Energy and Service-priority aware Trajectory Design for UAV-BSs using Double Q-Learning

Sayed Amir Hoseini[1], Ayub Bokani[2], Jahan Hassan[3], Shavbo Salehi[4] Salil S. Kanhere[5]
[1,2,3] School of Engineering and Technology, The Central Queensland University, Sydney, Australia
{s.hoseini, a.bokani, j.hassan}@cqu.edu.au
[4] Electrical and Computer Engineering Department, Urmia University, Urmia, Iran
shavbo.salehi@urmia.ac.ir
[5] School of Computer Science and Engineering, The University of New South Wales, Sydney, Australia
salil.kanhere@unsw.edu.au

***Abstract*—Next generation mobile networks have proposed the integration of Unmanned Aerial Vehicles (UAVs) as aerial base stations (UAV-BS) to serve ground nodes. Despite having advantages of using UAV-BSs, their dependence on the on-board, limited-capacity battery hinders their service continuity. Shorter trajectories can save flying energy, however UAV-BSs must also serve nodes based on their service priority since nodes' service requirements are not always the same. In this paper, we present an energy-efficient trajectory optimization for a UAV assisted IoT system in which the UAV-BS considers the IoT nodes' service priorities in making its movement decisions. We solve the trajectory optimization problem using Double Q-Learning algorithm. Simulation results reveal that the Q-Learning based optimized trajectory outperforms a benchmark algorithm, namely Greedily-served algorithm, in terms of reducing the average energy consumption of the UAV-BS as well as the service delay for high priority nodes.**

I. Introduction

Next generation wireless communication systems, e.g., 5G and beyond, have proposed the integration of Unmanned Aerial Vehicles (UAVs) as aerial base stations (UAV-BS) to provide terrestrial communication services [1]. Due to their dynamic deployability and mobility, UAVs can act as flying base stations to collect sensor data by moving closer to the Internet of Things (IoT) sensor nodes in remote farming or disaster areas without cellular or Wi-Fi coverage. An implementation of such a UAV-based sensor data collection system in the rural area of China for agricultural monitoring has been reported in [2]. However, unlike the terrestrial base stations that have continuous power supply, the UAV-BSs, or UAVs in general, have limited power supply from their on-board battery. For example, off-the-shelf drones such as DJI Spreading Wings S900 can stay afloat for around 18 minutes when fully charged [3]. Once the power is drained, the service provided by the UAV needs to pause for battery replacement or recharging. This leads to service-time limitations and discontinuity of UAV-delivered services.

Although some researchers are looking into aerial recharging of UAVs to address this [4], [5], the power limitation issue certainly motivates the design of methods and mechanisms for UAVs to function in an energy-efficient manner allowing the UAVs to do more within their battery budget, irrespective of the charging mechanisms. Since the major energy consumption

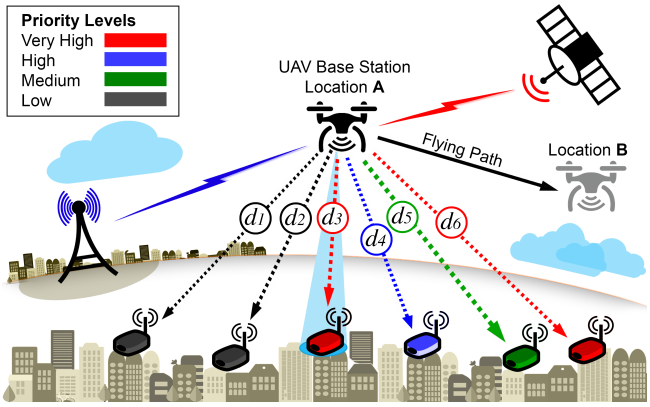


Fig. 1: **UAV assisted IoT System** where the UAV serves nodes, e.g., IoT devices or sensors, with different priority levels (shown with color-coding) and distances (shown with labelled arrows). The nodes should be in communication range, therefore to achieve this the UAV will fly above the nodes at a certain distance to collect data.

of an UAV comes from its mechanical actions, e.g., flying, shortening the flight length in its mission is an effective way to reduce energy consumption, which inevitably leads to a trajectory optimization problem. While conserving energy is highly important, in a UAV-based IoT sensor data collection scenario, the UAV must also serve ground sensors as per their priority or urgency levels, e.g., based on their delay tolerance [6] levels. Therefore, while optimizing the trajectory for energy-savings, the nodes' priorities must be considered by the UAVs. Related works in trajectory design address these two requirements independently; as examples, see [7] for energy-efficient trajectory optimization, and [6], [8] for IoT nodes' priority-based trajectory optimization. Our aim is to combine both requirements in the trajectory design.

