ICC 2022 - IEEE International Conference on Communications | 978-1-5386-8347-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICC45855.2022.9838773

Joint Optimization of UAV Trajectory and Task Allocation for Wireless Sensor Network Based on O-RAN Architecture

Chuan Pham*, Kim Khoa Nguyen*, Mohamed Cheriet*,

*Synchromedia - École de Technologie Supérieure, Université du Québec, H3C1K3, Canada,

***Abstract*—Unmanned aerial vehicles (UAVs) are increasingly deployed as flying base stations to serve various applications, such as smart agriculture, emergency healthcare system, smart transportation, etc, thanks to their advantages of flexible movement, strong wireless communication, and heavy payload capability. To facilitate the deployment of the fifth-generation (5G) networks, the Open Radio Access Network (O-RAN) has presented a distributed architecture for terrestrial and non-terrestrial networks. Unfortunately, O-RAN architecture for wireless sensor networks is still in development. In this paper, we investigate a 5G integration of multi-flying base stations in a wireless sensor network using O-RAN. Specifically, we formulate a joint optimization problem of UAV trajectory and resource allocations to process sensing data, named UTRA. We use decomposition to address it based on two solvable sub-problems and provide learning methodologies solve UTRA based on the multi-agent reinforcement learning and online learning methods, both of which are well supported by the O-RAN architecture. Our extensive numerical simulations show that our proposed approaches are efficient in a variety of settings and validation scenarios.**

***Index Terms*—Unmanned aerial vehicles, Open Radio Access Network (O-RAN), Resource allocation.**

I. Introduction

Unmanned aerial vehicles (UAVs) have emerged as a viable option for dealing with a high number of connections in the Internet of Things (IoT) network, such as monitoring device connections, smart traffic sensor connections, smart farming sensor connections, and so on. In research and industry, the usage of UAV base stations (UAV-BSs) is especially tempting to fulfill the ever-increasing thirst for diverse demand in a range of unexpected and temporary conditions that cannot be provided by the present architecture [1]. UAVs with sophisticated computer resources may be used to service ground users, enhancing wireless communication between users and UAV-BSs. These qualities distinguish UAVs from standard BSs and make them really feasible for low-cost deployment in unexpected operation scenarios [1], [2].

In this research, we study the released Open Radio Access Network (O-RAN) [3] architecture to deploy a flying 5G O-RAN architecture to gather data and serve sensing tasks to assist the wireless sensor network, as illustrated in Fig. 1. As shown in [3], the performance of radio access networks

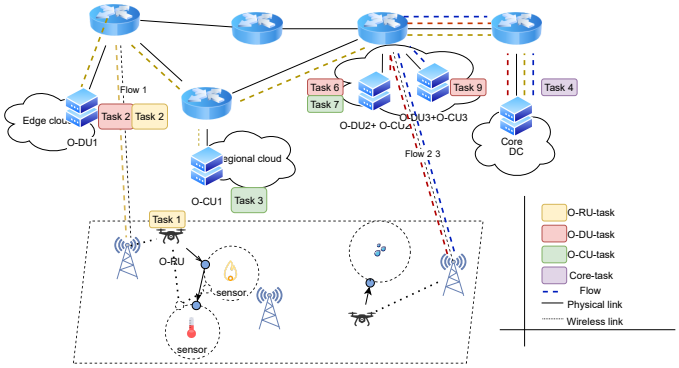


Fig. 1: Overview architecture of U-ORAN.

can be improved significantly by leveraging virtual network functions (VNFs) in a flexible design with open interfaces to vendor-neutral hardware and software-defined technologies. Unfortunately, few studies have researched the combination of strong skills of UAV-BSs and O-RAN to supply a dispersed and flying O-RAN among the different issues of UAVs that have been explored recently. Furthermore, O-RAN 's intelligent design will provide strong support for the UAV-BS system, in which learning technologies may be used to aid many network applications from UAVs to core networks.

The multi-objective joint optimization of trajectory and resource allocation in the UAV communication system is provided in [4]–[6]. However, the computing resources to serve offloading jobs, such as the association decision, transmission power, and the amount of CPU to serve for each offloading work, are largely investigated to run UAV-BSs. Unlike previous studies, such as [4], [5], [7], ours is the first to apply learning solutions to the flying O-RAN architecture to optimize not only the UAV-BS trajectory for gathering sensing data but also the distributed resources, such as the O-RAN Radio Unit (O-RU), O-RAN Distributed Unit (O-DU), and O-RAN Central Unit (O-CU) to serve offloading tasks requested by sensing devices.

