Traffic Steering for Cellular-Enabled UAVs: A Federated Deep Reinforcement Learning Approach

Hamed Hellaoui[1], Bin Yang[2], Tarik Taleb[3], and Jukka Manner[1]

[1]Aalto University, Communications and Networking Department, Finland. Email: {firstname.lastname}@aalto.fi
[2]Chuzhou University, School of Computer and Information Engineering, China. Email: yangbinchi@gmail.com
[3]University of Oulu, Centre for Wireless Communications, Finland. Email: tarik.taleb@oulu.fi

***Abstract*—This paper investigates the fundamental traffic steering issue for cellular-enabled unmanned aerial vehicles (UAVs), where each UAV needs to select one from different Mobile Network Operators (MNOs) to steer its traffic for improving the Quality-of-Service (QoS). To this end, we first formulate the issue as an optimization problem aiming to minimize the maximum outage probabilities of the UAVs. This problem is non-convex and non-linear, which is generally difficult to be solved. We propose a solution based on the framework of deep reinforcement learning (DRL) to solve it, in which we define the environment and the agent elements. Furthermore, to avoid sharing the learned experiences by the UAV in this solution, we further propose a federated deep reinforcement learning (FDRL)-based solution. Specifically, each UAV serves as a distributed agent to train separate model, and is then communicated to a special agent (dubbed coordinator) to aggregate all training models. Moreover, to optimize the aggregation process, we also introduce a FDRL with DRL-based aggregation (DRL2A) approach, in which the coordinator implements a DRL algorithm to learn optimal parameters of the aggregation. We consider deep Q-learning (DQN) algorithm for the distributed agents and Advantage Actor-Critic (A2C) for the coordinator. Simulation results are presented to validate the effectiveness of the proposed approach.**

***Index Terms*—Unmanned Aerial Vehicles (UAVs), Cellular Networks, Connection Steering, Deep Reinforcement Learning (DRL), Federated Deep Reinforcement Learning (FDRL), FDRL with DRL-based Aggregation (DRL2A).**

I. Introduction

Unmanned Aerial Vehicles (UAVs) have shown great prospects in a wide range of applications, such as crowd surveillance, rescue management and disaster recovery, which urgently demand the support of high performance communications. A promising technology is to integrate UAVs into cellular networks, where UAVs can utilize ubiquitous base stations (BSs) to communicate with distant control center/users. Meanwhile, UAVs can reuse the licensed spectrum resources of cellular networks to achieve high rate, reliability and security for data transmission. Therefore, cellular UAVs have been identified as one important component of the next generation wireless networks. The connection steering, which is to dynamically steer UAVs' communications to one of Mobile Network Operators (MNOs), is significantly important to guarantee the Quality-of-Service (QoS) for supporting various applications of cellular UAVs.

As shown in Fig. 1, a framework for traffic steering would enable a UAV to be connected to several MNOs in the same time, while sending the traffic via the one ensuring the best QoS. In this framework, a steering gateway is considered in an edge cloud located near to the BSs of the concerned MNOs. This gateway has the role of preserving one IP address when communicating with the internet, ensuing therefore seamless steering. Based on the above observations, two fundamental and interesting issues arise for cellular-enabled UAVs. One issue is how to select an optimal MNO for each UAV to enhance the performance of UAV communications. Another issue is how to solve the complex optimization problem for large-scale networks in a relatively short time.

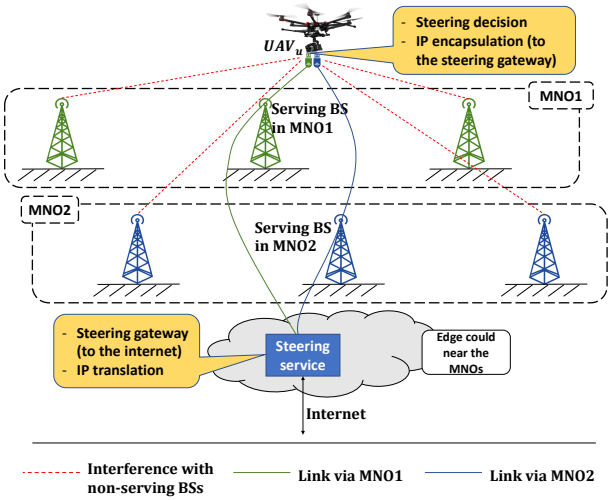


Fig. 1: Framework for traffic steering via different mobile networks.

In the literature, different methods have been proposed to improve the QoS of cellular UAVs. The work in [1] aims at minimizing the outage probability by jointly optimizing channel and power allocation based on game theory. The data rate is maximized by jointly optimizing channel and power allocation, where a weighted mean square error (MSE) is used to model the problem [2]. The work in [3] aims to optimize the transmit power of UAVs serving as aerial BSs while maximizing the data rate from the UAVs to their served users based on the transport theory and facility location. In [4], the authors propose a price-based power allocation scheme to maximize the system utility using Stackelberg game theory.