In this paper, therefore, we design a UAV-BS's trajectory in a UAV assisted node priority oriented IoT system that minimizes flying costs while serving nodes as per their priorities. Therefore, an intelligent model that the UAV-BS can use to make the best node-visiting decisions at different states is required. The problem can be formulated as that of

selecting an action from a finite set of choices based on the repetitive observations of the environment. We optimize the UAV path using Double Q-Learning [9] which is a model-free reinforcement learning algorithm. Q-Learning not only learns in which order the nodes should be served by the UAV-BS after some experiences, but also can dynamically update the decision policy if the environment or nodes behaviour changes. Our contributions in this paper are summarized below:

- We demonstrate that the UAV-BS trajectory can be optimized based on energy-efficiency and service-priority using Double Q-Learning in a UAV-based sensor data collection scenario, and evaluate the performance of the proposed system using simulations.
- Our simulation results confirm that the optimized trajectory not only achieves significant energy savings compared to the bench-marking algorithm, but also serves nodes as per their priorities thereby enhancing the practicality of such systems.

The rest of the paper is organized as follows: Section II describes related work on both energy-efficient trajectory design and priority-based services in UAV-assisted networks. In Section III, we present our system model and Double Q-Learning formulation of the optimization problem. Section IV provides details of our simulation studies and a discussion of results. Finally, we conclude our paper in Section V.

II. Related Works

The electronic energy consumption in UAVs is negligible compared to the total energy consumption [10]. As such, related work predominantly focuses on the reduction of mechanical energy consumption in various ways, such as by controlling flight radius, speed, and height [3]. For UAV-BSs, another way to reduce the mechanical energy consumption is by optimizing the path planning or trajectory design as proposed in [7], [11], [12], etc. Zeng et al. [11] presented an energy-efficient trajectory optimization for UAVs which also considered the communication throughput as a performance metric. Authors of [7] addressed the total energy minimization of wireless communication with rotary-wing UAV by optimizing the propulsion energy as well as the communication energy in the trajectory design. Work by authors in [13] focused on the trajectory optimization of a rotary-wing UAV, acting as a relay node in a cellular network setting, whereby they aim to achieve a trade-off between long term communication delay and power consumption for UAV's mobility.

Authors in [8] presented a data acquisition framework for sensor networks using an UAV to collect sensor data, whereby authors introduce sensor-node priorities in the frame selection at the MAC layer. The sensor nodes get assigned different transmission priorities in different frames, based on their location with respect to the UAV, resulting in throughput maximization. Wang et al. [6] presented a trajectory design for UAV-based, time-sensitive IoT network where the nodes are assigned various priorities based on their delay tolerance sensitivity. Authors used these priorities in the cost function of a Deep Q-Learning based optimization of the trajectory to minimize system cost which they defined in terms of latency performance of the heterogeneous network.

IoT nodes' priority level values can also be set *using nodes' residual energy levels* [14] which determines active/sleep schedule of sensor nodes. This information can be used by the UAV to determine node serving priority as nodes with lower residual energy are preferred to be served first.

In our prior work [15], UAV-BS trajectory design was optimized using the Travelling Salesman Problem (TSP) method for energy efficiency but the applicability of TSP is limited when the service priory has to be considered as well. Therefore, we employ Double Q-Learning in this paper to optimize the trajectory considering both energy efficiency and service priority.