The contributions of our work are as follows: (i) investigating the *U-ORAN system model* in the wireless sensor network

to formulate a *joint UAV trajectory and resource allocation problem (UTRA)*, which can optimize the network utility formulated under the communication data rate and resource allocation; (ii) constructing *a control loop* to control the UAV-BSs and allocate the dispersed resources for serving sensor's tasks (iii) and decomposing UTRA, into sub-problems, which is solvable by using *multi-agent reinforcement learning and online learning and matching approaches*.

II. System model and Problem statement

A. System model

We analyze a system in which O-RAN architecture with flying BSs are placed to serve ground sensors located at specified coordinates during T time slots. The U-ORAN system model is shown in Fig. 1. We consider that a UAV can carry a container-based network function due to their payload capacity [8]. Connecting to O-RU, an O-RAN distributed unit (O-DU) is a logical node that hosts RLC/MAC/High-PHY layers based on a lower layer functional split in O-RAN. The radio resource control (RRC) and Packet Data Convergence Protocol (PDCP) levels are operated by the O-RAN Central Unit (O-CU), which controls a series of eNB/gNB functions. An O-CU can link to numerous O-DUs in order to support multiple gNBs, according to the O-RAN standards. Given the set of containers with different functions, we aim to determine the optimal UAV-sensor associations and task offloading scheme.

Here, we use $\mathcal{D}$, $\mathcal{C}$ and $\mathcal{R}$ to denote the set of O-DU, O-CU, and containers on core data centers, respectively. Furthermore, we denote the set of UAV-BSs by $\mathcal{V}$. Hence, we denote the set of computing resources in our O-RAN-based model by $\mathcal{N} = \mathcal{V} \cup \mathcal{D} \cup \mathcal{C} \cup \mathcal{R}$.

B. Problem formulation

1) UAV model: We first consider to deploy a set $\mathcal{V}$ of UAV-BSs in a specific area with $W \times W$ square during a specific period T. According to [9], the energy consumption of UAV-BS v is calculated by $E_v^{fly} = \sum_{t=1}^{T-1} 0.5\kappa_v \frac{||o_v(t)-o_v(t-1)||}{\Delta_t}$, where κ_v is the payload of UAV-BS v and $o_v(t) = (x_v(t), y_v(t), z_v(t)), 0 \le x_v(t), y_v(t) \le W$ is the coordinate of UAV v at time t.

We consider a set $\mathcal{I} = \{i\}$ of terrestrial sensor nodes in this area. The air-to-ground channel between a UAV and a sensor node depends on the LoS component and Non-LoS component [2]. Following [6], the probability of LoS of a communication channel between UAV v and sensor i depends on the elevation angle θ and it is calculated as follows $P_{iv}^{LoS} = \frac{1}{1+\alpha \exp(-\psi_1[\theta-\psi_2])}$, where ψ_1 and $\psi 2$ are terrain parameters depending on the environment (e.g., urban, rural) and $\theta = \arctan(h_v/d_{iv})$ with the UAV elevation h_v and UAV-BS-sensor distance d_{iv}. Based on P_{LoS}, the average path loss from a UAV to a sensor can be calculated as

$$\begin{cases} \bar{L}_{iv} = P_{iv}^{LoS}(\theta) L_{iv}^{LoS}(d_{iv}) + [1 - P_{iv}^{LoS}(\theta)] L_{iv}^{NLoS}(d_{iv}) \\ L_{iv}^{LoS} = 20\log(4\pi f d_{iv}/c) + \eta_{LoS} \\ L_{iv}^{NLoS} = 20\log(4\pi f d_{iv}/c) + \eta_{NLoS} \end{cases}$$

η_{LoS} and η_{NLoS} are the average additional loss.

To measure the signal quality that a sensor could experience, we use γ_{iv} to denote the signal to noise ratio (SNR) for the UAV-sensor with the elevation angle θ_v and the distance d_{iv} from a certain UAV. Hence, we have $\gamma_{iv} = (P_i - \bar{L}_{iv})/N_0$, where P_i is the transmission power of sensor i and N_0 is the power of background noise. We define the association variable between UAV v and sensor i at time t by $a_{iv}(t)$. Based on the Shanon calculation, we can obtain the data rate by $w_{iv}(t) = a_{iv}(t) B \log_2(1 + \gamma_{iv}(t))$, where B is the channel bandwidth.