Recently, some initial works exploit machine learning (ML) approaches to optimize the performance of UAV communications [5], [6]. The work in [5] first formulates the optimal deployment of multiple UAVs acting as BSs to serve ground

users as a mixed-integer program, and then solves it using an unsupervised learning approach. In [6], the goal of the work is to maximize the data rate by jointly optimizing user association, power allocation and trajectory design based on a deep reinforcement Learning (DRL) approach. Other works such as [7]–[9] use DRL to perform path planning for cellular-enabled UAVs.

However, the connection steering has not been well explored for cellular-enabled UAVs. Fewer works have addressed the problem of connection steering, especially between the connected devices and the serving BSs. Initial solutions have been provided in [10], [11]. However, this solutions require a central agent that collects experiences from all the UAVs and issue decisions on the selected MNOs. Such solutions can not be applicable in situations where the UAVs are not willing to share their experiences and consider them as private data. To address these challenging issues, this paper advances a Federated Deep Reinforcement Learning (FDRL) solution for UAV traffic steering in cellular networks. The main contributions of this paper are summarized as follows.

- We propose a DRL framework, based on deep Q-learning (DQN), for UAV traffic steering in which we define the environment and the agent elements.
- We further propose a FDRL solution, where distributed agents train local models without sharing their experiences. The local models will be communicated a coordinator to aggregate them.
- To enhance the aggregation process, we also propose a DRL-based aggregation (FDRL2A) approach, where the coordinator also implements a DRL algorithm. The advantage actor-critic (A2C) is considered for the coordinator.

The rest of the paper is organized as follows. The system model and the problem formulation are provided in Section II. The proposed DRL framework for UAV traffic steering in cellular networks is introduced in Section III. The extended FDRL is thereafter presented in Section IV. Performance evaluations are then provided in Section V. Finally, Section VI concludes this paper.

II. System Model and Problem Formulation

1) System Model: We consider an uplink transmission scenario, where data is transmitted from the flying UAVs to the serving BSs. We use $\mathcal{U}$ and $\mathcal{O}$ to denote the set of UAVs and MNOs, respectively. We also use $\mathcal{V}_o$ and $\mathcal{C}_o$ to denote the set of BSs belonging to the MNO $o \in \mathcal{O}$ and the set of RB belonging to the same MNO, respectively. In the concerned scenario, each UAV is connected to several MNOs and steers the traffic via a selected one. Let uv_o denote the link between the UAV $u \in \mathcal{U}$ and its serving BS $v_o \in \mathcal{V}_o$ in the MNO $o \in \mathcal{O}$, and denote tv_o the link between the interfering UAV $t \in \mathcal{U}$ to the BS $v_o \in \mathcal{V}_o$ in the MNO $o \in \mathcal{O}$.

The received signal-to-noise ratio (SNR) γ_{uv_o} for the link uv_o, is given by

$$\gamma_{uv_o} = P_u |\alpha_{uv_o}|^2 / N_0, \tag{1}$$

where P_u denotes the transmit power of UAV u, α_{uv_o} denotes the channel gain between the transmitter u and the receiver v_o, and N_0 stands for the variance of a zero-mean additive white Gaussian process. We can define the instantaneous received signal-to-interference-plus-noise ratio (SINR) for the link between a UAV u and the BS v_o as

$$SINR_{uv_o} = \gamma_{uv_o} / (1 + \sum_{t \in \mathcal{U}}^{t \neq u} \gamma_{tv_o}). \tag{2}$$

We use $P_{uv_o}^{out}(\gamma_{th})$ to denote the outage probability of the UAV u, which is defined as the probability that u fails in transmitting its data to its serving BS v_o in the MNO o. Then, we have

$$P_{uv_o}^{out}(\gamma_{th}) = \sum_{j=1}^{m} \left(\sigma_{1j} \frac{(-1)^j}{(j-1)!} \left(\frac{m}{A_{uv_o}} \right)^{-j} \left(\Gamma(j) + \sum_{t=1}^{N} \delta'_t f_{j,1}(B_{tv_o}) - \sum_{t=1}^{N} \sum_{j'=1}^{m} \delta_{t,j'} \frac{(-1)^{j'}}{(j'-1)!} f_{j,j'}\left(\frac{A_{tv_o}}{m}\right) \right) \right) - \sigma_{21} B_{uv_o} \left[1 + \exp\left(-\frac{\gamma_{th}}{B_{uv_o}} \right) \left(\sum_{t=1}^{N} \frac{\delta'_t}{\frac{\gamma_{th}}{B_{uv_o}} + \frac{1}{B_{tv_o}}} - \sum_{t=1}^{N} \sum_{j=1}^{m} \frac{\delta_{t,j}}{\left(\frac{\gamma_{th}}{B_{uv_o}} + \frac{m}{A_{tv_o}} \right)^j} \frac{(-1)^j}{(j-1)!} \Gamma(j) \right) \right], \tag{3}$$

