Age of Information-Aware Resource Management in UAV-Assisted Mobile-Edge Computing Systems

Xianfu Chen, Celimuge Wu, Tao Chen, Zhi Liu, Mehdi Bennis, and Yusheng Ji

***Abstract*—This paper investigates the problem of age of information (AoI)-aware resource awareness in an unmanned aerial vehicle (UAV)-assisted mobile-edge computing (MEC) system, which is deployed by an infrastructure provider (InP). A service provider leases resources from the InP to serve the mobile users (MUs) with sporadic computation requests. Due to the limited number of channels and the finite shared I/O resource of the UAV, the MUs compete to schedule local and remote task computations in accordance with the observations of system dynamics. The aim of each MU is to selfishly maximize the expected long-term computation performance. We formulate the non-cooperative interactions among the MUs as a stochastic game. To approach the Nash equilibrium solutions, we propose a novel online deep reinforcement learning (DRL) scheme, which enables each MU to behave using its local conjectures only. The DRL scheme employs two separate deep Q-networks to approximate the Q-factor and the post-decision Q-factor for each MU. Numerical experiments show the potentials of the online DRL scheme in balancing the tradeoff between AoI and energy consumption.**

I. Introduction

Strategic computation offloading in mobile-edge computing (MEC) not only greatly improves the computation Quality-of-Experience (QoE) and Quality-of-Service (QoS) for a mobile users (MU), but also augments the capability of the mobile device for running a variety of resource-demanding applications. Recent years have witnessed a large body of related research [1, and references therein]. Offloading a computation task to the edge server involves wireless transmissions, which encounter the environmental dynamics. Particularly, the time-varying channel qualities due to the MU mobility constrain the computation performance [2]. Because of among others, the low deployment cost, the flexibility and the line-of-sight (LOS) connections, unmanned aerial vehicles (UAVs) are expected to play an important role in advancing the network-generation wireless networks [3]. Leveraging UAVs in a MEC system has shown substantial impacts. In [4], Hu et al. put forward an alternating algorithm to minimize the weighted sum energy consumption for a UAV-assisted MEC system. In [5], Li et al. adopted the Dinkelbach algorithm and the successive convex approximation technique to maximize UAV energy efficiency. However, most existing literature is based on finite time and cannot characterize the long-term performance.

This paper focuses on a UAV-assisted cellular network as in [6], which can be deployed by an infrastructure provider (InP). In addition to the traditional communication service, the three-dimensional system brings computation capability at the edge as well, where the UAV is implemented as a complementary computing server flying in the air [3]. Such a system opens up a new business model, following which a third-party service provider (SP) leases resources from the InP to serve the subscribed MUs with computation requests [2]. Specifically, the resource orchestrator (RO) of the SP manages a limited number of channels, which provide the MUs with access to the UAV-assisted MEC system. The channel allocation is regulated via a Vickrey-Clarke-Groves (VCG) pricing mechanism [7]. Consequently, a MU is able to not only process a computation task locally, but also offload the task for remote execution. At the UAV, the tasks are co-executed by creating isolated virtual machines (VMs) [8], while sharing the same physical UAV platform causes I/O interference and hence leads to computation rate reduction for each VM.

Under this context, each MU competes to schedule local and remote task computations with the awareness of system dynamics. The aim is to maximize its own expected long-term computation performance. In practice, one critical concern is the knowledge freshness from the computation outcomes [9]. A key metric for capturing the freshness is the age of information (AoI) [9], [10]. By definition, AoI is the amount of time elapsed since the outcome of the most recently scheduled computation was received. We formulate the non-cooperative interactions among the MUs as a stochastic game under a multi-agent Markov decision process (MDP). To avoid any information exchange, we propose that each MU behaves independently with the local conjectures, transforming the problem into a single-agent MDP. We develop a novel online deep reinforcement learning (DRL) scheme to deal with the huge local state space. The DRL scheme maintains two separate deep Q-networks (DQNs) [11] to approximate, respectively, the Q-factor and the post-decision Q-factor for each MU.

II. System Descriptions

We assume that an InP deploys a three-dimensional cellular network, where the terrestrial radio access network (RAN) provides computation service with the help of a UAV flying at a fixed altitude of H. A set $\mathcal{B} = \{1, 2, \cdots, B\}$ of base stations (BSs) are connected via wired backhaul to the resource-rich server at the edge, while the UAV is integrated with a parallel

X. Chen and T. Chen are with the VTT Technical Research Centre of Finland, Finland (e-mail: {xianfu.chen, tao.chen}@vtt.fi). C. Wu is with the Graduate School of Informatics and Engineering, University of Electro-Communications, Tokyo, Japan (e-mail: clmg@is.uec.ac.jp). Z. Liu is with the Department of Mathematical and Systems Engineering, Shizuoka University, Japan (e-mail: liu@ieee.org). M. Bennis is with the Centre for Wireless Communications, University of Oulu, Finland (e-mail: mehdi.bennis@oulu.fi). Y. Ji is with the Information Systems Architecture Research Division, National Institute of Informatics, Tokyo, Japan (e-mail: kei@nii.ac.jp).

computing server. A SP serves a set $\mathcal{K}$ of MUs with sporadic computation requests. We use a finite set $\mathcal{L}$ of locations to denote both the terrestrial service region covered by the RAN and the region of the UAV mapped vertically from the air to the ground. Let $\mathcal{L}_b$ denote the locations covered by a BS $b \in \mathcal{B}$. For any two BSs b and $b' \in \mathcal{B} \setminus \{b\}$, we have $\mathcal{L}_b \cap \mathcal{L}_{b'} = \emptyset$. Then $\mathcal{L} = \cup_{b\in\mathcal{B}}\mathcal{L}_b$. The geographical topology of the BSs is represented a two-tuple graph $\langle \mathcal{B}, \mathcal{E}\rangle$, where $\mathcal{E} = \{e_{b,b'} : b, b' \in \mathcal{B}, b \neq b'\}$ with each $e_{b,b'}$ being equal to 1 if BSs b and b' are neighbours, and 0, otherwise. The system operates over discrete decision epochs, each of which is with equal duration δ and indexed by $j \in \mathbb{N}_+$.

