



# Energy and Quality Aware Multi-UAV Flight Path Design Through Q-Learning Algorithms

Hend Zouaoui<sup>1</sup>, Simone Faricelli<sup>1</sup>, Francesca Cuomo<sup>1</sup>  ,  
Stefania Colonnese<sup>1</sup> , and Luca Chiaraviglio<sup>2,3</sup> 

<sup>1</sup> University of Rome La Sapienza, Rome, Italy  
{zouaoui.1800809, faricelli.1647406}@studenti.uniroma1.it,  
{francesca.cuomo, stefania.colonnese}@uniroma1.it

<sup>2</sup> University of Rome Tor Vergata, Rome, Italy  
luca.chiaraviglio@uniroma2.it

<sup>3</sup> Consorzio Nazionale Interuniversitario per le Telecomunicazioni (CNIT),  
Rome, Italy

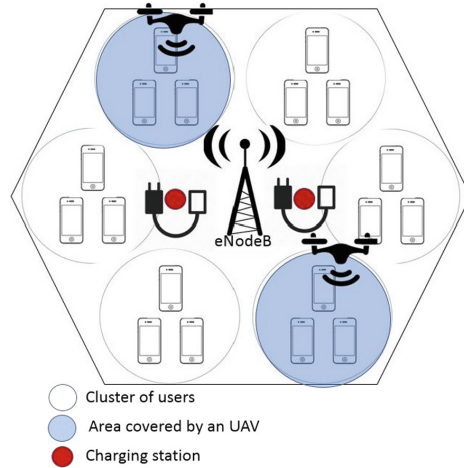
**Abstract.** We address the problem of devising an optimized energy aware flight plan for multiple Unmanned Aerial Vehicles (UAVs) mounted Base Stations (BS) within heterogeneous networks. The chosen approach makes use of Q-learning algorithms, through the definition of a reward related to relevant quality and battery consumption metrics, providing also service overlapping avoidance between UAVs, that is two or more UAVs serving the same cluster area. Numerical simulations and different training show the effectiveness of the devised flight paths in improving the general quality of the heterogeneous network users.

**Keywords:** Q-learning · UAV · Heterogeneous networks

## 1 Introduction

In mobile networks, the Quality of Experience (QoE) depends on the bandwidth request of users over space and time. Relying on fixed Base Stations (BSs) to satisfy the users bandwidth request may not comply with the fluctuating nature of that request [1]. In some particular cases, e.g. events when a large number of users is concentrated in the same area, or disasters affecting the network, the QoE drops dramatically. Employing UAVs as mobile network elements provides a possible solution to mitigate this effect [2].

In this work, we address the problem of planning the path of UAVs mounting eNodeB (eNB) functionality to provide support to the fixed BS and offer a constant good quality to users in an area where the request fluctuates in a cyclic fashion every 24 h. The use of UAVs as BSs has been advocated and discussed in several recent papers (see e.g., [3–5]). Different approaches can be adopted for computing the optimal deployment of the UAVs such as optimization or planning algorithms, for instance MILP in [10], and attractive approaches leveraging



**Fig. 1.** Example scenario: one eNB, 6 clusters, 2 UAVs and 2 CSs

machine learning techniques [6]. More in detail, the problem of planning the path of the UAVs has been previously addressed in [7], by employing the well established Q-learning algorithm [8]. However, the work in [7] is tailored to a single UAV, which poses limits to the applicability in a complex scenario composed of multiple UAVs. Moreover, the method in [7] does not take into account the need to recharge the UAV battery and the case in which many UAVs are collaborating. To overcome these important issues, in this paper we target the problem of planning the path of a set of UAVs carrying BSs, by taking into account: (i) the energy consumed by each UAV, and consequently the battery recharge and (ii) the deployment of many UAVs in the same area. We then employ a Q-learning based approach to solve the aforementioned problem in a realistic scenario. Our results demonstrate the effectiveness of the proposed approach.

The rest of the paper is organized as follows. Section 2 presents the considered scenario where clusters of nodes are identified. The Q-learning design is described in Sect. 3 where both the models and the approach are described. The relevant performance analysis is in Sect. 4 while Sect. 5 concludes the paper.