III. System Model and Q-Learning Formulation

In our considered scenarios, we assume static ground nodes that are randomly positioned in different locations and require data gathering services from the the UAV with different priority levels as shown in Figure 1. Priority values change as per nodes' residual energy level. Nodes' location and initial service priority are pre-loaded in UAV software. Each time the UAV collects data from a node, it learns the updated node's residual energy level and once all nodes are served, the new values are used by the UAV to determine the priorities in the next round. Since, the UAV should take action based on the observed environment and each experience can contribute the decision enhancement, we find this optimisation is fitted to Reinforcement Learning.

We model the service area on the ground as a grid and consider the UAV as the Q-Learning agent. The state space is created by the UAV's observations of its own location, nodes' locations, and service priority level of each node. The Q-Learning agent must take action and fly to the next node which should be served. Therefore, the number of possible actions equals to the number of unserved nodes. The Q-Learning model consists of following four basic components:

- **Agent** (UAV) is the flying base station which should serve nodes one by one based on their locations and service priority levels.
- **Action** (a) is the UAV's next flying destination which is determined by the location of next node to serve.
- **State** (S) is defined based on the observed information of UAV's current location and nodes information. Therefore, our system state is defined as $S = \{L_{uav}, L_{nd}, \Omega_{nd}\}$ where L_{uav} is UAV's location, $L_{nd} = [L_{nd_1}, L_{nd_2}, ..., L_{nd_n}]$ is a vector that represent the location of node 1 to n and $\Omega_{nd} = [\Omega_{nd_1}, \Omega_{nd_2}, ..., \Omega_{nd_n}]$ is a vector that denotes nodes' service priority (sp) and status (i.e., served or not). We consider a number from 1 to 4 to represent low, medium, high and very high sp levels respectively. Once a node is served, we set its sp level to zero to distinguish it from the other nodes waiting to be served.
- **Revenue** (R) is a function that returns a real number for each $state - action$ pair after the Q-Learning agent

has moved to the next state s' by taking action a. Our revenue function is a linear combination of $rewards$ and $penalties$. Serving a node results in a reward, whereas penalties are given for flying energy cost and service delay.

In this model, we consider variable time steps due to having different distances between the UAV and each ground node. At each time step, the UAV selects the next waiting node and flies towards it. The service time is assumed to be negligible and the UAV is assumed to fly at a fixed speed and quickly communicates with nodes. To save ground nodes' energy, the UAV communicates data in the closest distance with these devices which means the UAV collects data when it is on top of them. In the Q-Learning the rewards of each state-action are saved in Q-Table and are being updated by new experiment. While traditional Q-Learning uses one Q-Table, the double Q-Learning employs two Q-Tables to avoid possible local optima and achieves the global optimum. Hence, we denote these Q-Tables as Q_A-Table and Q_B-Table. After serving each node, the Q-values of the Q-table that was used in serving the node are updated using the relevant double Q-Learning equation from the below equations:

$$Q_A^{new}(s,a) = (1-\alpha)Q_A(s,a) + \alpha(R(s,a) + \gamma Q_B(s', a^*)), \tag{1}$$

$$Q_B^{new}(s,a) = (1-\alpha)Q_B(s,a) + \alpha(R(s,a) + \gamma Q_A(s', b^*)), \tag{2}$$

where α and γ are the learning rate and discount factor respectively. R is the revenue function, s′ is the next state after taking action a at state s and a^* and b^* are the maximum Q-value of all state-action pairs on state s′ as:

$$a^* = \arg\max_a Q_A(s', a), \tag{3}$$

$$b^* = \arg\max_a Q_B(s', a). \tag{4}$$

For the Q-Learning revenue function $R(s,a)$, we consider a *reward* when the UAV delivers services with a high priority and we apply different $penalties$ for the service delivery delays and flying energy consumption. Such considered rewards and penalties help the Q-Learning agent, i.e., the UAV, to learn the model and find an optimal trajectory that minimizes the total energy consumption, serves the least delay tolerant nodes first, and increases the overall QoE. Hence, The revenue function R is calculated as:

$$R = w_1 \Omega_{nd_a} - w_2 \sum_{i=1, i\neq a}^{n} \Omega_{nd_i} t_s - w_3 \int_0^{t_s} P(V) dt, \tag{5}$$