We denote a monitoring data offloaded from a node i by a tuple (λ_i, c_i, l_i) where λ_i is the data size, c_i is the amount of computing resources requested by device i (i.e., the number of the required CPU cycles for executing 1-bit of input data) and L_i is the tolerant latency. Hence, if sensor i communicates with UAV v to transfer its data, it requires a serving time by $\frac{\lambda_i}{w_{iv}}, \forall i \in \mathcal{I}, \forall v \in \mathcal{V}$. Thus, we calculate the hovering energy consumption of UAV v as follows $E_v^{hov}(t) = \sum_{i \in \mathcal{I}} a_{iv}(t) \frac{\lambda_i}{w_{iv}(t)} \psi_v$, where ψ_v is the amount of hovering energy consumed in a unit time period.

2) O-RAN computing resource model: To make things easier, we view each sensor node as requesting a single set of monitoring data (called a task). Hence, given the present location of the UAV at time t, there is a set $\mathcal{I}$ of requested tasks. The O-RAN elements (O-RU, O-DU, O-CU, and core network containers) are connected as a network graph $(\mathcal{N}, \mathcal{E})$, with $\mathcal{E}$ denoting the set of links. In order to make the resource allocation decision, we define $x_{in}(t)$ as a probability variable to represent the probability of task $i \in \mathcal{I}$ that is executed at container $n \in \mathcal{N}$ at time t.

Resource capacity. A node can only serve offloading task depending its availability of computing resources. Thus, if task i is executed by container n, the allocation has to satisfy the following resource constraint $\sum_{i \in \mathcal{I}} \lambda_i c_i x_{in}(t) \le \mu_n, \forall n \in \mathcal{N}$, where μ_n is the service rate of container n.

Propagation latency. We take into account the propagation delay when the offloading task i is routed from sensor node i to the executing container. Suppose that UAV v can pre-calculate the backhaul route from v to a container n based on the shortest path algorithm. Thus, we denote the propagation latency between v and n by l_{vn}.

For simplicity, we assume that the sensing data is routed through a single path from the source to the destination. Thus, we have the propagation latency as follows $l_{iv} a_{iv}(t) + l_{vn} x_{in}(t) \le l_i, \forall i \in \mathcal{I}$, where l_{iv} is the propagation latency between sensor i and UAV v. Suppose the packet size is given, thus the propagation latency l_{iv} calculation only depends on the data rate $w_{iv}(t)$.

3) Objective function and problem formulation: Our goal in this work is to optimize total network utility, which may be translated into resource allocations (e.g., computing resources and connection data rate), subject to the aforementioned constraints. As follows, we represent $u_i(.)$ as the utility function for each task i based on the association decision and the selected container n

$$u_i(\boldsymbol{w_i}, \boldsymbol{x_i}) = \alpha \sum_{n \in \mathcal{N}} P_{in} \lambda_i c_i x_{in}(t) + \beta \sum_{(mn) \in \mathcal{E}} w_{(mn)}^i(t),$$

where P_{in} is the given probability of container n that can serve task i, α and β are weighted parameters, and $w^i_{(mn)}(t)$ is the routing rate of task i on link (mn). For sake of notation, we also use $w^i_{(mn)}(t)$ for the rate $w_{iv}(t)$ on wireless link (iv).

Hence, we formulate the joint multi-UAV Trajectory and Offloading Task allocation optimization (UTRA) problem for the UAV-ORAN based architecture as follows:

$$\max \quad \sum\nolimits_{t=0}^{T-1} \sum_{i\in\mathcal{I}} u_i(\boldsymbol{x_i}, \boldsymbol{w_i}) \tag{1a}$$

$$\text{s.t.} \quad \sum\nolimits_{n\in\mathcal{N}} x_{in}(t) = 1, \forall i \in \mathcal{I}, \tag{1b}$$

$$\sum\nolimits_{v\in\mathcal{V}} a_{iv}(t) = 1, \forall i \in \mathcal{I}, \tag{1c}$$

$$\sum\nolimits_{i\in\mathcal{I}} \lambda_i c_i x_{in}(t) \le \mu_n, \forall n \in \mathcal{N}, \tag{1d}$$

$$\sum\nolimits_{i\in\mathcal{I}} w^i_{(mn)}(t) \le C_{uv}, \forall (uv) \in \mathcal{E}, \tag{1e}$$

$$\sum\nolimits_{t=0}^{T-1} E_v^{fly}(t) + E_v^{exe}(t) + E_v^{hov}(t) \le \bar{E}_v, \tag{1f}$$