## 2 Considered Scenario

We consider a scenario (as shown in Fig. 1) in which several HetNet users are under the coverage of a fixed eNB. We then assume that the total area covered by the fixed eNB is divided into a set of non-overlapping clusters. Each user is then assigned to a cluster, based on its spatial location inside the area [12]. Each cluster is then characterized by a bit rate request, which is computed as the average bit rate of the users in the cluster. Without loss of generality, we also assume that the coverage of each UAV is overlapping the area of the cluster it is serving.

**Table 1.** UAV parameters

Parameters	Value
Max speed ( $v_{\max}$ )	8.3 m/s
Max acceleration ( $a_{\max}$ )	4 m/s <sup>2</sup>
Battery autonomy	30 min
Flight altitude	50 m
Weight	6 kg
eNB mounted radius (UAV footprint)	500 m

Clearly, the transmission undergoes a path loss, which depends on the distance between the user and the serving BS (either the fixed eNB or the UAV). Depending on the channel conditions, each user will be subject to a given channel quality, which is characterized by a specific Spectral Efficiency, typically expressed in terms of Channel Quality Indicator (CQI). Clearly, the channel quality has a large impact on the achievable throughput, and hence on several user application (like video streaming as in [13]).

In this scenario, a UAV supposedly flies at an altitude higher than the buildings height and covers a circular area on the ground. On the other hand, UAVs have to deal with a limited battery, which has to be mandatory recharged before running out of energy. A number  $N_{CS}$  of Charging Stations (CSs), equal to the number of UAVs, are placed on a given distance from the central eNB. The UAVs can access the CSs and autonomously charge their battery.

The flight path optimization is carried out offline in a centralized way. The offline approach, also adopted in [11], (i) allows to prioritize service on area where the expected reward is higher, (ii) relieves the UAVs of inter-UAV communication, and (iii) assures that UAVs do not overlap in serving the same areas. This is realized by deterministically preventing overlap during the learning stage, whereas in online distributed optimization this can be tackled by decentralized strategies, like the bio-inspired one presented in [9].

### 3 Q-Learning Design

#### 3.1 UAV Characterization

The UAV model used in the simulation analysis has the features shown in Table 1. We use those features to parameterize the simulation so that both the time needed to perform each action and the energy consumed are realistic. We assume that the UAVs are all similar, and that they move between clusters barycenters and CSs through a straight line. While a UAV is moving or charging or waiting at a CS, it does not serve any user (i.e., it does not allocated any bandwidth).

#### 3.2 Users Clustering

The central eNB is covering an area of radius  $R$ , this area can be divided in clusters of radius  $r$  that depends on the mounted eNB footprint. Among these

clusters, the central one benefits of the best CQI and does not need to be covered by a UAV. If we do consider all the remaining clusters, the size of the Q-learning problem (represented by a Q-matrix as discussed below) maybe very big and in that case the computations would be very long, we also do know that in real life, some clusters have a high or low bandwidth request depending on the geographical area, for example: green spaces, schools, houses, industrial buildings, warehouses etc.

Therefore, to optimize the computations, we can consider a number  $N_C$  of clusters  $\mathcal{C}$  identified by their position  $(x, y)$  in space and defined as follow:

$$\mathcal{C} = \{ c^{(i)} = (x^{(i)}, y^{(i)}), i = 1 \dots N_C \}$$

Each cluster is also characterized by a Spectral Efficiency value  $\mathcal{SE} [bps/Hz]$ , which is maximal near the eNB station (4 bps/Hz) in our case, and drops exponentially with the distance from it. The number of clusters is supposed to be larger than the number of UAVs  $N_{UAV}$ , where each UAV can cover one cluster at a time and a cluster cannot be covered by more than one UAV.