where γ_{th} is the SINR threshold. A_{uv_o} and B_{uv_o} refer respectively to the mean SNR characterizing the LoS and NLoS conditions for the link uv_o. m is the parameter of the Nakagami distribution which is used for LoS link. $\Gamma(j)$ is the gamma function. $([1, \ldots, N])$ refers to the list of interferer UAVs. The terms σ_{1j}, σ_{21}, δ'_t, and $\delta_{t,j}$ have unique values satisfying the fractional decomposition formulas [12, Eq. (10) and Eq. (12)]. $f_{j,j'}(y)$ is a Laguerre polynomial-based function [12, Eq. (13)]. The proof of the outage probability formula is provided in [12].

2) Problem Formulation: To enhance the QoS, our objective is to minimize the maximum outage probability by optimizing the selection of MNO for each UAV. We use a Boolean variable x_{uo} to decide whether a UAV selects one from the set $\mathcal{O}$ of MNOs or not, where

$$x_{uo} = \begin{cases} 1 & \text{if the UAV } u \in \mathcal{U} \text{ selects the MNO } o \in \mathcal{O} \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Then, the traffic steering can be formulated as the following constrained optimization problem:

$$\underset{\{x_{uo}\}}{\text{minimize}} \max_{u \in \mathcal{U}} \left(\sum_{o \in \mathcal{O}} x_{uo} P_{uv_o}^{out}(\gamma_{th}) \right) \tag{5}$$

s.t.

$$\sum_{o \in \mathcal{O}} x_{uo} = 1, \quad \forall u \in \mathcal{U} \tag{6}$$

$$x_{uo} \in \{0, 1\}, \quad \forall u \in \mathcal{U}, \forall o \in \mathcal{O}. \tag{7}$$

However, this problem is nonlinear due to the nonlinear expression of the outage probability defined in the objective function of the above optimization problem. To solve it, we propose a deep reinforcement learning framework of UAV traffic steering in the next section.

III. A Deep Reinforcement Learning Framework for UAV Traffic Steering

In this section, we provide a solution for the problem of UAV traffic steering based on the framework of deep reinforcement learning. Throughout interactions with the environment, an agent can learn complex tasks and decide on their execution in a way to optimize a given objective. A central agent is considered in the proposed framework to train a model allowing to select the optimal MNO for each UAV. At a time slot t, the agent gets current state of the environment s_t, and then decides on the appropriate action a_t (MNOs to be selected) to execute. The agent will thereafter get the next state s_{t+1} and the associated reward r_t. A replay memory $\mathcal{M}$ is also used to store the experiences that will be used to train the model. In what follows, we further define the system state, the action space and the system reward. In addition, we also introduce the learning process for training the model where a DQN algorithm is considered.

A. System State

The system state is used to capture the features characterizing the network deployment. At each time slot t, it can be described as

$$s_t = [\varphi_t, D_t, \Omega_t, \Theta_t] \in \mathcal{S}, \tag{8}$$

where

$$\begin{cases} \varphi_t &= [\psi_{uv}]_{u,v} \in [-\pi, \pi]^{|\mathcal{U}|\times|\cup_{o\in\mathcal{O}}\mathcal{V}_o|}, \\ D_t &= [d_{uv}]_{u,v} \in \mathbb{R}^{|\mathcal{U}|\times|\cup_{o\in\mathcal{O}}\mathcal{V}_o|}, \\ \Omega_t &= [v_{uo}]_{u,o} \in \cup_{o\in\mathcal{O}}(\mathcal{V}_o^{|\mathcal{U}|}), \\ \Theta_t &= [c_{uo}]_{u,o} \in \cup_{o\in\mathcal{O}}(\mathcal{C}_o^{|\mathcal{U}|}), \end{cases} \tag{9}$$

and $\mathcal{S}$ is the set of states. In (9), ψ_{uv} refers to the orientation angle formed between the UAV $u \in \mathcal{U}$ and the BS $v \in \cup_{o\in\mathcal{O}}\mathcal{V}_o$, which is computed as $\psi_{uv} = tan^{-1}(\Delta^y_{uv}/\Delta^x_{uv})$, where Δ^x_{uv} and Δ^y_{uv} respectively correspond to the difference in the x and the y coordinates between u and v. d_{uv} refers to the distance between the UAV $u \in \mathcal{U}$ and the BS $v \in \cup_{o\in\mathcal{O}}\mathcal{V}_o$. v_{uo} is the serving BS of the UAV $u \in \mathcal{U}$ in the MNO $o \in \mathcal{O}$. As for c_{uo}, it corresponds to the assigned resource block to the UAV $u \in \mathcal{U}$ in the MNO $o \in \mathcal{O}$.