$$l_{iv}a_{iv}(t) + l_{vn}x_{in}(t) \le l_i, \forall i \in \mathcal{I}, \forall i \in \mathcal{I}, \tag{1g}$$

$$\sum\nolimits_{t=0}^{T-1} \sum_{i\in\mathcal{I}} \sum_{u\in\mathcal{U}} a_{iu}(t) = |\mathcal{I}|, \tag{1h}$$

where (1f) is to ensure the safety level of the battery of v with the execution power consumption of UAV v calculated by $E_v^{exe}(t) = \sum_{t=0}^{T-1} \sum_{i\in\mathcal{I}} \lambda_i p_i x_{in}(t)$. The amount of energy required to process 1-bit data is denoted by p_i. In addition, (1h) ensures that all of the sensor nodes that will be covered.

III. Proposed framework

We propose a learning control loop to address this joint problem, UTRA, which includes two procedures: i) *the UAV-based control procedure to optimize UAV trajectory*, and ii) *the offloading task allocation procedure to distribute offloading tasks to appropriate executing containers*

A. Overview architecture

An architecture is suggested to address the joint problem based on the general learning architecture O-RAN [10]. The proposed architecture's flow is made up of following operations. First, data is gathered and assessed at the Non-RT RIC before learning models are built. After training and testing, these models are stored and supplied to xApps for use in the Near-RT RIC. Container images can be easily stored, published, and launched at different points across the system. We run two learning models (UAV-based control and resource allocation) on separate xApps.

B. Stochastic game model for UAV-based Control

To make the association between UAVs and sensor devices, we first address the UAV-based controlling sub-problem.

$$\textbf{P1:} \max \quad \sum_{t=0}^{T-1} \sum_{i\in\mathcal{I}} \sum_{v\in\mathcal{V}} \beta w_{iv}(t), \tag{2}$$

$$\text{s.t.} \quad \text{(1c), (1f), (1g), (1h).}$$

Because the sub-problem **P1** solely examines communication between UAVs and sensor, the routing variable may be expressed as $w_{iv}(t)$, which represents the uplink data rate from sensor i to UAV v, which is mostly determined by the distance between UAVs and device i.

To solve **P1**, we advocate a multi-agent reinforcement learning framework which is based on the Markov decision problem (MDP).

Definition 1. A stochastic multi-UAV control model can be defined by $(\mathcal{V}, \mathcal{S}, \mathcal{A}, R, T)$, where $\mathcal{S}$ is the discrete set of environmental states that consists of the possible status of all agents, $\mathcal{A} = A_1 \times A_2 \times ...A_{|\mathcal{V}|}$ is the joint action set of all agents, $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the reward function and $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition function.

An agent v makes an action based on a strategy $\pi_v : \mathcal{S} \to \mathcal{A}_v$, where $\mathcal{A}_v$ is the action set of UAV v. Similar to the single-agent MDP, each agent aims to find a policy $\pi_v : \mathcal{S} \to \mathcal{A}_v, \forall v \in \mathcal{V}$. Suppose this policy is stationary, hence, we have

$$V_{\pi,v}(s) = E_\pi[r_v(t)|s_0 = s], \forall s \in \mathcal{S}$$

where $r_v(t)$ is the reward at time t for the agent v. In addition, the Q-value for each state-action is

$$Q_{\pi,v}(s, \{a_v\}_{v\in\mathcal{V}}) = r_v(s, a_{vv\in\mathcal{V}}) + \varrho \sum_{s'} P(s, \{a_v\}_{v\in\mathcal{V}}, s')V_{\pi,v}(s).$$

Definition 2. A joint policy $\{\pi_v\}_{v\in\mathcal{V}}$ is a Nash equilibrium if each π_v is the best response to other UAVs.

We consider the reward function of UAV v by

$$r_v(t) = \alpha \sum_{i\in\mathcal{I}} u(\boldsymbol{x_i}, \boldsymbol{w_i}) - \beta(E_v^{fly}(t) + E_v^{exe}(t) + E_v^{hov}(t)) + M,$$

where M is large enough.

Cooperative approach to joint actions. In this stochastic game, a mix joint strategy $\pi_v : S_v \to A_v$ is the mapping from the state set S_v to the action set A_v. In detail, UAV v at time t and state s_v has a mixed strategy $\pi_v(s_v) = \{\pi_v(s_v, a_v)|a_v \in A_v\}$. A joint strategy $\pi = \{\pi_v(s_v)\}_{v\in\mathcal{V}}$ is a vector strategy of all UAVs. Consider for a long-term reward of UAV v, we have $\mathbf{r}_v = \sum_{t=0}^{+\infty} \varrho r_v(t) = E\big(\sum_{t=0}^{+\infty} \varrho r_v(t)|s_v(0)\big)$, where $E(.)$ is the expectation function. Since the UAVs have different expected rewards depending on the joint strategy of all UAVs, each cannot maximize its expected reward.