### 3.3 Energy Aware Q-Learning Algorithm

We describe here the energy aware learning algorithm, exploiting the widely known Q-learning approach formerly introduced in [8] and ever since applied in a huge variety of frameworks, particularly in [7] where it is applied for one UAV path planning. In a nutshell, the Q-learning problem space consists of an agent, a set  $\mathcal{S}$  of states which the agent can achieve, and a set of actions per state  $\mathcal{A}$ . The algorithm computes a reward for each state-action couple. At each iteration, referred to as one epoch, the algorithm explores a chain of consecutive states and updates the objective function  $Q$  stored in a matrix. Within the  $e$ -th epoch, each one composed by  $N_k$  steps corresponding to a fixed number of time-slots in our case, the computation explores a sequence of states as follows: from each state  $s_k \in \mathcal{S}$  the agent can choose an action  $a_k \in \mathcal{A}$  that will lead the agent to a next state  $s_{k+1} \in \mathcal{S}$ ,  $k$  being the index of the state within the  $e$ -th epoch state sequence. Executing an action  $a_k$  in a specific state  $s_k$  provides the agent with a reward. The learning algorithm maximizes its cumulative reward according to an  $\varepsilon$ -greedy policy. This means that at each state, with probability  $\varepsilon$  it chooses a random action and with probability  $1 - \varepsilon$  it selects the action that gives a maximum reward [7]. The value  $\varepsilon$  is initialized at 1 and is updated at each epoch to slowly decrease, this makes the algorithm try many random actions at the beginning and maximize the reward at the end of the training.

In the proposed energy and quality aware learning algorithm, the agent is one of the UAVs, and the states and actions are defined as follows:  $s = s(P, B, T)$  where:

- $P$  is the center position of a cluster or of a CS,  $P \in \{Cl_1 \dots Cl_{N_C}, CS_1 \dots CS_{N_{CS}}\}$
- $B$  is the battery level which is an integer varying from 1 to  $N_B$ ,  $B \in \{1 \dots 10\}$
- $T$  is the  $k$ -th training step, which is also the actual timeslot, varying from a value of 1 to  $N_k$ .

The actions  $a \in \{GoCl_1..GoCl_{N_C}, GoCS_1..GoCS_{N_{CS}}, Cover, Charge, Stay\}$ , are:

- go to a cluster or to a CS;
- remain at the actual cluster and cover;
- remain at the actual CS and charge;
- idle at the actual CS without charging and wait.

The *Stay* action is usually performed when the reached CS is already taken by another UAV. It is important to note that not all actions are accessible from all states.

The Q-matrix where the state-action rewards are stored is of size  $N_{States} \times N_{Actions} \times N_{UAV}$ , where:

$$N_{States} = (N_C + N_{CS}) \times N_B \times N_k \quad N_{Actions} = N_C + N_{CS} + 3$$

When at a state  $s$  an action  $a$  is performed, we obtain a state  $s' = (P', B', T')$

Where  $P'$  is the new position if the action performed was *Go* and remains unchanged for the other actions. And  $B'$  is the new battery level that decreases if the action was *Go* or *cover*, increases if the action was *charge* and remains unchanged if the action was *stay*. And  $T' = T + 1$ .

We initialize the Q-matrix by setting a  $-\infty$  reward for the forbidden actions at each state. The rules to define the forbidden actions are stated below:

- At a cluster's center, it is forbidden to charge or to stay (idle mode, not covering).
- In a CS, it is forbidden to cover.
- When the battery is full, it is forbidden to charge.
- From any location, it is forbidden to go to a cluster from which the battery level won't allow to reach a CS in the next time-slot.
- When the battery is low, it is forbidden to cover a cluster.