where w_1, w_2 and w_3 are the tuning parameters, nd_a is the served node, and t_s is elapsed time from serving the last node. $P(V)$ is UAV's power consumption for flying with speed V which is calculated as [7]:

$$\begin{aligned} P(V) =& P_0 \left(1 + \frac{3V^2}{U^2}\right) + P_i \left(\sqrt{1 + \frac{V^4}{4\nu_0^4}} - \frac{V^2}{2\nu_0^2}\right)^{1/2} \\ &+ \frac{1}{2} d_0 \rho s A V^3, \end{aligned} \tag{6}$$

Algorithm 1 : Double Q-Learning Episode

```
Load Q_A-Table and Q_B-Table from previous episode
Initialize UAV location
exploration = epsilon
Observe nodes' location
Set Q-Selector = A
while there is a node to serve
do {
  Observe nodes' service priority
  Current state = (UAV location, nodes' location, nodes' priority)
  Generate a random number p ∈[0,1]
  if p is less than Exploration
     Select next node randomly
  else
     if Q-Selector = A
        Choose next node based on Q_A-Values
     else if Q-Selector = B
        Choose next node based on Q_B-Values
  Move UAV to next node location
  Current node = Next node
  Collect current node data
  Calculate Revenue(consumed energy by UAV,
   current node service priority, service delay)
  if Q-Selector = A
     Update Q_A-Table using (1)
  else if Q-Selector = B
     Update Q_B-Table using (2)
  Set current node service priority to zero
  Toggle Q-Selector between A and B
  }
Decrease epsilon for the next episode
```

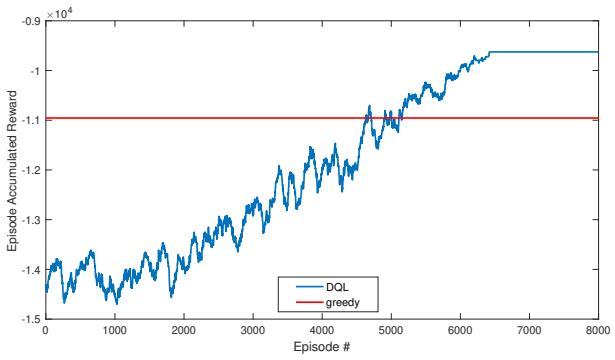
where relevant parameters are listed in Table I.

We use *epsilon-greedy* scheme [16] in our algorithm where actions are taken randomly at the beginning of learning process and the agent is fully in *exploration* mode. Besides, by decreasing the epsilon value, the chance of *exploitation* increases and the action with the highest Q-Value may be taken at each step. This is adjusted to rely on Double Q-Learning policy gradually over the time.

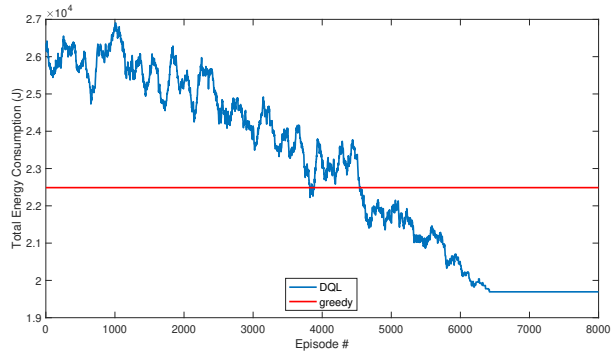
At each step of time, UAV observes the state s, then takes an action a and receives a revenue $R(s,a)$ after moving to the state s'. The goal of the training phase is to find the sequential order of served nodes that maximizes the total future revenues. Our revenue function will result in finding a flying path that minimizes energy consumption and improves the QoE. Each episode of Double Q-Learning is presented in Algorithm 1. The $Q_A(s,a)$ and $Q_B(s,a)$ values are saved and updated in Q_A-Table and Q_B-Table. The algorithm forces the UAV to