We assume that the UAV and the MUs move at the same speed following a Markov mobility model [2]. Let $L^j_{(\mathrm{v})} \in \mathcal{L}$ and $L^j_{(\mathrm{m}),k} \in \mathcal{L}$ denote, respectively, the mapped terrestrial location of the UAV and the location of each MU $k \in \mathcal{K}$ during a decision epoch j. The computation arrivals at the MUs are assumed to be independent Bernoulli random variables with a common parameter $\lambda \in [0, 1]$. Each MU k employs a pre-processing buffer to temporarily store a computation task. It is reasonable for an incoming task with newer arrival time to replace an old task in the pre-processing buffer since a newer computation task is always with fresher information. We assume that a computation task is composed of $D_{(\max)}$ input data packets and each packet contains μ data bits. We let ϑ represent the number of CPU cycles required to accomplish one bit of a computation task. A computation task can be either computed locally at the mobile device or executed remotely. Specifically, let $X^j_k \in \mathcal{X} = \{0, 1, 2, 3\}$ denote the computation offloading decision of MU k at each epoch j, where $X^j_k = 1$, $X^j_k = 2$ and $X^j_k = 3$ indicate that the task in the pre-processing buffer is scheduled to be computed by the local CPU and offloaded to the edge server and the UAV, respectively, while $X^j_k = 0$ means that the task is not scheduled for computation.

The RO manages a set $\mathcal{C}$ of channels with bandwidth η. To upload the input data packets of a scheduled computation task for remote execution, the MUs compete for the limited channels using the VCG mechanism. At the beginning of each epoch j, each MU $k \in \mathcal{K}$ submits to the RO an auction bid $\boldsymbol{\beta}^j_k = (\nu^j_k, \mathbf{N}^j_k)$, where ν^j_k is the true valuation over $\mathbf{N}^j_k = (N^j_{(\mathrm{s}),k}, N^j_{(\mathrm{v}),k})$ with $N^j_{(\mathrm{s}),k}$ and $N^j_{(\mathrm{v}),k}$ being the numbers of demanded channels for data transmissions to the edge server and the UAV. Let $\boldsymbol{\rho}^j_k = (\rho^j_{k,c} : c \in \mathcal{C})$ be the channel allocation for MU k at epoch j, where $\rho^j_{k,c}$ equals 1 if MU k is assigned a channel $c \in \mathcal{C}$ and 0, otherwise. We take into account

$$\left(\sum_{k\in\mathcal{K}^j_{(\mathrm{s}),b}} \rho^j_{k,c}\right) \cdot \left(\sum_{k\in\mathcal{K}^j_{(\mathrm{s}),b'}} \rho^j_{k,c}\right) = 0, \quad \text{if } e_{b,b'} = 1, \forall e_{b,b'} \in \mathcal{E}, \forall c \in \mathcal{C}; \tag{1}$$

$$\left(\sum_{k\in\bigcup_{b\in\mathcal{B}}\mathcal{K}^j_{(\mathrm{s}),b}} \rho^j_{k,c}\right) \cdot \left(\sum_{k\in\mathcal{K}^j_{(\mathrm{v})}} \rho^j_{k,c}\right) = 0, \forall c \in \mathcal{C}; \tag{2}$$

$$\sum_{k\in\mathcal{K}^j_{(\mathrm{s}),b}} \rho^j_{k,c} \leq 1, \forall b \in \mathcal{B}, \forall c \in \mathcal{C}; \tag{3}$$

$$\sum_{k\in\mathcal{K}^j_{(\mathrm{v})}} \rho^j_{k,c} \leq 1, \forall c \in \mathcal{C}; \tag{4}$$

$$\sum_{c\in\mathcal{C}} \rho^j_{k,c} \leq 1, \forall k \in \mathcal{K}, \tag{5}$$

for the centralized channel allocation, where $\mathcal{K}^j_{(\mathrm{s}),b} = \{k : k \in \mathcal{K}, L^j_{(\mathrm{m}),k} \in \mathcal{L}_b, N^j_{(\mathrm{s}),k} > 0\}$, $\forall b \in \mathcal{B}$, while $\mathcal{K}^j_{(\mathrm{v})} = \{k : k \in \mathcal{K}, N^j_{(\mathrm{v}),k} > 0\}$. We denote $\boldsymbol{\phi}^j = (\phi^j_k : k \in \mathcal{K})$ as the winner determination, where ϕ^j_k equals 1 if MU k wins and otherwise 0. The RO calculates $\boldsymbol{\phi}^j$ according to

$$\begin{aligned} \boldsymbol{\phi}^j = \arg\max_{\boldsymbol{\phi}} & \sum_{k\in\mathcal{K}} \phi_k \cdot \nu^j_k \\ \text{s.t.} \quad & \text{constraints (1), (2), (3), (4) and (5);} \\ & \sum_{k\in\mathcal{K}^j_{(\mathrm{s}),b}} \varphi^j_k = \phi_k \cdot N^j_{(\mathrm{s}),k}, \forall b \in \mathcal{B}, \forall k \in \mathcal{K}; \\ & \sum_{k\in\mathcal{K}^j_{(\mathrm{v})}} \varphi^j_k = \phi_k \cdot N^j_{(\mathrm{v}),k}, \forall k \in \mathcal{K}, \end{aligned} \tag{6}$$

where $\boldsymbol{\phi} = (\phi_k \in \{0, 1\} : k \in \mathcal{K})$ and $\varphi^j_k = \sum_{c\in\mathcal{C}} \rho^j_{k,c}$. We also rewrite φ^j_k as $\varphi_k(\boldsymbol{\beta}^j)$, where $\boldsymbol{\beta}^j = (\boldsymbol{\beta}^j_k, \boldsymbol{\beta}^j_{-k})$ with $-k$ denoting all the other MUs without the presence of MU k. The payment for MU k to the SP is calculated as

$$\tau^j_k = \max_{\boldsymbol{\phi}_{-k}} \sum_{\kappa\in\mathcal{K}\setminus\{k\}} \phi_\kappa \cdot \nu^j_\kappa - \sum_{\kappa\in\mathcal{K}\setminus\{k\}} \phi^j_\kappa \cdot \nu^j_\kappa, \tag{7}$$

which is incurred from the channel access.