The Q-matrix is filled during the learning, at each step  $k$  of an epoch  $e$ , the function  $Q(s_k, a_k)$  is computed using the following formula:

$$Q^{(e)}(s_k, a_k) \leftarrow (1 - \alpha_k) \cdot Q^{(e-1)}(s_k, a_k) + \alpha_k \cdot [R_k + \gamma \max_{a_{k+1}} Q^{(e-1)}(s_{k+1}, a_{k+1})] \quad (1)$$

Where  $\alpha_k \in [0, 1]$  is the learning rate,  $\gamma \in [0, 1]$  is the discount factor that trades off the importance of earlier versus current reward [7]. The elementary reward  $R_k$  below denoted  $R_{TOT}^{(t)}$  has two components, one related to the bandwidth and one to the battery consumption. It is defined as the gain in bandwidth per time-slot subtracted by the battery consumption observed during the transition from state  $s_k$  to the new state  $s_{k+1}$  performing the action  $a_k$ :

$$R_{TOT}^{(t)} = \alpha \times R_{BW}^{(t)} + \beta \times R_E^{(t)} \quad (2)$$

$\alpha$  and  $\beta$  are coefficients used to give a weight to each component.

The bandwidth related component is calculated as follow:

$$R_{BW}^{(t)} = \sum_{i=1}^{N_C} \left[ \delta_i \times B_{UAV} \times \frac{\mathcal{SE}_{UAV}}{bitrate_i} + (1 - \delta_i) \times \frac{B_{BS}}{N_C - \sum_{i=1}^{N_C} \delta_i} \times \frac{\mathcal{SE}_{BSi}}{bitrate_i} \right] - \sum_{i=1}^{N_C} \left[ \frac{B_{BS}}{N_C} \times \frac{\mathcal{SE}_{BSi}}{bitrate_i} \right] \quad (3)$$

where  $\delta_i = 1$  if the UAV is covering cluster  $i$ , 0 otherwise,  $B_{UAV}$  and  $B_{BS}$  are the bandwidth (expressed in Hz) available for the UAV and the base station respectively equal to 5 and 20 MHz. The  $bitrate_i$  is the data rate request of cluster  $i$  at time-slot  $t$ . The  $\mathcal{SE}_{UAV}$  is the spectral efficiency with respect to the UAV (assumed as 4 bps/Hz), while  $\mathcal{SE}_{BSi}$  is the spectral efficiency with respect to the base station which is the highest at the center of the area and decreases exponentially with the distance, namely it is 2.3222 for clusters 2–7, 1.0333 for clusters 8–13 and 0.6139 for clusters 14–19.

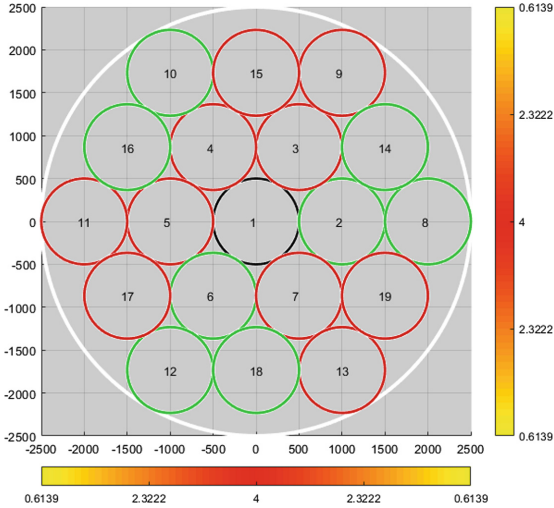
Given the total bandwidth resources  $B_{UAV}$  and  $B_{BS}$ , the reward term  $R_{BW}^{(t)}$  accounts for the excess data rate offered by the network (with or without UAV) with respect to the average data rate requested by each users' clusters.

The battery related component is based on the model established in [14], and depends on the action  $a$  as below:

$$R_E^{(t)} = \begin{cases} -(E_L + E_V + E_D) & \text{if } a = Go \\ -(E_L + E_{BS}) & \text{if } a = Cover \\ 1000 & \text{if } a = Charge \\ 0 & \text{if } a = Stay \end{cases} \quad (4)$$

Where  $E_L, E_V, E_D$  and  $E_{BS}$  are respectively the level flight energy, vertical flight energy, blade drag profile energy and the users serving energy computed as in [14], with the following parameters: weight of the UAV plus the BS, gravitational acceleration, air density, area of the UAV's rotor disk, profile drag coefficient and the BS power consumption.