Definition 3. If a joint strategy $\pi^* = \{\pi_v^*\}_{v\in\mathcal{V}}$ can maximize all the expected rewards of all UAVs, then this collection of strategies is a Nash equilibrium (NE). It means that $\mathbf{r}_v(\pi_v^*) \ge \mathbf{r}_v(\pi_v'), \forall \pi_v'$.

Deep Q-learning approach. Following [11], we investigate a deep Q-network (DQN) approach in which the convolution neural network (CNN) [11] is used to represent the input state s and the output of this neural network indicates all possible actions with the given input state s. In general, DQN can approximate the optimal policy in the Q-learning approach.

Following the updating Q-values (3), the extended DQN approximates Q-values by $Q(s, a|\omega)$, where ω is the weight parameter of the neural network. In detail, this parameter is updated to minimize the loss function at iteration k:

$$L(s, a|\omega_k) = (r(s, a) + \varrho \max_a Q(s', a|\omega_k) - Q(s, a|\omega_k))^2.$$

Algorithm 1: UAV controlling algorithm with multi-agent Q-learning approach.

```
1  Initialization: Set t = 0;
2  forall v ∈ V do
3      Inittialize Q_v^t;
4      Initialize the action value a_v(0), the strategy π_v(0) and
         the state s_v(0);
5  end
6  while t ≤ T − 1 do
7      forall v ∈ V do
8          Measure the level of SNR;
9          Update state s_v(t);
10         Forward observation (s_v(t), A_v) to the controller
             where Ā_v is the projection actions corresponding
             to state s_v;
11     end
12     Select a joint action {a_v}_{v∈A} corresponding to the NE
         value;
13     Forward actions to all agents;
14     Update reward r_v(t) by (III-B);
15     Update Q_v^{t+1} by(3);
16     Update t = t + 1
17 end
```

Algorithm 2: UAV controlling algorithm with multi-agent DQN approach.

```
1  Initialization: Set t = 0;
2  forall v ∈ V do
3      Inittialize Q_v^t;
4      Initialize the action value a_v(0), the strategy π_v(0) and
         the state s_v(0);
5  end
6  while t ≤ T − 1 do
7      forall v ∈ V do
8          Measure the level of SNR;
9          Update state s_v(t);
10         Forward observation (s_v(t), A_v) to the controller
             where Ā_v is the projection actions corresponding
             to state s_v;
11         Gather observeration of other UAVs;
12         Select the best Q-value based on the output of the
             neural network;
13     end
14     Update reward r_v(t) by (III-B);
15     Update Q_v^{t+1} by(3);
16     Update t = t + 1
17 end
```

The value at iteration $k+1$ is as follows $\omega_{k+1} = \omega_k + \nu \nabla_{\omega_k} L(\omega_k)$, where ν is the learning rate.

To apply DQN in our work, we take the advantage of the convolution neural network to encode the environment. We follow the typical DQN architecture and apply it to our problem. We use the image size $3 \times W \times L$ with W and L denoted as the width and the length of our two-dimensional map and 3 snapshot channels related to sensor information (i.e., sensor's data rate), states of all UAVs excluding v and the current state of UAV v. The output layer of the network represents the Q-values for all the actions that can be taken by the UAV. This design allows us to solve **P1** in a distributed manner in which each UAV is controlled by a controller.

C. Online learning and matching approach

Given the solution of **P1** including the current location of UAVs and the association between UAVs and devices, we formulate the task allocation sub-problem as follows.

$$\textbf{P2:} \max \quad \sum_{t=0}^{T-1} \sum_{i \in \mathcal{I}} \alpha \sum_{n \in \mathcal{N}} P_{in} \lambda_i c_i x_{in}(t) + \beta \sum_{(mn) \in \mathcal{E}} w^i_{(mn)}(t),$$

$$\text{s.t.} \quad \text{(1b), (1c), (1d), (1e) and (1g).}$$

1) Online learning and matching based method: The formulation of **P2** can be seen as an online matching problem where offloading tasks refers to buyers and the set of containers can be mapped to the set of items. Note that we relax constraints (1e) and (1g) of **P2** to find the optimal container and then we exploit the shortest paths to forward the requests.