Algorithm 2 : Greedy

```
Initialize UAV location
Observe nodes' location
while there is a node to serve
do {
  remaining nodes = nodes with non-zero service priority
  Calculate the distance between UAV and remaining nodes
  Select the node with minimum distance as next node
  Move UAV to next node location
  current node = next node
  Collect current node data
  Set current node service priority to zero
  Decrease Exploration
  }
```



(a) Accumulated reward



(b) Total UAV energy consumption on the episode

Fig. 2: Double Q-Learning performance vs. Greedy (Nearest Neighbor) for scenario #1.

choose between *exploration* or *exploitation* approach for each decision. In exploration, UAV selects the next node to serve randomly and in exploitation, UAV takes the action with the highest Q-value for the observed state in one of the Q-Tables in turn. The exploration rate is adjusted by ϵ which is set to 1 in the early episodes to keep actions fully random and boost the training. Then through the episodes, it is decreased granularly to zero or a small value when the Double Q-Learning policy is reliable enough and most of the actions are taken based on Q-Values.

To compare Double Q-Learning performance, we consider a Greedy algorithm as a baseline. The Greedy (Nearest Neighbor) is presented in Algorithm 2 in which the UAV selects the nearest node to serve at each step, whereas the Double Q-Learning tries to achieve a balance between distance and node priority.

IV. Performance Evaluation

A. Simulation Setup

To evaluate the proposed method, we simulated two scenarios as illustrated in Figures 3 and 4 respectively, where six nodes are randomly distributed on a 6 by 6 grid service area. Each node has data for transmission with a random service priority which is represented by different colours as was illustrated in Figure 1. As mentioned above, ϵ-greedy scheme is used to force Double Q-Learning to take random actions at the start which is helpful in training to avoid local convergence. ϵ is equal to 1 for the first 1000 episodes, then it is reduced slowly to zero at episode # 6400. Therefore, the Double Q-Learning policy is fully operational for the last 1600 episodes. Simulation parameters are shown in Table I.

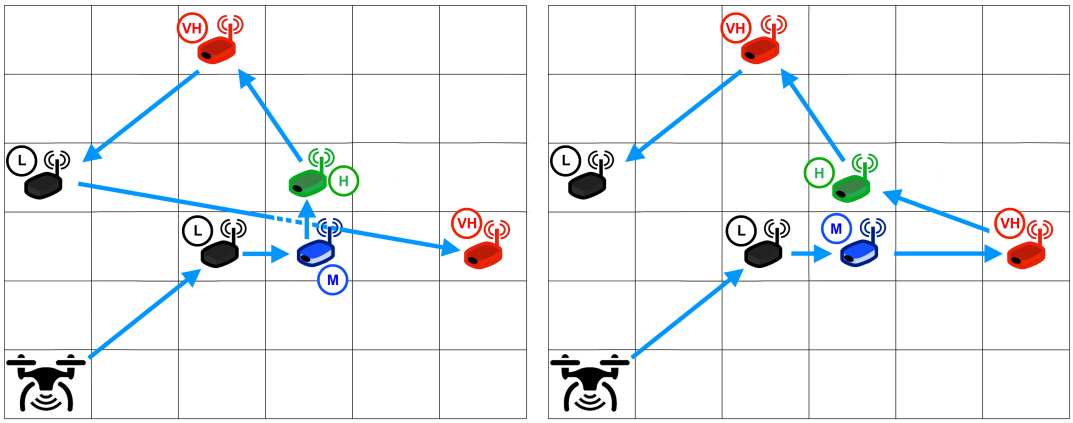
TABLE I: Simulation Parameters

Cell side	50 m
Learning rate (α)	0.5
Discount factor (γ)	0.95
Revenue tuning parameter (w_1)	30
Revenue tuning parameter (w_2)	7.5 s^{-1}
Revenue tuning parameter (w_3)	0.1 Joule^{-1}
Air density (ρ)	1.225 $\mathrm{kg/m}^3$
Rotor disc area (A)	0.181 m^2
Tip speed of the rotor blade (U)	96 m/s
Fuselage drag ratio (d_0)	0.9
Rotor solidity (s)	0.1
UAV speed (V)	5 m/s
Blade profile power (P_0)	29.4 W
induced power (P_i)	206.5 W
Mean rotor induced velocity (ν_0)	7.5 m/s