III. Computation and AoI Models

Let $T^j_k \in \mathbb{N}$ be the arrival epoch index of the computation task waiting in the pre-processing buffer of a MU $k \in \mathcal{K}$ at the beginning of an epoch j. We set $T^j_k = 0$ if the pre-processing buffer is empty. When $X^j_k = 1$ for MU k at epoch j, the number of required epochs is calculated as $\Delta = \lceil (D_{(\max)} \cdot \mu \cdot \vartheta)/(\delta \cdot \varrho)\rceil$, where $\lceil\cdot\rceil$ means the ceiling function and ϱ is the local CPU frequency. We describe by $W^j_{(\mathrm{m}),k} \in \{0, 1, \cdots, \Delta\}$ the local CPU state of MU k at the beginning of epoch j, which is the number of remaining epochs to finish the task. In particular, $W^j_{(\mathrm{m}),k} = 0$ indicates that the local CPU is idle and is available for a new task from epoch j. The local CPU energy consumption during epoch j is given by

$$F^j_{(\mathrm{m}),k} = \begin{cases} 0, & \text{for } W^j_{(\mathrm{m}),k} = 0; \\ \varsigma \cdot \left(D_{(\max)} \cdot \mu \cdot \vartheta - (\Delta - 1) \cdot \delta \cdot \varrho\right) \cdot (\varrho)^2, & \text{for } W^j_{(\mathrm{m}),k} = 1; \\ \varsigma \cdot \delta \cdot (\varrho)^3, & \text{for } W^j_{(\mathrm{m}),k} > 1, \end{cases} \tag{8}$$

where ς is the effective switched capacitance [12].

For remote execution, a MU has to be associated to the terrestrial RAN or with the UAV until the task is accomplished.

Let $I_k^j \in \mathcal{B} \cup \{B+1\}$ be the association state of each MU $k \in \mathcal{K}$ at the beginning of epoch j, namely, $I_k^j = b \in \mathcal{B}$ if MU k is associated with a BS b and if MU k is associated with the UAV, $I_k^j = B+1$. When $I_k^{j+1} \neq I_k^j, \forall j$, a handover is triggered [13]. Suppose that the energy consumed during one handover is negligible but the handover delay is ξ. The transmission time of MU k then is reduced to $\tilde{\delta}_k^j = \delta - \xi \cdot \mathbf{1}_{\{I_k^{j+1} \neq I_k^j\}}$, where $\mathbf{1}_{\{i\}}$ is an indicator function. Let $D_k^j \in \mathcal{D} = \{0, 1, \cdots, D_{(\max)}\}$ denote the local transmitter state of MU k at the beginning of each epoch j, which is the number of input data packets at the transmitter. Let R_k^j be the number of packets to be uploaded during epoch j, the transmitter state evolves to $D_k^{j+1} = D_k^j - \varphi_k^j \cdot R_k^j$. During epoch j, MU k experiences the channel gains $G_{b,k}^j = g_{(\mathrm{s})}(L_{(\mathrm{m}),k}^j)$ for the link between MU k and each BS b as well as $G_{(\mathrm{v}),k}^j = g_{(\mathrm{v})}(L_{(\mathrm{m}),k}^j, L_{(\mathrm{v})}^j)$ for the link between MU k and the UAV. Note that $0 \leq R_k^j \leq \min\{D_k^j, R_{(\max),k}^j\}$, where $R_{(\max),k}^j$ is jointly determined by the channel gain during epoch j, the transmission time and the maximum transmit power $P_{(\max)}$ at the MUs.

At the beginning of an epoch j, if $X_k^j = 2$ for a MU $k \in \mathcal{K}$, all input data packets need to be offloaded during subsequent epochs via the RAN. When $L_{(\mathrm{m}),k}^j \in \mathcal{L}_b$, $b \in \mathcal{B}$, the energy consumption for the transmitting $\varphi_k^j \cdot R_k^j$ packets is

$$F_{(\mathrm{s}),k}^j = \frac{\tilde{\delta}_k^j \cdot \eta \cdot \sigma^2}{G_{b,k}^j} \cdot \left(2^{\frac{\varphi_k^j \cdot \left(\mu \cdot R_k^j\right)}{\eta \cdot \tilde{\delta}_k^j}} - 1 \right), \tag{9}$$

where σ^2 is the noise power spectral density. This paper assumes rich computation resource at the edge server and thus, the task execution delay is ignored. Moreover, we assume that the time for sending the computation outcome back to the MU is negligible, due to the fact that the outcome is in general much smaller than the input data packets [14].

When all the input data packets of a computation task are received up to an epoch, the UAV starts to execute by creating a VM for the MU from the next epoch [8]. If $X_k^j = 3$ at a MU $k \in \mathcal{K}$, the energy consumed for the transmissions of $\varphi_k^j \cdot R_k^j$ input data packets turns to be

$$F_{(\mathrm{v}),k}^j = \frac{\tilde{\delta}_k^j \cdot \eta \cdot \sigma^2}{G_{(\mathrm{v}),k}^j} \cdot \left(2^{\frac{\varphi_k^j \cdot \left(\mu \cdot R_k^j\right)}{\eta \cdot \tilde{\delta}_k^j}} - 1 \right). \tag{10}$$

Let $\breve{\mathcal{K}}_{(\mathrm{v})}^j \subseteq \mathcal{K}_{(\mathrm{v})}^j$ represent the set of MUs, whose computation tasks are being simultaneously executed by the UAV during an epoch j. Let χ_0 be the computation service rate at the UAV given that the task is run in isolation, the degraded rate of each MU $k \in \breve{\mathcal{K}}_{(\mathrm{v})}^j$ is modeled as $\chi^j = \chi_0 \cdot (1+\varepsilon)^{1-|\breve{\mathcal{K}}_{(\mathrm{v})}^j|}$, where $|\cdot|$ denotes the cardinality of a set and $\varepsilon \in \mathbb{R}_+$ is a reduction factor. The remote processing state of MU k can be updated by $W_{(\mathrm{v}),k}^{j+1} = \max\{W_{(\mathrm{v}),k}^j - \chi^j \cdot \delta, 0\}$, where $W_{(\mathrm{v}),k}^j$ quantifies the amount of input data bits remaining at the UAV at the beginning of an epoch j.

To depict the knowledge freshness for each MU $k \in \mathcal{K}$ from computation, we define the AoI as the difference between the current time and the arrival time of the latest task, the outcome of which is received. Let A_k^j denote the AoI of MU k at each epoch j. The AoI evolution can be analysed as follows.