Online greedy approach. In this scenario, there is a simple strategy that each offloading task i at time t tries to optimize its benefit by solving its following problem.

$$\max \quad \sum\nolimits_{n \in \mathcal{N}} u_i(x_{in}) \tag{3a}$$

$$\text{s.t.} \quad \sum\nolimits_{n \in \mathcal{N}} x_{in} = 1, \tag{3b}$$

$$\sum\nolimits_{i \in \mathcal{I}} \delta_i c_i x_{in} \leq \mu_n, \forall n \in \mathcal{N} \tag{3c}$$

Calculating the available resources of each container n to service task i at arrival time t can yield the preference value $P_{in}(t)$. This online greedy approach is straightforward, but it cannot optimize the goal over time because it just utilizes the best strategy for the current time frame.

2) Exploring upper confidence bound and greedy value approach: To make a matching decision, this approach examines a strategy based on a mix of the upper confidence bound [12] and greed values. Let denote **D** as a binary reward matrix, with each item $D_{in}(t)$ indicating whether task i is served by container n. When the task arrives at time t, we use $N_{in}(t)$ to represent the number of selections made by task i when selecting container n, and $R_{in}(t)$ to represent the amount of rewards earned by assigning task i. $\bar{R}_{in}(t) = R_{in}(t)/N_{in}(t)$ is the average reward. The UCB function is as follows, based on the Upper Confidence Bound (UCB) algorithm in Reinforcement Learning:

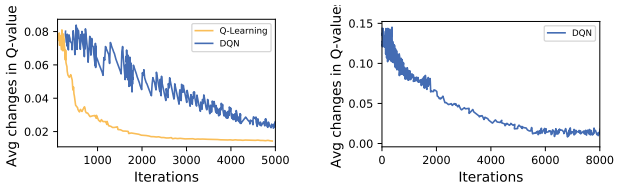
$$UCB_{in}(t-1) = \begin{cases} \infty & N_{in}(t-1) = 0, \\ \bar{R}_{in}(t-1) + \sqrt{\frac{3\log(t)}{2N_{in}(t-1)}} & \text{otherwise.} \end{cases} \tag{4}$$

and the reward is updated by $R_{in}(t) = R_{in}(t-1) + D_{in}$.

We propose an online learning and matching algorithm for the offloading task allocation presented in Alg. 3. In more detail, when a task is received, the controller will classify the task type in order to make an appropriate decision based on the kind of containers. If the preference changes, the task will be forwarded to a container having the highest UCB value, (Lines 5-7); otherwise, we can solve the problem based on the given preference at $t-1$ (Lines 8-9).

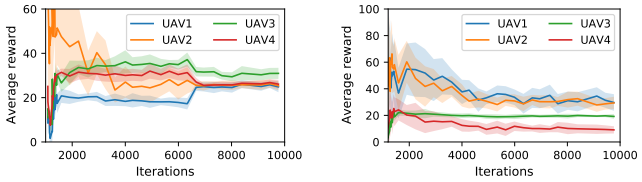
Algorithm 3: Online learning and matching solution for P2.

Initialization: Task arrivals $t = 1; ...; T$, number of task types m, number of containers n, initial matrix $\mathbf{P}$;
for $t = 1, ..., T$ **do**
  Observe the type of this task: j;
  Calculate UCB based on (4);
  if $P(t)$ *is changed* **then**
    Assign task to the container that has a maximum UCB;
  end
  else
    Solve (3) to assign task based on $P(t-1)$;
  end
  Update $P_{ij}(t) = R_{ij}(t)/N_{ij}(t)$;
  Project the solution to constraint (1e) and (1g);
  if *(1e) and (1g) are sastified* **then**
    Go to Step 2;
  end
  else
    Remove the selected container out of the input;
    Go to Step 3;
  end
end



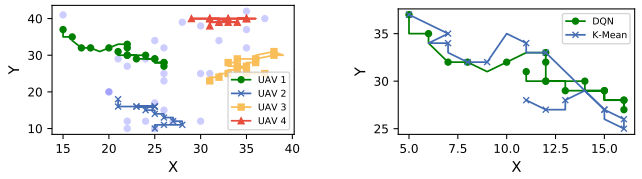
(a) Small-scale setting. (b) Large-scale setting.

Fig. 2: Changes of Q-values.