In order to perform multi-UAVs path planning, we first train a single UAV for a large number of epochs, filling a part of the Q-matrix for all the possible (State, Action) combinations. The remaining parts of the Q-matrix, relative to the other UAVs are then initialised with the obtained values, as the reward for a (state, action) combination does not depend on the specific UAV performing it: hence  $Q(s, a, d) \leftarrow Q(s, a, 1)$ . The optimal path for the first UAV is given by taking the action with the maximum reward at each state. These actions are set as forbidden for all the following UAVs, to avoid having two UAVs colliding in the center of a cluster. The following UAVs are trained one by one for a smaller number of epochs, always setting the optimal path undertaken by a UAV as forbidden for the following ones.



**Fig. 2.** Active clusters and spectral efficiency in space

**Table 2.** Scenario’s parameters

Parameters	Value
eNB radius $R$	2.5 [km]
Cluster radius	0.5 [km]
Total amount of clusters	19
Active clusters	8
Number of UAVs and CSs	4
CSs distance from eNB	1.5 [km]
Total time considered	24 h
Time-slot duration	10 min

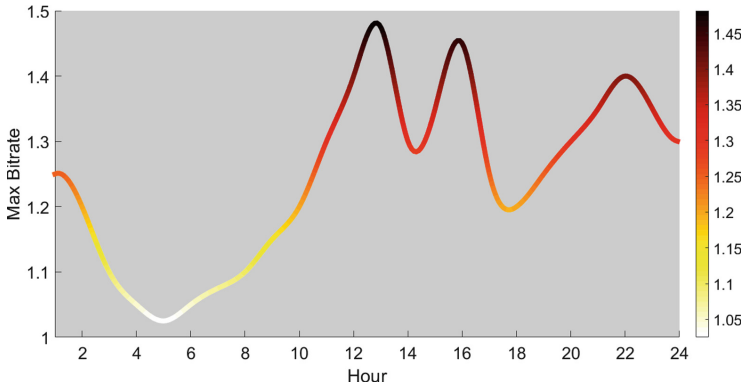
## 4 Performance Analysis

To evaluate the performance of the proposed approach we implemented a custom simulator in Matlab and evaluate the algorithm in a scenario where 4 UAVs move in an area around an eNB radius of 2.5 km. To this aim we set the parameters as in Table 2.

The 8 active clusters are selected randomly between the total 19 clusters, the maximum spectral efficiency is 4. The CSs are placed at a 1.5 km distance around the center, on the axis X and Y. The resulting simulation scenario is represented in Fig. 2.

To simulate the UAV’s behaviour, we make the following assumptions:

- the battery levels are integer values between 1 and 10;



**Fig. 3.** Data rate over time

- one battery unit per time-slot is consumed when performing covering;
- taking into account the total flight time, we calculate the battery consumption for the movement (that may be 1, 2 or 3 units depending on the travelled distance);
- one battery unit is gained per time-slot when charging;
- one time-slot is enough for a UAV to reach any destination<sup>1</sup>.

We consider 24h long epochs, divided in 288 time-slots of 5 min. The requested data rate of the 8 considered clusters changes value at every time-slot, but is repeated every 24h. For the data rate request we generate random values (between 1 and 1.5 Mbps) for 4 clusters and simulate a realistic request for the remaining 4. To achieve that, an entire day of max bit rate request is assumed, computing an interpolation of 24 points, setting a certain value of the bit rate for each hour. It has been assumed a high request during office hours (9–12, 14–16), a medium request during the afternoon and a low request during sleeping hours. From the obtained curve, showed in Fig. 3, we map the values of the data rate for each time-slot.

With these parameters, our Q-matrix is of size  $12 \times 10 \times 288 \times 15 \times 4 = 2,073,600$ . To let our agents learn, or to train them, we run the previously described Q-learning algorithm for 30000 epochs for the first UAV and 2000 epochs for the 3 others, after some trials, we fine-tune the elementary reward coefficients  $\alpha$  at 0.995 and  $\beta$  at 0.005. The learning rate  $\alpha_k$  is set at  $\frac{1}{1+Number_{NodeVisits}}$ , with  $Number_{NodeVisits}$  the times the cell corresponding to that particular (*state, action*) couple has been visited and updated during the training. Our agents have as an objective to maximize their respective rewards by improving the QoE of the users, to manage their batteries, and to avoid service overlapping of the clusters. The results show that the agents do improve the QoE, and satisfy the service overlapping conditions, all while managing their battery.

