging Topics in Computing*, pp. 1–1, 2019.

[17] S. Chichin, Q. B. Vo, and R. Kowalczyk, "Towards efficient and truthful market mechanisms for double-sided cloud markets", *IEEE Transactions on Services Computing*, vol. 10, no. 1, pp. 37–51, Jan 2017.

[18] Yutao Jiao, Ping Wang, Dusit Niyato, and Zehui Xiong, "Social welfare maximization auction in edge computing resource allocation for mobile blockchain", *CoRR*, vol. abs/1710.10595, 2017.

[19] Yutao Jiao, Ping Wang, Dusit Niyato, and Kongrath Suankaewmanee, "Auction mechanisms in cloud/fog computing resource allocation for public blockchain networks", *CoRR*, vol. abs/1804.09961, 2018.

[20] H. Moulin, *Fair Division and Collective Welfare*, MIT Press, Cambridge, MA, USA, 2003.

[21] C. J. C. H. Watkins and P. Dayan, "Q-learning", *Mach. Learn*, vol. 8, no. 3, pp. 279–292, 1992.

[22] Eduardo Rodrigues Gomes and Ryszard Kowalczyk, "Dynamic analysis of multiagent q-learning with ϵ-greedy exploration", in *Proceedings of the 26th Annual International Conference on Machine Learning*, New York, NY, USA, 2009, ICML '09, pp. 369–376, ACM.

[23] "On nonlinear fractional programming", *Management Science*, vol. 13, no. 7, pp. 492–498, 1967.

[24] Gareth O. Roberts and Jeffrey S. Rosenthal, "Harris recurrence of metropolis-within-gibbs and trans-dimensional markov chains", *The Annals of Applied Probability*, vol. 16, no. 4, pp. 2123–2139, 2006.

[25] L. N. de Castro and F. J. Von Zuben, "Learning and optimization using the clonal selection principle", *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 3, pp. 239–251, Jun 2002.

[26] Y. S. Soh, T. Q. S. Quek, M. Kountouris, and H. Shin, "Energy efficient heterogeneous cellular networks", *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 5, pp. 840–850, May 2013.

[27] X. Chen, H. Zhang, C. Wu, S. Mao, Y. Ji, and M. Bennis, "Performance optimization in mobile-edge computing via deep reinforcement learning", in *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*, 2018, pp. 1–6.

Ismail AlQerm is a postdoctoral research associate in the department of math and computer science at University of Missouri-Saint Louis (UMSL). Ismail received his PhD in computer science from King Abdullah University of Science and Technology (KAUST) in 2017 and was among the recipients of KAUST Provost Award. His research interests include edge computing, resource allocation in IoT networks, developing machine learning techniques for resource allocation in wireless networks, and software defined radio prototypes. He is a member of IEEE and ACM.

Jianli Pan is currently an Assistant Professor in the Department of Mathematics and Computer Science at the University of Missouri, St. Louis, MO USA. He obtained his MS. and Ph.D. degrees from the Department of Computer Science and Engineering of Washington University in St. Louis, USA. He also holds a M.S. degree in Information Engineering from Beijing University of Posts and Telecommunications (BUPT), China. He is an associate editor for both IEEE Communication Magazine and IEEE Access. His current research interests include Internet of Things (IoT), edge computing, machine learning, and cybersecurity.