B. Action Space

After receiving a system state s_t, the agent will decide on the action a_t to perform. The action space is defined to reflected to choice of the selected MNOs to be used for traffic steering for each UAV $u \in \mathcal{U}$. Therefore, at each time slot t, an action is described as

$$a_t = [a^u_t]_u \in \mathcal{O}^{|\mathcal{U}|}, \tag{10}$$

where $a^u_t \in \mathcal{O}$ corresponds to the MNO selected for the UAV u at time slot t.

C. System Reward

The system reward is defined based on the objective function in such a way that maximizing the reward values would be translated into minimizing the outage probabilities. Regarding a system state s_t and the action a_t, the corresponding system reward function $\mathcal{R}(s_t, a_t)$ is defined as

$$\begin{cases} \mathcal{R}(s_t, a_t) = [r^u_t]_u \in [0,1]^{|\mathcal{U}|}, \\ r^u_t = 1 - P^{out}_{uv_o}(\gamma_{th}), \end{cases} \tag{11}$$

where $P^{out}_{uv_o}(\gamma_{th})$ is the outage probability of the UAV u after considering the action a_t in the state s_t.

D. Learning Process

Based on the above definitions of the system state, action space and system reward, we provide in this subsection a learning process allowing to train a model to select optimal actions for given system states. To this end, we consider the DQN algorithm. The optimization objective is to find an optimal policy $\pi \in \Pi$ for maximizing the expected long-term reward, which is expressed as the following the V-function $V_\pi(s)$:

$$\begin{cases} V_*(s) = \max_{\{\pi\}} V_\pi(s), \\ V_\pi(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \tau^t \mathcal{R}(s_t, a_t) | s_0 = s\right], \end{cases} \tag{12}$$

where $\mathbb{E}[.]$ is the expectation operator and $\tau \in [0,1]$ reflects a discount factor. By applying the Bellman equation, $V_\pi(s)$ can be written as

$$V_\pi(s) = \sum_{a\in\mathcal{O}^{|\mathcal{U}|}} \pi(a|s) \underbrace{\left(\mathcal{R}(s,a) + \tau \sum_{s'\in\mathcal{S}} P(s'|(s,a)).V_\pi(s')\right)}_{Q_\pi(s,a)}, \tag{13}$$

where a is the action taken at the state s, $\pi(a|s)$ denotes the possibility of taking the action a when the sate is s, and s' is the possible resulting states after executing a. The function $Q_\pi(s,a)$ reflects the Q-function which defines the value of the taken action a in the state s under the policy π. Based on the Bellman optimality equation, the optimal policy can be formulated as

$$\begin{cases} V_*(s_t) = \max_{a_t} Q_*(s_t, a_t), \\ Q_*(s_t, a_t) = \mathcal{R}(s_t, a_t) + \tau \max_{a_{t+1}} Q_*(s_{t+1}, a_{t+1}). \end{cases} \tag{14}$$

Given the complexity of the environment, which is characterized by continuous state space, we consider a deep neural network to estimate the function $Q_\pi(s_t, a_t)$. Let θ_t denote the parameters of the model at time slot t. We can write $Q_\pi(s_t, a_t) \approx Q_\pi(s_t, a_t, \theta_t)$. We also use the history of experiences stored in the replay memory and the gradient decent to update the model parameters. More precisely, the parameters θ_t are learned by iteratively minimizing the loss function defined as

$$\mathcal{L}^q(\theta_t) = \sum_{(s_t,a_t)\in\mathcal{M}} \big(\mathcal{R}(s_t, a_t) + \tau \max_{a_{t+1}} Q_\pi(s_{t+1}, a_{t+1}, \theta_{t-1}) - Q_\pi(s_t, a_t, \theta_t)\big)^2. \tag{15}$$

The above learning process allows to train models to decide optimal selection of MNO for each UAV in a way to reduce their outage probabilities. This training process is performed

by an agent throughout the interaction with the environment. However, the use of a centralized agent requires UAVs to share their achieved outage probability with this agent. In addition, the latter is aware of the selection performed by each UAV as it holds the trained model. In order to address the issue where UAVs consider these information (i.e., the action and the captured reward) as private information, we propose a federated deep reinforcement learning framework for UAV traffic steering. The latter allows UAVs to train individual model without the need to share their achieved QoS or the selected MNOs.