1) When there is no computation outcome received at MU k, $A_k^{j+1} = A_k^j + \delta$.

2) When MU k receives only one computation outcome,

$$A_k^{j+1} = \begin{cases} \left(j - T_{(\mathrm{m}),k}^j - \Delta + 1\right) \cdot \delta + \dfrac{D_{(\max)} \cdot \mu \cdot \vartheta}{\varrho}, \\ \quad \text{for } W_{(\mathrm{m}),k}^j = 1, D_k^j = 0 \text{ and } W_{(\mathrm{v}),k}^j = 0; \\ \left(j - T_{(\mathrm{s}),k}^j + 1\right) \cdot \delta, \\ \quad \text{for } W_{(\mathrm{m}),k}^j = 0, D_k^j > 0 \text{ and } W_{(\mathrm{v}),k}^j = 0; \\ \left(j - T_{(\mathrm{v}),k}^j\right) \cdot \delta + \dfrac{W_{(\mathrm{v}),k}^j}{\chi^j}, \\ \quad \text{for } W_{(\mathrm{m}),k}^j = 0, D_k^j = 0 \text{ and } W_{(\mathrm{v}),k}^j > 0, \end{cases} \tag{11}$$

where $T_{(\mathrm{m}),k}^j$, $T_{(\mathrm{s}),k}^j$ and $T_{(\mathrm{v}),k}^j$ are the arrival epoch indices of the task processed at the local CPU, the edge server and the UAV.

3) When two computation outcomes arrive at MU k,

$$A_k^{j+1} = \begin{cases} \left(j - T_{(\mathrm{s}),k}^j + 1\right) \cdot \delta, \\ \quad \text{for } D_k^j > 0, W_{(\mathrm{v}),k}^j = 0 \text{ and } T_{(\mathrm{s}),k}^j > T_{(\mathrm{m}),k}^j; \\ \left(j - T_{(\mathrm{v}),k}^j\right) \cdot \delta + \dfrac{W_{(\mathrm{v}),k}^j}{\chi^j}, \\ \quad \text{for } D_k^j = 0, W_{(\mathrm{v}),k}^j > 0 \text{ and } T_{(\mathrm{v}),k}^j > T_{(\mathrm{m}),k}^j; \\ \left(j - T_{(\mathrm{m}),k}^j - \Delta + 1\right) \cdot \delta + \dfrac{D_{(\max)} \cdot \mu \cdot \vartheta}{\varrho}, \\ \quad \text{otherwise.} \end{cases} \tag{12}$$

By default, the AoI is initialized as $A_k^1 = 0$ and up-limited by $A_{(\max)}$ for each MU k.

IV. Stochastic Game Formulation

During each decision epoch j, the local state of each MU $k \in \mathcal{K}$ can be described by $\mathbf{S}_k^j = (L_{(\mathrm{v})}^j, L_{(\mathrm{m}),k}^j, \mathbf{1}_{\{T_k^j > 0\}}, I_k^j, W_{(\mathrm{m}),k}^j, W_{(\mathrm{v}),k}^j, D_k^j, A_k^j) \in \mathcal{S}$. $\mathbf{S}^j = (\mathbf{S}_k^j, \mathbf{S}_{-k}^j) \in \mathcal{S}^{|\mathcal{K}|}$ characterizes the global system state. Let $\boldsymbol{\pi}_k = (\pi_{(\mathrm{c}),k}, \pi_{(\mathrm{t}),k}, \pi_{(\mathrm{p}),k})$ denote the control policy of MU k, where $\pi_{(\mathrm{c}),k}$, $\pi_{(\mathrm{t}),k}$ and $\pi_{(\mathrm{p}),k}$ are the channel auction, the computation offloading and the packet scheduling policies. The joint control policy of all MUs can be given by $\boldsymbol{\pi} = (\boldsymbol{\pi}_k, \boldsymbol{\pi}_{-k})$. With the observation of $\mathbf{S}^j$ at epoch j, MU k makes decisions following $\boldsymbol{\pi}_k(\mathbf{S}^j) = (\pi_{(\mathrm{c}),k}(\mathbf{S}^j), \pi_{(\mathrm{t}),k}(\mathbf{S}_k^j), \pi_{(\mathrm{p}),k}(\mathbf{S}_k^j)) = (\boldsymbol{\beta}_k^j, X_k^j, R_k^j)$. We define an immediate payoff

$$\ell_k\left(\mathbf{S}^j, \left(\varphi_k^j, X_k^j, R_k^j\right)\right) = u_k\left(\mathbf{S}^j, \left(\varphi_k^j, X_k^j, R_k^j\right)\right) - \tau_k^j, \tag{13}$$

where $u_k(\mathbf{S}^j, (\varphi_k^j, X_k^j, R_k^j)) = \varpi_k \cdot \exp(-A_k^j) + \omega_k \cdot \exp(-F_k^j)$ and $\varphi_k^j = \varphi_k(\boldsymbol{\pi}_{(\mathrm{c})}(\mathbf{S}^j))$ with $\boldsymbol{\pi}_{(\mathrm{c})} = (\pi_{(\mathrm{c}),k}, \boldsymbol{\pi}_{(\mathrm{c}),-k})$. In the utility function $u_k(\mathbf{S}^j, (\varphi_k^j, X_k^j, R_k^j))$, $F_k^j = F_{(\mathrm{m}),k}^j + F_{(\mathrm{s}),k}^j +$

$F^{j}_{(\mathrm{v}),k}$ is the total local energy consumption, while $\varpi_k \in \mathbb{R}_+$ and $\omega_k \in \mathbb{R}_+$ are the weighting constants.