B. Results

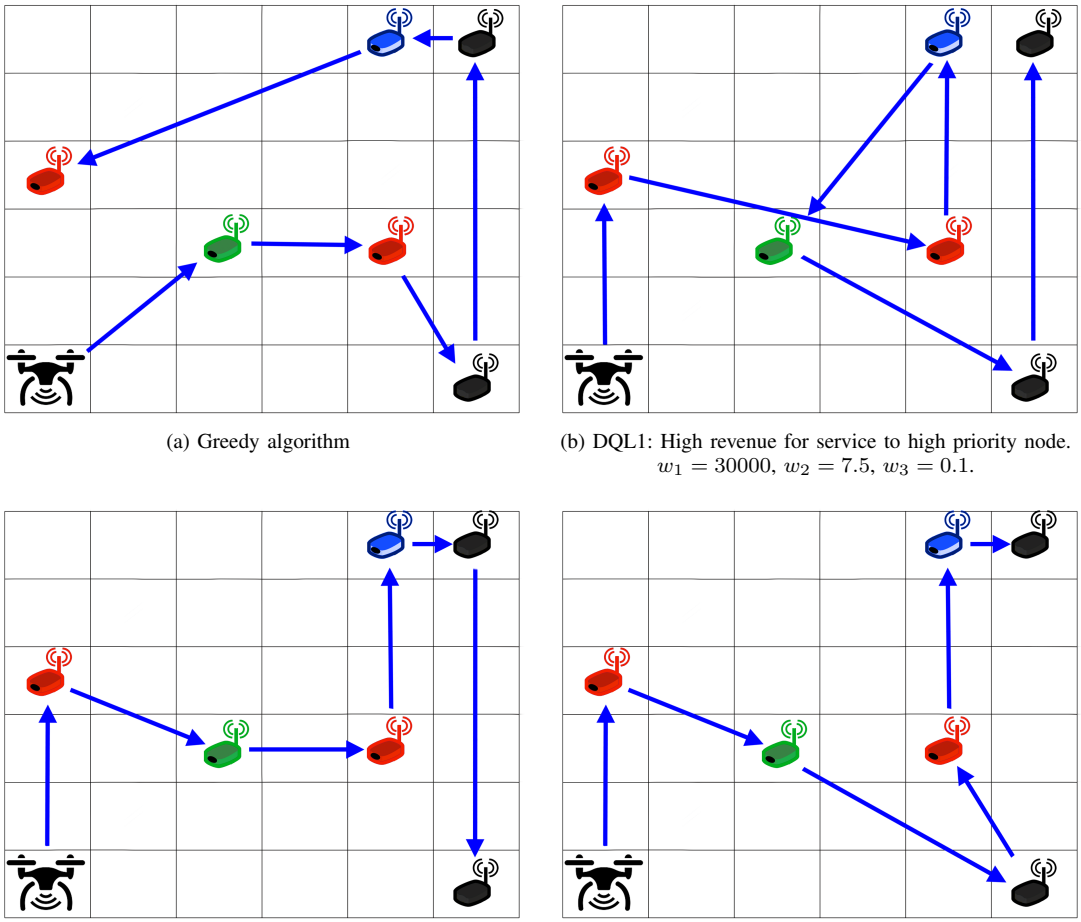
The first scenario is simulated using default tuning parameters of revenue function for Double Q-Learning and Greedy algorithm. The learning process of Double Q-Learning is presented in Figure 2 along with the Greedy algorithm result as baseline comparison. We plugged in the revenue function to the Greedy algorithm to compare accumulated reward for both algorithms. After around 5000 episodes, Double Q-Learning outperforms Greedy algorithm for both revenue and energy consumption. Moreover, the Double Q-Learning algorithm enhances QoE by serving high priority nodes first, whereas the Greedy algorithm ignores the priority values in making movement decisions. This can be observed in Figure 5 which we discuss later. Figure 3 shows the resulting trajectory for the same scenario for both of the algorithms, using the same priority color coding for nodes as was shown in Figure 1. The UAV starts flying from the bottom left corner and collects nodes' data one by one. Note that a near-optimal trajectory policy can be obtained offline by Double Q-Learning and utilized from the beginning of flight missions and gets updated by experiencing real-world operations continuously.