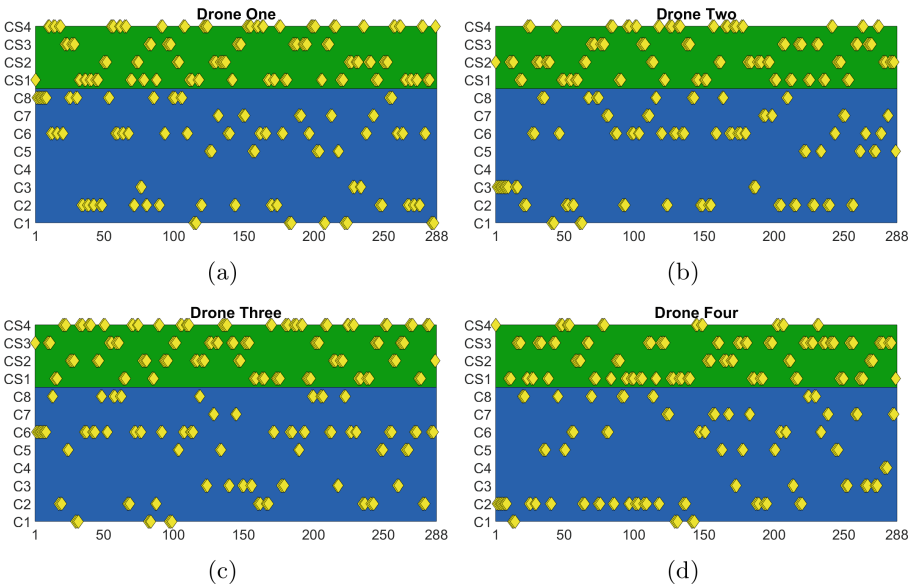
<sup>1</sup> With one time-slot (5 min) we can reach any destination, but with different consumption of battery levels depending on the travelled distance.



**Table 3.** Percentage of cover versus charge

UAV	Time-slots covering	Time-slots charging
1	27%	33%
2	30%	35%
3	28%	35%
4	28%	36%

Each UAV’s path is represented in a different plot in Fig.4(a), (b), (c) and (d) where we can observe that the UAVs move between the clusters and the CSs and also that a UAV stays at a certain cluster for several time-slots to cover.



**Fig. 4.** Positions of the 4 UAVs over time

The percentage of time-slots spent covering and charging with respect to the total time-slots for each UAV are presented in Table 3. Knowing that the energy consumed while covering for 1 time-slot is 1 battery level, the same as the energy gained when charging for 1 time-slot, it is expected that the percentage of time-slots charging is a bit greater than the percentage of time-slots covering, as the energy obtained while charging is spent while moving and covering. If we had used a larger battery level representation, in rounded up percentage for example, and applied the same energy model as used to calculate the reward, the energy gained during 1 time-slot charging would allow to cover for 9 time-slots, but this means multiplying the Q-matrix size by 10.