Each MU $k \in \mathcal{K}$ aims to device a best-response control policy $\boldsymbol{\pi}^*_k = (\pi^*_{(\mathrm{c}),k}, \pi^*_{(\mathrm{t}),k}, \pi^*_{(\mathrm{p}),k})$ such that

$$\boldsymbol{\pi}^*_k = \arg\max_{\boldsymbol{\pi}_k} V_k(\mathbf{S}, \boldsymbol{\pi}), \tag{14}$$

for any global system state $\mathbf{S} = (\mathbf{S}_k = (L_{(\mathrm{v})}, L_{(\mathrm{m}),k}, \mathbf{1}_{\{T_k>0\}}, I_k, W_{(\mathrm{m}),k}, W_{(\mathrm{v}),k}, D_k, A_k) : k \in \mathcal{K}) \in \mathcal{S}^{|\mathcal{K}|}$, where

$$V_k(\mathbf{S}, \boldsymbol{\pi}) = \tag{15}$$
$$(1-\gamma) \cdot \mathsf{E}_{\boldsymbol{\pi}}\left[\sum_{j=1}^{\infty} (\gamma)^{j-1} \cdot \ell_k\Big(\mathbf{S}^j, \Big(\varphi^j_k, X^j_k, R^j_k\Big)\Big) \,|\mathbf{S}^1 = \mathbf{S}\right].$$

Herein, $\gamma \in [0, 1)$ is the discount factor and $V_k(\mathbf{S}, \boldsymbol{\pi})$ is also termed as the state-value function of $\mathbf{S}$ under $\boldsymbol{\pi}$ [15]. Due to the limited number of channels, the shared I/O resource at the UAV and the stochastic nature of the system, we formulate the interactions among the competing MUs over the infinite time-horizon as a non-cooperative stochastic game. A Nash equilibrium (NE) describes the rational behaviours of the MUs in a stochastic game. Specifically, a NE is a tuple of control policies $\langle \boldsymbol{\pi}^*_k : k \in \mathcal{K} \rangle$, where $\boldsymbol{\pi}^*_k$ is the best response to $\boldsymbol{\pi}^*_{-k}$ [16]. For brevity, define $V_k(\mathbf{S}) = V_k(\mathbf{S}, \boldsymbol{\pi}^*_k, \boldsymbol{\pi}^*_{-k})$ as the optimal state-value function. It can be easily found that $V_k(\mathbf{S}, \boldsymbol{\pi})$ of a MU k depends on information of not only the global system states across the time-horizon but also the joint control policy $\boldsymbol{\pi}$. In other words, the decision makings from all MUs are coupled in the stochastic game.

V. DRL with Local Conjectures

We develop an online DRL scheme for the MUs to approach the NE control policy with only local information.

A. Local Conjectures

In the stochastic game, it is challenging for each MU $k \in \mathcal{K}$ to obtain the private information from other MUs. On the other hand, the coupling of the decision makings exists in the channel auction and the remote task execution at the UAV. From the viewpoint of MU k, the payment τ^j_k and the computation service rate χ^j at an epoch j are realized under $\mathbf{S}^j_{-k}$. We allow a MU k to conjecture $\mathbf{S}^{j+1}$ during the next epoch $j+1$ as $\widehat{\mathbf{S}}^{j+1}_k = (\mathbf{S}^{j+1}_k, \mathbf{O}^{j+1}_k)$, where $\mathbf{O}^{j+1}_k = (\tau^j_k, \chi^j) \in \mathcal{O}_k$. Now we transform (15) into

$$V_k\Big(\widehat{\mathbf{S}}_k, \boldsymbol{\pi}\Big) = \tag{16}$$
$$(1-\gamma) \cdot \mathsf{E}_{\boldsymbol{\pi}}\left[\sum_{j=1}^{\infty} (\gamma)^{j-1} \cdot \ell_k\Big(\mathbf{S}^j, \Big(\varphi^j_k, X^j_k, R^j_k\Big)\Big) \,|\widehat{\mathbf{S}}^1_k = \widehat{\mathbf{S}}_k\right],$$

where $\widehat{\mathbf{S}}_k = (\mathbf{S}_k, \mathbf{O}_k) \in \widehat{\mathcal{S}}_k = \mathcal{S} \times \mathcal{O}_k$ with $\mathbf{O}_k$ being the initial local conjecture of $\mathbf{S}_{-k}$, while $\boldsymbol{\pi}$ hereinafter refers to the conjecture based joint control policy. Each MU k then switches to maximize $V_k(\widehat{\mathbf{S}}_k, \boldsymbol{\pi})$, $\forall \widehat{\mathbf{S}}_k \in \widehat{\mathcal{S}}_k$, which is basically a single-agent MDP. With a slight abuse of notation, we let $V_k(\widehat{\mathbf{S}}_k) = V_k(\widehat{\mathbf{S}}_k, \boldsymbol{\pi}^*)$, $\forall k \in \mathcal{K}$, where $\boldsymbol{\pi}^*$ is the best-response control policy profile of all MUs with local conjectures and the Bellman's optimality equation is given by (17).

When all MUs follow the best-response control policy profile $\boldsymbol{\pi}^*$ based on the local conjectures, each MU $k \in \mathcal{K}$ announces at the beginning of a current decision epoch to the RO the channel demands

$$N_{(\mathrm{s}),k} = z_k \cdot \mathbf{1}_{\left\{\pi^*_{(\mathrm{t}),k}(\mathbf{S}_k)=2\right\}}, \tag{18}$$
$$N_{(\mathrm{v}),k} = z_k \cdot \mathbf{1}_{\left\{\pi^*_{(\mathrm{t}),k}(\mathbf{S}_k)=3\right\}}, \tag{19}$$

together with the true valuation being specified as

$$\nu_k = u_k\Big(\mathbf{S}^j, \Big(z_k, \pi^*_{(\mathrm{t}),k}(\mathbf{S}_k), \pi^*_{(\mathrm{p}),k}(\mathbf{S}_k)\Big)\Big) + \frac{\gamma}{1-\gamma} \cdot \tag{20}$$
$$\sum_{\widehat{\mathbf{S}}'_k \in \widehat{\mathcal{S}}_k} \mathbb{P}\Big(\widehat{\mathbf{S}}'_k | \widehat{\mathbf{S}}_k, \Big(z_k, \pi^*_{(\mathrm{t}),k}(\mathbf{S}_k), \pi^*_{(\mathrm{p}),k}(\mathbf{S}_k)\Big)\Big) \cdot V_k\Big(\widehat{\mathbf{S}}'_k\Big),$$

where z_k denotes the preference of winning one channel and $\widehat{\mathbf{S}}'_k$ is the subsequent local state. It remains challenging to come up with an optimal bid without the local system dynamics statistics and the structure of the payment function.