IV. A Federated Deep Reinforcement Learning Framework for UAV Traffic Steering

This section introduces the proposed framework of FDRL for UAV traffic steering. The aim is to enable training models in a distributed way to select the MNO used to steering the traffic for each UAV, without sharing the experiences or the selected MNOs by the UAVs. The general architecture of the framework is depicted in Fig. 2. In this framework, each UAV will be associated with an agent that deals with individual experiences and trains a model to issue individual actions. On the other hand, a coordinator is considered to aggregated the individual models. Therefore, we first derive the structure of the individual agents. Then, we introduce the structure of the coordinator.

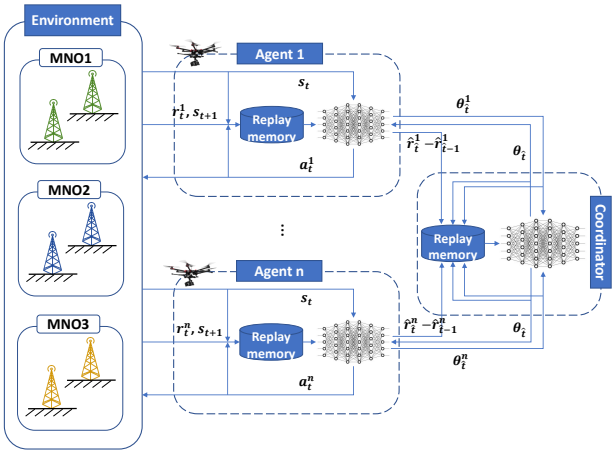


Fig. 2: Architecture of the FDRL2A model.

A. Distributed Reinforcement Learning

In this subsection, we propose a distributed reinforcement learning for the problem of UAV traffic steering. Considering (10) and (11), we can write the system actions and rewards as

$$\begin{cases} a_t = [a_t^u]_u, \\ \mathcal{R}(s_t, a_t) = [r_t^u]_u = [\mathcal{R}^u(s_t, a_t^u)]_u. \end{cases} \tag{16}$$

We can see from (16) that the system actions and rewards are based on individual values from the different UAVs. Note that the function $\mathcal{R}^u(s_t, a_t^u)$ in practice is computed locally, as each device can perceive such QoS. On the other hand, the system state is kept shared among all the agents. This is commonly known as vertical federated reinforcement learning [13]. As reflected in Fig. 2, each agent $u \in \mathcal{U}$ deals with individual actions, a_t^u, and rewards, r_t^u, while the states s_t are kept shared. Based on these formulations of the system actions and rewards, we derive the learning process to be considered locally by each UAV.

Considering the DQN algorithm described in the previous section, the underlying optimization aims to maximize the long-term reward. We first express the value function $V_\pi(s)$ as

$$\begin{aligned} V_\pi(s) &= \mathbb{E}\left[\sum_{t=0}^{\infty} \tau^t \mathcal{R}(s_t, a_t) | s_0 = s\right] \\ &= \mathbb{E}\left[\sum_{t=0}^{\infty} \tau^t [\mathcal{R}^u(s_t, a_t^u)]_u | s_0 = s\right] = [V_\pi^u(s)]_u. \end{aligned} \tag{17}$$

As we can see from (17), the value function $V_\pi(s)$ can also be expressed based on the functions $V_\pi^u(s)$ corresponding to each UAV $u \in \mathcal{U}$. Each agent will therefore target to maximize the expected long term-term reward and find the optimal strategy that achieves $V_*^u(s)$. By considering the Bellman equation and following the same approach of the previous section, we can express the optimal policy for each agent $u \in U$ as

$$\begin{cases} V_*^u(s_t) = \max_{a_t^u} Q_*^u(s_t, a_t^u), \\ Q_*^u(s_t, a_t^u) = \mathcal{R}^u(s_t, a_t^u) + \tau \max_{a_{t+1}^u} Q_*^u(s_{t+1}, a_{t+1}^u), \end{cases} \tag{18}$$