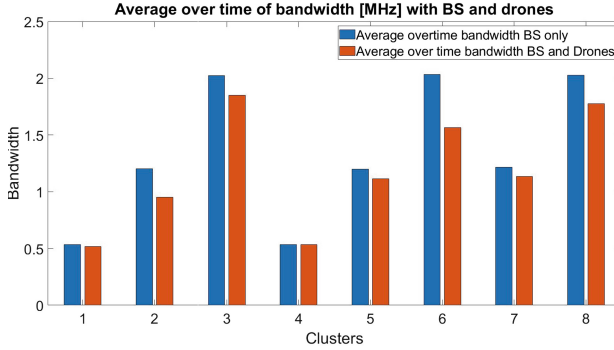


Fig. 5. Average improvements of bandwidths

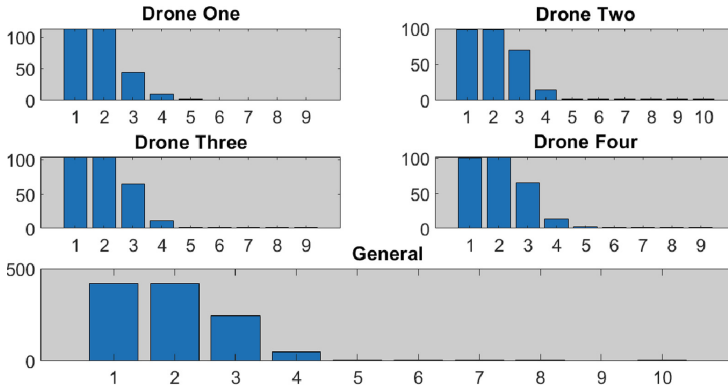


Fig. 6. Battery levels

Concretely the improvement of the bandwidth (MHz) is shown in Fig. 5 which shows the average data rate request (Mbps) divided by the spectral efficiency (bps/Hz) for all the clusters with and without the use of the UAVs. The vertical axis represents the average bandwidth and the horizontal axis represents the different clusters. The blue bars represent the value of the bandwidth without the use of the UAVs, the orange bars represent the bandwidth with the use of the UAVs. Overall, the allocated bandwidth is improved for all the clusters, more specifically the clusters with the highest original bandwidth request witness a consequent improvement. Cluster 4 in particular, which is the closest to the BS and has the highest  $\mathcal{SE}$  has no improvements with the UAVs.

The management of the battery can be evaluated by observing Fig. 6 which represents the number of times a battery level was reached. Normally the battery should be used almost fully before being recharged fully, and all the levels should be reached an almost equal number of times. In the case the agent learns to optimize the charging/covering actions, it may manage the battery differently, this adaptation makes the battery levels unequally distributed. In our case, the

most frequent levels are 1, 2 and 3, this could be improved by increasing the reward coefficient  $\beta$  for the action *charge* or by assigning a higher reward for full charging.

The total reward per epoch increases during the learning for all the UAVs. Figure 7 represents the reward over epochs for the four UAVs separately. UAV 1 was trained for 30,000 epochs and started with an empty Q-matrix, the reward starts at  $-16$  and reaches 5000, we notice that it is still increasing and requires a longer training to converge. The rewards over epochs for UAVs 2,3 and 4, which started with the Q-matrix learned by UAV 1 and trained for 2000 epochs, their initial reward is 2000 and reaches 6000, also here the reward is increasing but not converging yet.

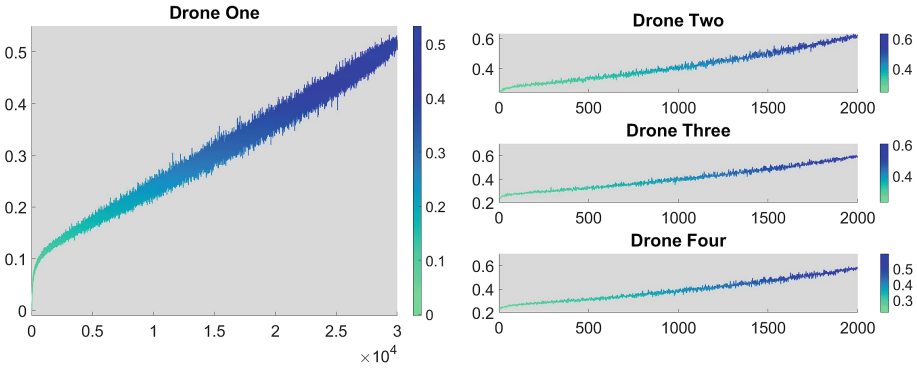


Fig. 7. Reward of UAVs over epochs

## 5 Conclusion and Future Work

We have presented an energy and quality aware flight planning strategy based on a Q-learning approach. The proposed solution is able to tackle a scenario where multi-UAVs are deployed. Moreover, we explicitly take into account the limited UAV battery. More in detail, we have defined separate states, actions and rewards for each UAV. In addition, we have imposed the service overlapping avoidance by setting a priority order. This has reduced the size of both the states and the actions domains, thus leading to satisfying results. Several training steps with different parameters were performed before reaching these results. However, we point out that there is always room for improvement, by e.g., increasing the number of epochs, fine-tuning the reward function parameters, or through a different state representation with a larger domain for the battery level.