B. Proposed DRL Scheme

To make the calculation of an auction bid feasible, we introduce a local post-decision state [17] for the MUs. At each current epoch, the local post-decision state of a MU $k \in \mathcal{K}$ is defined as $\widetilde{\mathbf{S}}_k = (L_{(\mathrm{v})}, L_{(\mathrm{m}),k}, \mathbf{1}_{\{T_k>0\}}, I_k, W_{(\mathrm{m}),k}, W_{(\mathrm{v}),k}, \widetilde{D}_k, A_k, \mathbf{O}_k) \in \widehat{\mathcal{S}}_k$ by letting $\widetilde{D}_k = D_k - \varphi_k(\boldsymbol{\beta}) \cdot R_k$, where $\boldsymbol{\beta} = (\beta_k, \boldsymbol{\beta}_{-k})$. For each MU k, we define the right-hand-side of (17) as a Q-factor, namely,

$$Q_k\Big(\widehat{\mathbf{S}}_k, (\varphi_k, X_k, R_k)\Big) = (1-\gamma) \cdot \ell_k(\mathbf{S}, (\varphi_k, X_k, R_k)) +$$
$$\gamma \cdot \sum_{\widehat{\mathbf{S}}'_k \in \widehat{\mathcal{S}}_k} \mathbb{P}\Big(\widehat{\mathbf{S}}'_k | \widehat{\mathbf{S}}_k, (\varphi_k, X_k, R_k)\Big) \cdot V_k\Big(\widehat{\mathbf{S}}'_k\Big), \tag{21}$$

where φ_k, X_k and R_k correspond to the channel allocation, the computation offloading and the packet scheduling decisions under $\widehat{\mathbf{S}}_k$. We further define a post-decision Q-factor by

$$\widetilde{Q}_k\Big(\widetilde{\mathbf{S}}_k, (\varphi_k, X_k, R_k)\Big) = \gamma \cdot \sum_{\widehat{\mathbf{S}}'_k \in \widehat{\mathcal{S}}_k} \mathbb{P}\Big(\widehat{\mathbf{S}}'_k | \widetilde{\mathbf{S}}_k, (\varphi_k, X_k, R_k)\Big)$$
$$\cdot\ V_k\Big(\widehat{\mathbf{S}}'_k\Big). \tag{22}$$

By substituting (22) back into (20), we obtain

$$\nu_k = u_k\Big(\mathbf{S}^j, \Big(z_k, \pi^*_{(\mathrm{t}),k}(\mathbf{S}_k), \pi^*_{(\mathrm{p}),k}(\mathbf{S}_k)\Big)\Big)$$
$$+ \frac{1}{1-\gamma} \cdot \widetilde{Q}_k\Big(\widetilde{\mathbf{S}}_k, \Big(z_k, \pi^*_{(\mathrm{t}),k}(\mathbf{S}_k), \pi^*_{(\mathrm{p}),k}(\mathbf{S}_k)\Big)\Big). \tag{23}$$

where z_k can be hence derived from

$$z_k = \arg\max_{z \in \{0,1\}} Q_k\Big(\widehat{\mathbf{S}}_k, \Big(z, \pi^*_{(\mathrm{t}),k}(\mathbf{S}_k), \pi^*_{(\mathrm{p}),k}(\mathbf{S}_k)\Big)\Big). \tag{24}$$

It is easy to observe that $\widehat{\mathcal{S}}_k$ faced by each MU $k \in \mathcal{K}$ is extremely huge. The tabular nature in representing the Q-factor and the post-decision Q-factor values makes the conventional Q-learning rule impractical. Inspired by the widespread success of a deep neural network [18], we propose to adopt two

$$V_k\left(\widehat{\mathbf{S}}_k\right) = \max_{\boldsymbol{\pi}_k\left(\widehat{\mathbf{S}}_k\right)} \Bigg\{ (1-\gamma)\cdot \ell_k\Big(\mathbf{S}, \varphi_k\Big(\pi_{(\mathrm{c}),k}\Big(\widehat{\mathbf{S}}_k\Big), \boldsymbol{\pi}^*_{(\mathrm{c}),-k}\Big(\widehat{\mathbf{S}}_{-k}\Big)\Big), \pi_{(\mathrm{t}),k}(\mathbf{S}_k), \pi_{(\mathrm{p}),k}(\mathbf{S}_k)\Big)$$
$$+ \gamma \cdot \sum_{\widehat{\mathbf{S}}'_k \in \widehat{\mathcal{S}}_k} \mathbb{P}\Big(\widehat{\mathbf{S}}'_k | \widehat{\mathbf{S}}_k, \Big(\varphi_k\Big(\pi_{(\mathrm{c}),k}\Big(\widehat{\mathbf{S}}_k\Big), \boldsymbol{\pi}^*_{(\mathrm{c}),-k}\Big(\widehat{\mathbf{S}}_{-k}\Big)\Big), \pi_{(\mathrm{t}),k}(\mathbf{S}_k), \pi_{(\mathrm{p}),k}(\mathbf{S}_k)\Big)\Big) \cdot V_k\Big(\widehat{\mathbf{S}}'_k\Big) \Bigg\} \quad (17)$$

separate deep Q-networks (DQNs), namely, DQN-I and DQN-II, to reproduce the Q-factor and the post-decision Q-factor of a MU. Specifically, for each MU k, we model the Q-factor in (21) by, $\forall(\widehat{\mathbf{S}}_k, (\varphi_k, X_k, R_k)) \in \widehat{\mathcal{S}}_k \times \{0,1\} \times \mathcal{X} \times \mathcal{D}$,

$$Q_k\Big(\widehat{\mathbf{S}}_k, (\varphi_k, X_k, R_k)\Big) \approx Q_k\Big(\widehat{\mathbf{S}}_k, (\varphi_k, X_k, R_k); \boldsymbol{\theta}_k\Big), \quad (25)$$

and the post-decision Q-factor in (22) by, $\forall(\widetilde{\mathbf{S}}_k, (\varphi_k, X_k, R_k)) \in \widehat{\mathcal{S}}_k \times \{0,1\} \times \mathcal{X} \times \mathcal{D}$,

$$\widetilde{Q}_k\Big(\widetilde{\mathbf{S}}_k, (\varphi_k, X_k, R_k)\Big) \approx \widetilde{Q}_k\Big(\widetilde{\mathbf{S}}_k, (\varphi_k, X_k, R_k); \tilde{\boldsymbol{\theta}}_k\Big), \quad (26)$$

where $\boldsymbol{\theta}_k$ and $\tilde{\boldsymbol{\theta}}_k$ denote the vectors of parameters that are associated with DQN-I and DQN-II.