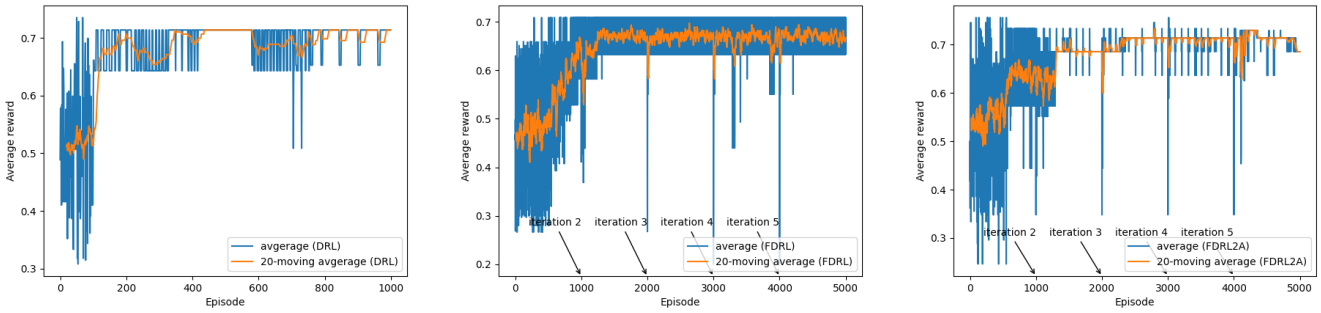
where $Q_*^u(s_t, a_t^u)$ is the Q-function implemented by the agent u. Note that the agents are dealing with local experiences and their policies would be different. In order to estimate the function $Q_\pi^u(s_t, a_t^u)$, each agent $u \in \mathcal{U}$ operates a local model whose parameters at time slot t are denoted by θ_t^u. We can therefore write $Q_\pi^u(s_t, a_t^u) \approx Q_\pi^u(s_t, a_t^u, \theta_t^u)$.

B. Model Aggregation

As the distributed agents do not share their experiences, they operate different models. As depicted in Fig. 2, a coordinator is considered in the proposed framework to aggregate the different models and return the resulting one to the distributed agents. The coordinator therefore does not deal with the experiences of the agents, maintaining therefore the privacy of the data. The aggregation is performed at a predefined round of iterations. Let $\hat{t}$ denote the evolving iteration round. The model aggregation is expressed as

$$\theta_{\hat{t}} = \sum_{u \in \mathcal{U}} \beta_{\hat{t}}^u \theta_{\hat{t}}^u, \tag{19}$$

where $[\beta_{\hat{t}}^u]_u \in [0,1]^{|\mathcal{U}|}$ is a weighting parameter, and $\sum_{u \in \mathcal{U}} \beta_{\hat{t}}^u = 1$. $\theta_{\hat{t}}$ corresponds to the model that will be shared back with the distributed agents. The value of the weighting parameters $[\beta_{\hat{t}}^u]_u$ affects the inference result. To further optimize this process, we propose in this paper to consider the coordinator as an agent whose objective is to learn optimal values for the weighting parameters. We propose a DRL-based aggregation (FDRL2A) approach. Under such an approach, the coordinator decides on the values of the weighting parameters, which is used to produce the aggregated



(a) DRL (b) FDRL (c) FDRL2A

Fig. 3: Evaluation of the proposed solutions (DRL, FDRL and FDRL2A).

model for the distributed agents. The coordinator also gets a reward value that corresponds to its decision. We derive in what follows the system state, the action space, the system reward as well as the learning process of the coordinator.

The system state is defined in a way to capture the feature characterizing the agents, which is reflected in their respective models. Therefore, the system state $\hat{s}_{\hat{t}}$ at the iteration round $\hat{t}$ is described as

$$\hat{s}_{\hat{t}} = [\theta^u_{\hat{t}}]_u. \tag{20}$$

The action space reflects the values to be selected for the weighting parameters, and then an action $\hat{a}_{\hat{t}}$ to be performed at an iteration round $\hat{t}$ is described as

$$\hat{a}_{\hat{t}} = [\beta^u_{\hat{t}}]_u. \tag{21}$$

The action will therefore allow to produce the aggregated model $\theta_{\hat{t}}$, using (19), which will be sent back to the agents.

The system reward is defined as value actions allowing to enhance the aggregation process. To this end, each agent $u \in \mathcal{U}$ computes at the end of each iteration round $\hat{t}$ the average reward $\hat{r}^u_{\hat{t}}$ achieved at this iteration. The system reward when applying the action $\hat{a}_{\hat{t}}$ on the state $\hat{s}_{\hat{t}}$ is defined as

$$\hat{\mathcal{R}}(\hat{s}_{\hat{t}}, \hat{a}_{\hat{t}}) = [\hat{r}^u_{\hat{t}} - \hat{r}^u_{\hat{t}-1}]_u. \tag{22}$$

As we can see, the reward increases only when the average reward of the associated agents progresses from the last iteration round. Maximizing the expected reward for the coordinator is therefore translated into selecting values for the weighting parameters that increase the rewards for the distributed agents.

Learning process allows building a model that selects optimal values for the weighting parameters. To this end, we consider the A2C algorithm which is a policy-based algorithm that directly parameterizes the policy π. Two deep neural networks are used in A2C: one is a dubbed actor used to approximate the agent policy, and another is a critic used to approximate the value function. We use $\ddot{\theta}_{\hat{t}}$ to denote the parameters of the actor network and use $\dot{\theta}_{\hat{t}}$ to denote the parameters of the critic network.