**Acknowledgement.** This work has received funding from the University of Rome Tor Vergata BRIGHT project (Mission Sustainability Call).

## References

1. Sackl, A., Casas, P., Schatz, R., Janowski, L., Irmer, R.: Quantifying the impact of network bandwidth fluctuations and outages on web QoE. In: Seventh International Workshop on Quality of Multimedia Experience (QoMEX), Pylos-Nestoras, pp. 1–6 (2015)
2. Lyu, J., Zeng, Y., Zhang, R.: UAV-aided offloading for cellular hotspot. *IEEE Trans. Wireless Commun.* **17**(6), 3988–4001 (2018)
3. Zeng, Y., Lyu, J., Zhang, R.: Cellular-connected UAV: potential, challenges and promising technologies. *IEEE Wireless Commun.* **26**(1), 120–127 (2019)
4. Mozaffari, M., Saad, W., Bennis, M., Nam, Y.-H., Debbah, M.: A tutorial on UAVs for wireless networks: applications, challenges, and open problems. *IEEE Commun. Surv. Tutorials* (2019, in press)
5. Zeng, Y., Zhang, R., Lim, T.J.: Wireless communications with unmanned aerial vehicles: opportunities and challenges. *IEEE Commun. Mag.* **54**(5), 36–42 (2016)
6. Zhang, Q., Mozaffari, M., Saad, W., Bennis, M., Debbah, M.: Machine learning for predictive on-demand deployment of UAVs for wireless communications. In: *IEEE Global Communications Conference (GLOBECOM)*, Abu Dhabi, United Arab Emirates, pp. 1–6 (2018)
7. Colonnese, S., Carlesimo, A., Brigato, L., Cuomo, F.: QoE-aware UAV flight path design for mobile video streaming in HetNet. In: *2018 IEEE 10th Sensor Array and Multichannel Signal Processing Workshop (SAM)* (2018)
8. Watkins, C.: Learning from delayed rewards. Ph.D. thesis, May 1989
9. Trotta, A., Di Felice, M., Montori, F., Chowdhury, K.R., Bononi, L.: Joint coverage, connectivity, and charging strategies for distributed UAV networks. *IEEE Trans. Rob.* **34**(4), 883–900 (2018)
10. Song, B.D., Kim, J., Kim, J., Park, H., Morrison, J.R., Shim, D.H.: Persistent UAV service: an improved scheduling formulation and prototypes of system components. *J. Intell. Robot. Syst.* **74**(1), 221–232 (2014)
11. Scherer, J., Rinner, B.: Persistent multiUAV surveillance with energy and communication constraints. In: *Proceedings IEEE International Conference on Automation Science and Engineering*, Fort Worth, TX, USA, pp. 1225–1230 (2016)
12. Afshang, M., Dhillon, H.S.: Poisson cluster process based analysis of HetNets with correlated user and base station locations. *IEEE Trans. Wireless Commun.* **17**(4), 2417–2431 (2018)
13. Colonnese, S., Cuomo, F., Chiaraviglio, L., Salvatore, V., Melodia, T., Rubin, I.: Clever: a cooperative and cross-layer approach to video streaming in HetNets. *IEEE Trans. Mob. Comput.* **17**(7), 1497–1510 (2018)
14. Chiaraviglio, L., Amorosi, L., Malandrino, F., Chiasserini, C.F., Dell’Olmo, P., Casetti, C.: Optimal throughput management in UAV-based networks during disasters. In: *1st Mission-Oriented Wireless Sensor, UAV and Robot Networking Workshop (INFOCOM 2019 WKSHPs - MiSARN 2019)*, April 2019