During the online DRL, each MU $k \in \mathcal{K}$ is equipped with a finite replay memory $\mathcal{M}_k^j = \{\mathbf{y}_k^{j-M+1}, \cdots, \mathbf{y}_k^j\}$ to track the most recent M historical experiences up to a decision epoch j, where an experience $\mathbf{y}_k^{j-m+1}$ $(1 \le m \le M)$ given by

$$\mathbf{y}_k^{j-m+1} = \Big(\widehat{\mathbf{S}}_k^{j-m}, \Big(\varphi_k^{j-m}, X_k^{j-m}, R_k^{j-m}\Big), \quad (27)$$
$$\ell_k\Big(\mathbf{S}^{j-m}, \Big(\varphi_k^{j-m}, X_k^{j-m}, R_k^{j-m}\Big)\Big), \widehat{\mathbf{S}}_k^{j-m+1}\Big).$$

1) DQN-I Training: MU k keeps a $Q_k(\widehat{\mathbf{S}}_k, (\varphi_k, X_k, R_k); \boldsymbol{\theta}_k^j)$ and a target $Q_k(\widehat{\mathbf{S}}_k, (\varphi_k, X_k, R_k); \boldsymbol{\theta}_k^{j,-})$, where $\boldsymbol{\theta}_k^j$ and $\boldsymbol{\theta}_k^{j,-}$ are the associated parameters at each decision epoch j and from a previous epoch before j, respectively. To perform experience replay [19], MU k randomly samples a mini-batch $\mathcal{Y}_k^j \subseteq \mathcal{M}_k^j$ to train DQN-I, the objective of which is to update $\boldsymbol{\theta}_k^j$ in the direction of minimizing the loss given by (28).

2) DQN-II Training: At each decision epoch j, we designate $\tilde{\boldsymbol{\theta}}_k^j$ as the parameters associated with DQN-II of MU k. Taking $\boldsymbol{\theta}_k^j$ from DQN-I as an input, MU k updates $\tilde{\boldsymbol{\theta}}_k^j$ to minimize the loss given by (29) over the mini-batch $\mathcal{Y}_k^j$.

VI. Numerical Experiments

To evaluate the performance of the proposed online DRL scheme, we conduct numerical experiments based on TensorFlow [20]. We build an experimental RAN covering a 0.4×0.4 Km^2 square area, where there are $B = 4$ BSs and $|\mathcal{K}| = 20$ MUs. The BSs are placed at equal distance apart, and the service area is divided into $|\mathcal{L}| = 1600$ locations. The UAV flies at the altitude of $H = 100$ meters. For each MU $k \in \mathcal{K}$, $G_{b,k}^j$ and $G_{(\mathrm{v}),k}^j$, $\forall b \in \mathcal{B}$ and $\forall j$, follow the channel model in [2] and the LOS model in [21], respectively. The state transition probability matrices of all MUs and the UAV are generated independently and randomly. We design the DQN-I and the DQN-II of each MU assuming two hidden layers, each containing 32 neurons. ReLU is selected as the activation function [22] and Adam as the optimizer [23]. Other parameter values are listed in Table I. We consider the following baseline schemes as well for the performance comparisons.

TABLE I
Parameter values in experiments.

Parameter	Value	Parameter	Value
$D_{(\max)}$	10	μ	500 Kbits
ϑ	1300	$A_{(\max)}$	30 seconds
η	1 MHz	σ^2	-144 dBm/Hz
δ	1 second	$P_{(\max)}$	3 Watt
ϖ_k	10, $\forall k$	ω_k	2, $\forall k$
ϱ	1 GHz	ξ	10^{-2} seconds
χ_0	$2 \cdot 10^7$ bits/second	ε	0.2
ς	10^{-27}	M	5000

1) *Local Computation (Baseline 1)* – Each MU processes the computation tasks only at the local mobile device.
2) *Server Execution (Baseline 2)* – Each MU always offloads the computations to the edge server for execution.
3) *UAV Execution (Baseline 3)* – All computation tasks of the MU are processed at the UAV.
4) *Greedy Processing (Baseline 4)* – Whenever possible, a task is computed locally or executed remotely.

In Baselines 2, 3 and 4, the valuation at each decision epoch is calculated as the utility from transmitting a maximum number of input data packets.

The first experiment demonstrates the average utility per MU per decision epoch by changing the task arriving probabilities. We assume $|\mathcal{C}| = 16$ channels that can be utilized by the MUs. The results are exhibited in Fig. 1, from which we observe that the proposed DRL scheme achieves the best performance. As the computation task arriving probability increases, each MU consumes more energy for task processing to maintain the fresh knowledge. With the chosen weights, the AoI increasingly dominates the utility function value when the energy consumption increases, which confirms the average utility performance trends of all schemes. Then we simulate the average utility performance versus the numbers of channels, where we select $\lambda = 0.5$. The more channels available, the more likely a MU wins from the channel auction. Therefore, with Baselines 2, 3 and 4, the MU consumes more energy to offload more input data packets for remote execution, while with the proposed scheme, there are more opportunities for a MU to process a computation task with less energy consumption. Using Baseline 1, the average utility keeps constant since the MUs do not participate the channel auction. Last but not least, both experiments corroborate the performance gains of the proposed scheme.

VII. Conclusions

This paper aims to design an optimal control policy for the MUs in a three-dimensional UAV-assisted MEC system.