A2C also updates the parameters in the direction $\nabla log(\pi(\hat{a}_{\hat{t}}|\hat{s}_{\hat{t}}))A_\pi(\hat{s}_{\hat{t}}, \hat{a}_{\hat{t}})$, which is an unbiased estimation of $\nabla\mathbb{E}[\sum_{k=0}^{\infty} \hat{\tau}^k \hat{\mathcal{R}}(\hat{s}_{\hat{t}+k}, \hat{a}_{\hat{t}+k})]$, where $A_\pi(\hat{s}_{\hat{t}}, \hat{a}_{\hat{t}})$ is the advantage value which is defined as

$$A_\pi(\hat{s}_{\hat{t}}, \hat{a}_{\hat{t}}) = Q_\pi(\hat{s}_{\hat{t}}, \hat{a}_{\hat{t}}) - V_\pi(\hat{s}_{\hat{t}}). \tag{23}$$

Furthermore, by considering the Bellman equation, the advantage can be expressed as

$$A_\pi(\hat{s}_{\hat{t}}, \hat{a}_{\hat{t}}) = \hat{\mathcal{R}}(\hat{s}_{\hat{t}}, \hat{a}_{\hat{t}}) + \hat{\tau} V_\pi(\hat{s}_{\hat{t}+1}) - V_\pi(\hat{s}_{\hat{t}}), \tag{24}$$

where $\hat{\tau} \in [0, 1]$ is the discount factor. The parameters of critic network are learned by minimizing the error of the value function as

$$\mathcal{L}^c(\dot{\theta}_{\hat{t}}) = \mathbb{E}[A_\pi(\hat{s}_{\hat{t}}, \hat{a}_{\hat{t}})^2]. \tag{25}$$

As for the actor network, the parameters are learned by minimizing the negative log likelihood scaled by the advantage as

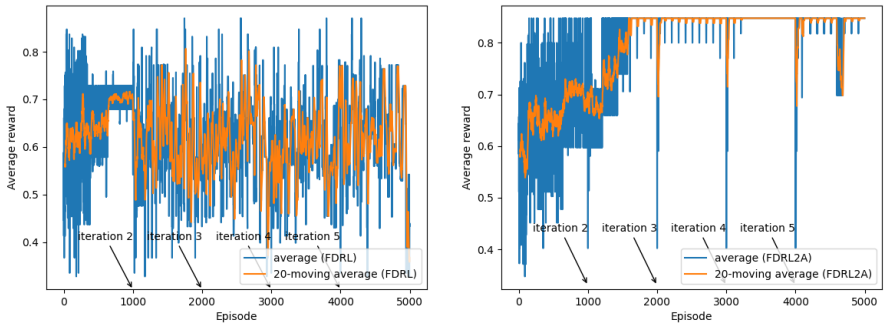
$$\mathcal{L}^a(\ddot{\theta}_{\hat{t}}) = \mathbb{E}[A_\pi(\hat{s}_{\hat{t}}, \hat{a}_{\hat{t}}) log(\pi(\hat{a}_{\hat{t}}|\hat{s}_{\hat{t}}))]. \tag{26}$$

V. Performance Evaluations

This section presents the results of the performance evaluations. The simulation considers a Nakagami parameter $m = 2$ and a noise variance N_0 of $-130dBm$ [14]. In addition, in order to limit the action space, the detection area for a UAV is limited to a zone of $500m \times 500m$, 2 MNOs, 4 BSs per MNO and 8 UAVs.

Three Deep reinforcement learning solutions have been implemented; In addition to the proposed FDRL2A approach, we have also implemented the DRL solution presented in Section III and also a FDRL solution where the model aggregation performed by the coordinator is based on averaging the models of the different agents. The two implementations, DRL and FDRL, are used as baselines for our proposed approach. The comparison is based on the archived reward values by the different solutions. Note that the achieved outage probabilities can directly be derived from those of the reward values, as the later is expressed as $1 - P^{out}_{uv_o}$ (See equation (11)).

We have first compared the proposed FDRL2A solution with the DRL and the FDRL solutions. The results of the evaluations are depicted in Fig. 3. As we can see, the three approaches are able to learn optimal strategies allowing to enhance the reward values. This is directly translated into selecting optimal MNO for each UAV in a way to reduce



(a) FDRL (b) FDRL2A

Fig. 4: Evaluation of the proposed FDRL2A against FDRL with 3 malfunctioning agents.

the outage probability. In the case of the DRL solution, one centralized agent agent is considered to collect the experiences, learn a model and take decision. However, in the case of the proposed FDRL2A approach and also the FDRL solution, distributed agents (corresponding to the different UAVs) take individual decisions without sharing their experiences. At the end of each iteration (5 iterations are considered in the evaluation), the coordinator aggregates the models of the agents. We can see from the conducted evaluation that the reward decreases for the first episodes after each model aggregation (Fig. 3 (b) and Fig. 3 (c)). However, this only lasts for a couple of episode then increases again.