The revenue function in (5) has an essential role in Double Q-Learning behavior. The trajectory optimisation of UAV in our model depends to service priority of nodes, service delay and energy consumption of UAV. Therefore, there is a trade-off which can be adjusted by the revenue function. To explore this, we simulated the second scenario for three different tuning parameter set of the revenue function. In each set, we choose a very large value for one of w_1, w_2 and w_3. We also simulate the Greedy algorithm for the same scenario for comparison, with results presented in Figure 4 and Figure 5. The trajectory is plotted and the average delay is calculated after Double Q-Learning converges to a stable outcome for the fixed scenario. While the Greedy algorithm selects node service order based on the nearest neighbor approach, Double Q-Learning results in a different trajectory depending the parameter adjustments. When w_1 is set to very high value as



(a) Greedy Algorithm (b) Double Q-Learning

Fig. 3: UAV base station trajectory for scenario #1.



(a) Greedy algorithm

(b) DQL1: High revenue for service to high priority node. $w_1 = 30000$, $w_2 = 7.5$, $w_3 = 0.1$.

(c) DQL2: High revenue for QoE.$w_1 = 30$, $w_2 = 750$, $w_3 = 0.1$

(d) DQL3: High penalty for energy consumption. $w_1 = 30$, $w_2 = 7.5$, $w_3 = 100$.

Fig. 4: Greedy and Double Q-Learning Trajectories for scenario #2. Double Q-Learning results are generated for different revenue adjustment parameters.

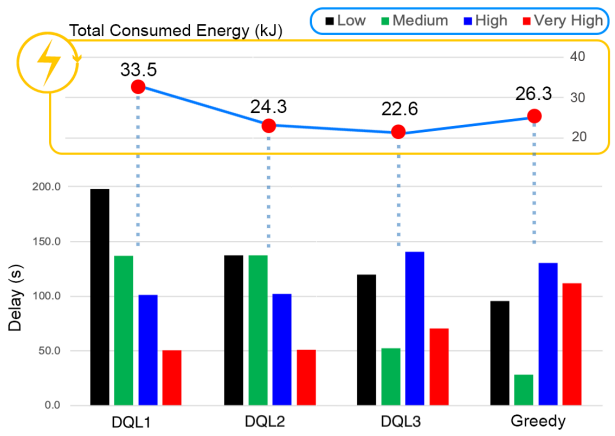


Fig. 5: Average Serving Delay and Energy Consumption Comparison of various revenue parameter set of Double Q-Learning and Greedy algorithm for scenario #2.

in Figure 4b (DQL1), UAV collects data of high priority nodes first. This is changed to a balanced behavior when a very high w_2 is set as in Figure 4c (DQL2). As shown in Figure 5, the balanced Double Q-Learning presents an overall better QoE and energy consumption. While serving high priority nodes results in 33.5 kJ of energy consumption, this is reduced to 24.3 kJ for the balanced mode. Further, the delay for serving low and moderate priority nodes are also reduced for the balanced Double Q-Learning in comparison to serving high priority nodes settings. Finally, we set w_3 to a very high value to minimize energy consumption. The result is presented in Figure 4d (DQL3). As can be seen in Figure 5, UAV is forced to travel the near shortest path and consumes only 22.6 kJ of energy which comes at a cost of high experienced delay for the higher priority nodes.