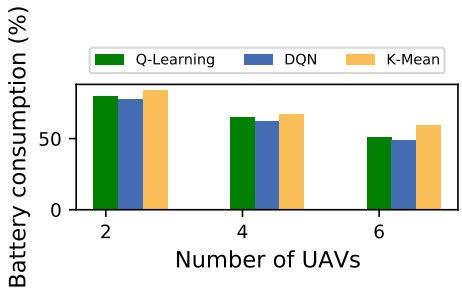


(a) Q-learning. (b) DQN.

Fig. 3: Average reward.



(a) Trajectory of UAVs. (b) Trajectory comparison.



(c) Battery consumption.

Fig. 4: UAV controlling evaluation.

IV. Numerical Results

A. Network settings

To design the distributed and flying U-ORAN network, we use Python and the Keras framework to establish a simulation environment. The sensor devices are generated at random in a 500m $\times$ 500m region. We undertake an ideal configuration for the elevation of UAVs to simplify our model with a fixed elevation determined by an optimal height in the range [100m, 125m] relying on our current environment. The speed of the UAVs is set to 10 m/s, with each time period lasting 0.1s. we generate sensing tasks at random based on the number of sensors with a latency threshold of [30ms, 150ms] for each task. The amount of the offloading tasks' desired data is set in the range of [50 to 100] $\times 10^4$ bits, and the processors' necessary cycles are set in the range of [5, 20] cycles.

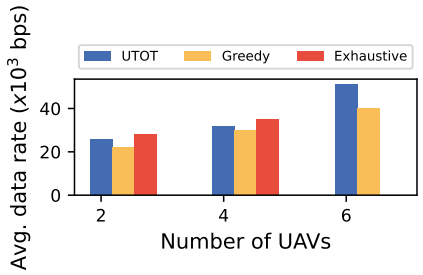
B. Results

1) Convergence: We first evaluate the convergence of our proposed algorithm. We compare the convergence of multi-agent Q-learning and DQN approaches in the small-scale setting with 4 UAVs. In such a setting, we are able to handle the Q-table of a joint actions of all UAVs. The change of Q-values in which the Q-learning approach converges faster than DQN after 2000 iterations is shown in Fig. 2. In a large-scale environment, Fig. 2b depicts DQN convergence approximately 8000 training rounds. In addition, we show the average reward of various strategies over the training time. The average reward of Q-learning and DQN are plotted in Fig. 3. The rewards of UAVs have a small gap (Fig. 3a) when compared to the reward in DQN when a Nash equilibrium action is chosen. DQN tries to optimize for each UAV, resulting in a differentiated average reward (Fig. 3b).

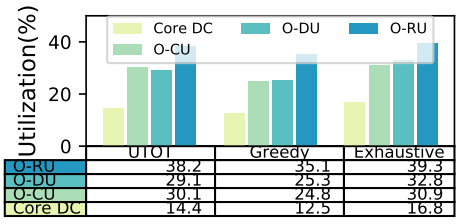
2) Evaluation of controlling UAV-BSs:

Trajectory of UAVs. The trajectory of all UAVs utilizing DQN is captured in Fig. 4a, where the blue dots indicate sensor positions. The graphic depicts UAVs flying in a proper trajectory to cover ground devices. In addition, we explore the baseline K-Mean technique, in which we cluster the sensors and then use the branch and bound algorithm to solve the traveling salesman problem (TSP) in each cluster. The offloading task decision in K-Mean is implemented following the random strategy. The trajectory of the chosen UAV via the DQN approximately follows the movement of this baseline, as seen in Fig. 4b.

Battery consumption. To illustrate the energy economy of our method, we compare Q-learning, DQN, and K-Mean methodologies as shown in Fig. 4c. We only use four UAVs to accomplish Q-learning because of its high complexity. In all three instances, Q-learning and DQN had lower battery usage

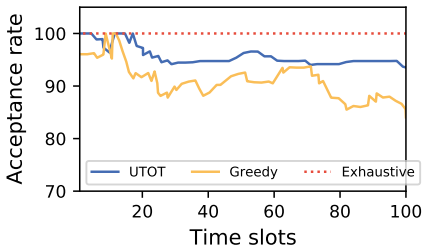


(a) Offloading rate.



	UTOT	Greedy	Exhaustive
O-RU	38.2	35.1	39.3
O-DU	29.1	25.3	32.8
O-CU	30.1	24.8	30.9
Core DC	14.4	12.5	16.8

(b) Resource utilization.



(c) Acceptance rate.

than K-Mean. On average of three cases, Q-learning and DQN consumes 65.33% and 62.9% of battery, respectively while K-mean spends more than 70% of that. Although K-Mean technique can find the shortest path across all sensors, they are not aware to offloading tasks, which have a substantial impact on UAV energy consumption.