Furthermore, in order to show added-value of the FDRL2A approach against the FDRL solution, we have performed another evaluation where some agents send random and unlearned models to the coordinator (case of malfunctioning agents). This directly affects the aggregated model produced by the coordinator. The obtained results are depicted in Fig. 4. As we can see, while the consideration of 3 malfunctioning agents has induced the coordinator to produce an aggregated model that led to unstable reward values in the FDRL solution, the proposed FDRL2A approach showed better results. This is due to the fact that the coordinator agent in the proposed FDRL2A solution learned optimal strategies to select the values of the weighting parameters based on the agents' models, whereas the FDRL solution performs averaging-based aggregation. This proves that the proposed approach can still operate even with the presence of malfunctioning agents.

VI. Conclusion

This paper investigated the traffic steering for cellular-enabled UAVs. The paper advanced the approach of FDRL2A, where the coordinator of a FDRL environment implements a DRL algorithm to learn optimal strategies to perform the aggregation. The simulation results illustrate that the proposed FDRL2A approach can achieve the similar performance as the DRL and FDRL solutions. Remarkably, in comparison with DRL, the proposed FDRL2A can avoid sharing their learned experiences among the agents. Furthermore, the evaluations against a FDRL solution show that the proposed FDRL2A can learn optimal strategies to perform model aggregation, even with the presence of malfunctioning agents.

References

[1] H. Hellaoui, M. Bagaa, A. Chelli, and T. Taleb, "Joint sub-carrier and power allocation for efficient communication of cellular uavs," *IEEE Transactions on Wireless Communications*, vol. 19, no. 12, pp. 8287–8302, 2020.

[2] X. Guan, Y. Huang, and Q. Shi, "Joint subcarrier and power allocation for multi-uav systems," *China Communications*, vol. 16, no. 1, pp. 47–56, 2019.

[3] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Optimal transport theory for power-efficient deployment of unmanned aerial vehicles," in *2016 IEEE International Conference on Communications (ICC)*, May 2016, pp. 1–6.

[4] X. Liu, L. Li, F. Yang, X. Li, W. Chen, and W. Xu, "Price-based power allocation for multi-uav enabled wireless networks," in *2019 28th Wireless and Optical Communications Conference (WOCC)*, 2019, pp. 1–5.

[5] S. Sharafeddine and R. Islambouli, "On-demand deployment of multiple aerial base stations for traffic offloading and network recovery," *Computer Networks*, vol. 156, pp. 52–61, 2019.

[6] Z. Chang, W. Guo, X. Guo, and T. Ristaniemi, "Machine learning-based resource allocation for multi-uav communications system," in *2020 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2020, pp. 1–6.

[7] U. Challita, W. Saad, and C. Bettstetter, "Interference management for cellular-connected uavs: A deep reinforcement learning approach," *IEEE Transactions on Wireless Communications*, vol. 18, no. 4, pp. 2125–2140, 2019.

[8] H. Huang, Y. Yang, H. Wang, Z. Ding, H. Sari, and F. Adachi, "Deep reinforcement learning for uav navigation through massive mimo technique," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 1, pp. 1117–1121, 2020.

[9] S. Fu, Y. Tang, Y. Wu, N. Zhang, H. Gu, C. Chen, and M. Liu, "Energy-efficient uav enabled data collection via wireless charging: A reinforcement learning approach," *IEEE Internet of Things Journal*, pp. 1–1, 2021.

[10] H. Hellaoui, A. Chelli, M. Bagaa, and T. Taleb, "Efficient steering mechanism for mobile network-enabled uavs," in *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1–6.

[11] H. Hellaoui, B. Yang, and T. Taleb, "Towards using deep reinforcement learning for connection steering in cellular uavs," in *2021 IEEE Global Communications Conference (GLOBECOM)*, 2021, pp. 01–06.

[12] H. Hellaoui, A. Chelli, M. Bagaa, and T. Taleb, "Towards mitigating the impact of uavs on cellular communications," in *2018 IEEE Global Communications Conference (GLOBECOM)*, Dec 2018, pp. 1–7.

[13] J. Qi, Q. Zhou, L. Lei, and K. Zheng, "Federated reinforcement learning: Techniques, applications, and open challenges," *arXiv preprint arXiv:2108.11887*, 2021.

[14] A. F. Molisch, *Wireless Communications*. Chichester: John Wiley & Sons, 2005.