$$\mathsf{LOSS}_{(\text{DQN-I}),k}\left(\boldsymbol{\theta}_k^j\right) = \mathsf{E}_{\left\{\left(\widehat{\mathbf{S}}_k,(\varphi_k,X_k,R_k),\ell_k(\mathbf{S},(\varphi_k,X_k,R_k)),\widehat{\mathbf{S}}_k'\right)\in\mathcal{Y}_k^j\right\}}\Bigg[\Bigg((1-\gamma)\cdot\ell_k(\mathbf{S},(\varphi_k,X_k,R_k)) + \gamma\cdot Q_k\left(\widehat{\mathbf{S}}_k', \underset{\varphi_k',X_k',R_k'}{\arg\max} Q_k\left(\widehat{\mathbf{S}}_k',(\varphi_k',X_k',R_k');\boldsymbol{\theta}_k^j\right);\boldsymbol{\theta}_k^{j,-}\right) - Q_k\left(\widehat{\mathbf{S}}_k,(\varphi_k,X_k,R_k);\boldsymbol{\theta}_k^j\right)\Bigg)^2\Bigg] \quad (28)$$

$$\mathsf{LOSS}_{(\text{DQN-II}),k}\left(\tilde{\boldsymbol{\theta}}_k^j\right) = \mathsf{E}_{\left\{\left(\widehat{\mathbf{S}}_k,(\varphi_k,X_k,R_k),\ell_k(\mathbf{S},(\varphi_k,X_k,R_k)),\widehat{\mathbf{S}}_k'\right)\in\mathcal{Y}_k^j\right\}}\Bigg[\Bigg(\gamma\cdot\max_{\varphi_k',X_k',R_k'} Q_k\left(\widehat{\mathbf{S}}_k',(\varphi_k',X_k',R_k');\boldsymbol{\theta}_k^j\right) - \widetilde{Q}_k\left(\widetilde{\mathbf{S}}_k,(\varphi_k,X_k,R_k);\tilde{\boldsymbol{\theta}}_k^j\right)\Bigg)^2\Bigg] \quad (29)$$

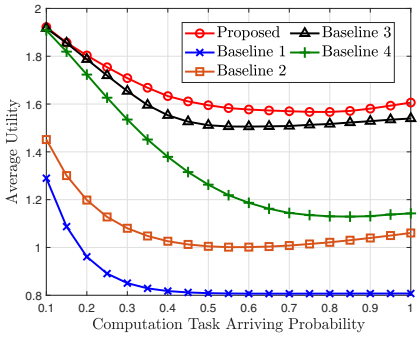

Fig. 1. Average utility performance per MU across the learning procedure versus computation task arriving probability.

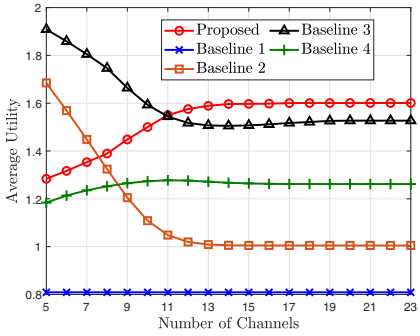

Fig. 2. Average utility performance per MU across the learning procedure versus number of channels.

Each MU selfishly maximizes its own expected long-term computation performance. We formally formulate the non-cooperative interactions among the MUs as a stochastic game. To approach the NE, we develop a novel online DRL scheme, which maintains two separate DQNs to approximate the Q-factor and the post-decision Q-factor for each MU. Implementing the proposed DRL scheme, each MU makes the decisions of computation offloading, channel auction and input data packet scheduling with only the local information. From numerical experiments, we find that compared with the four baselines, the proposed scheme achieves better average utility performance, indicating a better tradeoff between AoI and energy consumption.

REFERENCES

[1] Y. Mao *et al.*, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, Q4 2017.

[2] X. Chen *et al.*, "Multi-tenant cross-slice resource orchestration: A deep reinforcement learning approach," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2377–2392, Oct. 2019.

[3] M. Mozaffari *et al.*, "A tutorial on UAVs for wireless networks: Applications, challenges, and open problems," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, Q3 2019.

[4] X. Hu, K.-K. Wong, K. Yang, and Z. Zheng, "UAV-assisted relaying and edge computing: Scheduling and trajectory optimization," *IEEE Trans. Wireless Commun.*, vol. 18, no. 10, pp. 4738–4752, Oct. 2019.

[5] M. Li *et al.*, "Energy-efficient UAV-assisted mobile edge computing: Resource allocation and trajectory optimization," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3424–3438, Mar. 2020.

[6] X. Chen *et al.*, "Resource awareness in unmanned aerial vehicle-assisted mobile-edge computing systems," in *Proc. IEEE VTC*, May 2020.

[7] Z. Ji and K. J. R. Liu, "Dynamic spectrum sharing: A game theoretical overview," *IEEE Commun. Mag.*, vol. 45, no. 5, pp. 88–94, May 2007.

[8] Z. Liang, Y. Liu, T.-M. Lok, and K. Huang, "Multiuser computation offloading and downloading for edge computing with virtualization," *IEEE Trans. Wireless Commun.*, vol. 18, no. 9, pp. 4298–4311, Sep. 2019.

[9] S. Kaul, M. Gruteser, V. Rai, and J. Kenney, "Minimizing age of information in vehicular networks," in *Proc. IEEE SECON*, Salt Lake City, UT, Jun. 2011.

[10] X. Chen *et al.*, "Age of information-aware radio resource management in vehicular networks: A proactive deep reinforcement learning perspective," *IEEE Trans. Wireless Commun.*, vol. 19, no. 4, pp. 2268–2281, Apr. 2020.

[11] H. van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. AAAI*, Phoenix, AZ, Feb. 2016.

[12] T. D. Burd and R. W. Brodersen, "Processor design for portable systems," *J. VLSI Signal Process. Syst.*, vol. 13, no. 2–3, pp. 203–221, Aug. 1996.

[13] X. Chen *et al.*, "Optimized computation offloading performance in virtual edge computing systems via deep reinforcement learning," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4005–4018, Jun. 2019.

[14] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.

[15] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.

[16] A. M. Fink, "Equilibrium in a stochastic n-person game," *J. Sci. Hiroshima Univ. Ser. A-I*, vol. 28, pp. 89–93, 1964.

[17] X. Chen *et al.*, "Wireless resource scheduling in virtualized radio access networks using stochastic learning," *IEEE Trans. Mobile Comput.*, vol. 17, no. 4, pp. 961–974, Apr. 2018.

[18] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.

[19] L.-J. Lin, "Reinforcement learning for robots using neural networks," Carnegie Mellon University, 1992.

[20] M. Abadi *et al.*, "Tensorflow: A system for large-scale machine learning," in *Proc. OSDI*, Savannah, GA, Nov. 2016.

[21] Y. Zeng, R. Zhang, and T. J. Lim, "Throughput maximization for UAV-enabled mobile relaying systems," *IEEE Trans. Commun.*, vol. 64, no. 12, pp. 4983–4996, Dec. 2016.

[22] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proc. ICML*, Haifa, Israel, Jun. 2010.

[23] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *Proc. ICLR*, San Diego, CA, May 2015.