3) Evaluation of task allocation: In this part, we consider the resource utilization and total utility in the system to evaluate the performance of the proposed algorithm.

Data rate evaluation. We scale the number of base stations to show the average offloading data rate as shown in Fig. 5a. The average offloading rate of our technique rises dramatically when we use additional UAV-BSs, from 26×10^3bps to 51×10^3bps, corresponding to 2 and 6 UAVs, respectively. Such a result cannot be obtained by Greedy.

Resource utilization. In Fig. 5b, we display the system's resource use in several approaches. The resource usage of O-RU and O-DU containers is higher than that of O-CU and core DCs due to their small configurations. When compared to Greedy baselines, UTRA consumes more resources, which may be justified by its higher acceptance rate, as seen in Fig. 5c. UTRA has a 96 percent acceptance rate, whereas the Greedy baseline has 91 percent acceptance rate. In all types of containers, the difference in resource use between UTRA and Exhaustive baseline is not considerable. The Greedy has an inconsistent acceptance rate, resulting in the lowest resource consumption. The Exhaustive baseline is used to filter all of the infeasible scenarios that have no solution. Therefore, we choose this baseline as the comparison's highest standard.

V. Conclusion

This study looked into the possibilities of using a flying O-RAN architecture for sensing tasks in the wireless sensor network. The suggested model and architecture might be a feasible solution for deploying flying base stations in the network and serving sensing functions under the O-RAN architecture. We contributed to model the combined multi-UAV trajectory and resource allocation issue, UTRA, using this architecture. We also illustrated that solving UTRA using a combination of reinforcement learning and online learning methodologies is a promising approach. The performance of the recommended methods was then evaluated using extensive numerical simulations with a variety of settings and features. In terms of resource use, utility values, and service acceptance rate, our recommended algorithms outperformed baselines.

Acknowledgment

The authors thank Mitacs, Ciena, and ENCQOR for funding this research under the IT13947 grant.

References

[1] B. Li, Z. Fei, and Y. Zhang, "Uav communications for 5g and beyond: Recent advances and future trends," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2241–2263, 2019.

[2] M. Mozaffari, W. Saad, M. Bennis, Y.-H. Nam, and M. Debbah, "A tutorial on uavs for wireless networks: Applications, challenges, and open problems," *IEEE Communications Surveys Tutorials*, vol. 21, no. 3, pp. 2334–2360, 2019.

[3] O-ran. [Online]. Available: https://www.o-ran.org/resources

[4] E. Kalantari, M. Z. Shakir, H. Yanikomeroglu, and A. Yongacoglu, "Backhaul-aware robust 3d drone placement in 5g+ wireless networks," in *2017 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2017, pp. 109–114.

[5] C. Qiu, Z. Wei, Z. Feng, and P. Zhang, "Joint resource allocation, placement and user association of multiple uav-mounted base stations with in-band wireless backhaul," *IEEE Wireless Communications Letters*, vol. 8, no. 6, pp. 1575–1578, 2019.

[6] H. Wang, H. Zhao, W. Wu, J. Xiong, D. Ma, and J. Wei, "Deployment algorithms of flying base stations: 5g and beyond with uavs," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10 009–10 027, 2019.

[7] M. F. Sohail, C. Y. Leow, and S. Won, "Non-orthogonal multiple access for unmanned aerial vehicle assisted communication," *IEEE Access*, vol. 6, pp. 22 716–22 727, 2018.

[8] O. Bekkouche, K. Samdanis, M. Bagaa, and T. Taleb, "A service-based architecture for enabling uav enhanced network services," *IEEE Network*, vol. 34, no. 4, pp. 328–335, 2020.

[9] S. Jeong, O. Simeone, and J. Kang, "Mobile edge computing via a uav-mounted cloudlet: Optimization of bit allocation and path planning," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 3, pp. 2049–2063, 2018.

[10] L. Bonati, S. D'Oro, M. Polese, S. Basagni, and T. Melodia, "Intelligence and learning in o-ran for data-driven nextg cellular networks," 2021.

[11] V. François-Lavet, P. Henderson, R. Islam, M. G. Bellemare, and J. Pineau, "An introduction to deep reinforcement learning," *Foundations and Trends® in Machine Learning*, vol. 11, no. 3-4, p. 219–354, 2018. [Online]. Available: http://dx.doi.org/10.1561/2200000071

[12] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: A Bradford Book, 2018.