V. Conclusion

In this paper, we studied the scenario of a UAV providing data collection services to nodes with various service priorities in a UAV assisted IoT system. We optimized the UAV's trajectory using Double Q-Learning, with a view to reduce energy consumption while serving requesting nodes as per their required service priority. Simulation results revealed that the Q-Learning based trajectory outperformed the benchmark node-serving algorithm in reducing the average energy consumption of the UAV as well as enhancing QoE in regards to the service delay for high priority nodes. Through adjusting the tuning parameter set of the revenue function, we explored various behaviour of the Q-Learning model. We found that for a balanced setting of the parameters, the Q-Learning based trajectory was able to achieve the best trade-offs in terms of energy consumption and QoE for serving the highest priority nodes while also improving the QoE for other priority-category nodes. We note that Double Q-Learning has performance limitations when applied to an extended UAV observation space, e.g., when the number of nodes increase. For a more scalable model, in our future work we aim to use Deep Reinforcement Learning (DRL), where we can employ the Deep Neural Network to estimate Q-Values in a large Q-Table. However, it is worth avoiding the complexity and instability of DRL [17] for small scale applications for which Double Q-Learning's efficiency is proven in this work.

Acknowledgment

This work is supported by the Central Queensland University Research Grant RSH5137.

References

[1] Y. Zeng, M. Debbah, D. Gesbert, I. Guvenc, S. Jin, and J. Xu, "Integrating UAVs into 5G and Beyond (Guest Editorial)," vol. 26, no. 1, pp. 10–11, February 2019.

[2] G. Sylvester, *E-agriculture in action: Drones for agriculture*. Bangkok, Thailand: Food and Agriculture Organization of the United Nations and International Telecommunication Union, 2018.

[3] A. Fotouhi, H. Qiang, M. Ding, M. Hassan, L. G. Giordano, A. Garcia-Rodriguez, and J. Yuan, "Survey on UAV cellular communications: Practical aspects, standardization advancements, regulation, and security challenges," vol. 21, no. 4, 2019.

[4] J. Hassan, A. Bokani, and S. S. Kanhere, "Recharging of flying base stations using airborne RF energy sources," in *IEEE WCNC Workshop (WCNCW)*, 2019.

[5] S. A. Hoseini, J. Hassan, A. Bokani, and S. S. Kanhere, "Trajectory optimization of flying energy sources using Q-learning to recharge hotspot UAVs," in *IEEE INFOCOM Workshops*, 2020.

[6] N. Wang, Y. Xin, J. Zheng, J. Wang, X. Liu, and Y. Liu, "Priority-oriented trajectory planning for UAV-aided time-sensitive IoT networks," in *IEEE ICC Workshops*, 2020.

[7] Y. Zeng, J. Xu, and R. Zhang, "Energy minimization for wireless communication with rotary-wing UAV," vol. 18, no. 4, 2019.

[8] S. Say, H. Inata, J. Liu, and S. Shimamoto, "Priority-based data gathering framework in UAV-assisted wireless sensor networks," vol. 16, no. 14, 2016.

[9] H. V. Hasselt, "Double Q-learning," in *Advances in neural information processing systems*, 2010, pp. 2613–2621.

[10] D. Zorbas, T. Razafindralambo, L. Di Puglia Pugliese, and F. Guerriero, "Energy efficient mobile target tracking using flying drones," vol. 19, 06 2013, pp. 80–87.

[11] Y. Zeng and R. Zhang, "Energy-Efficient UAV Communication With Trajectory Optimization," vol. 16, no. 6, pp. 3747–3760, June 2017.

[12] C. D. Franco and G. Buttazzo, "Energy-Aware Coverage Path Planning of UAVs," in *2015 IEEE International Conference on Autonomous Robot Systems and Competitions*, April 2015, pp. 111–117.

[13] M. Bliss and N. Michelusi, "Power-constrained trajectory optimization for wireless UAV relays with random requests," in *2020 IEEE ICC*, 2020.

[14] B. Zeng, L. Yao, and W. Hu, "Priority based data reporting algorithm in wireless sensor networks," *Journal of Shanghai Jiaotong University (Science)*, vol. 22, no. 1, pp. 60–65, 2017.

[15] S. Salehi, A. Bokani, J. Hassan, and S. S. Kanhere, "AETD: An application aware, energy efficient trajectory design for flying base stations," in *IEEE MICC*, December 2019.

[16] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[17] O. Anschel, N. Baram, and N. Shimkin, "Averaged-DQN: Variance reduction and stabilization for deep reinforcement learning," in *International Conference on Machine Learning*, 2017, pp. 176–185.