Reinforcement Learning for UAV Trajectory Control

YANG Zheyuan
yz019@ie.cuhk.edu.hk
Department of Information Engineering
The Chinese University of Hong Kong

ZHANG Yuming
zy219@ie.cuhk.edu.hk
Department of Information Engineering
The Chinese University of Hong Kong

Abstract

In this project, we study the optimal trajectory of an unmanned aerial vehicle (UAV) acting as a base station (BS) to serve multiple users. We leverage reinforcement learning (RL) with the UAV acting as an autonomous agent in the environment to learn the trajectory that maximizes the sum rate of the transmission during flying time. By applying Q-learning, a model-free RL technique, an agent is trained to make movement decisions for the UAV. We compare value iteration method and DQN method.

1 Introduction

Compared to traditional mobile network infrastructure, mounting base stations (BSs) or access points (APs) on unmanned aerial vehicles (UAVs) promises faster and dynamic network deployment, the possibility to extend coverage beyond existing stationary APs and provide additional capacity to users in localized areas of high demand, such as concerts and sports events.

In this work and as depicted in figure 1, we consider the UAV acting as a BS serving multiple users maximizing the sum of the information rate over the flying time. Our work focuses on a different scenario where the UAV carries a base station and becomes part of the mobile communication infrastructure serving a group of users. Movement decisions to maximize the sum rate over flying time are made directly by a reinforcement Q-learning system.

Previous works not employing machine learning often rely on strict models of the environment or assume the channel state information (CSI) to be predictable. In contrast, the Q-learning algorithm requires no explicit information about the environment and is able to learn the topology of the network to improve the system-wide performance. We compare table-based value iteration and a neural network as Q-function approximation.

We put our code in github, Here is the *project code link*

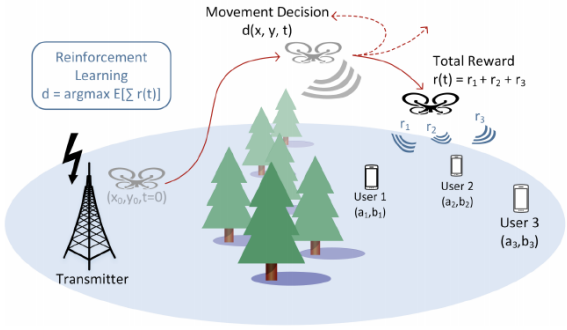


Figure 1: System Model

2 System Model

Consider the uplink (UL) for a wireless cellular network involving *K* ground user equipment (UEs). We assume these UEs will stay at same position as time goes by. Our goal is control a UAV agent which deployed as aerial base stations (BSs) to server the ground UEs. For *kth* user, the uplink is defile as the link from UE *k* to UAV agent. This graph shows the system model.

2.1 Ground-to-Air Path Loss Model

The fundamental communication model considered here is the uplink multiple access (MAC). The UEs are assumed to transmit data to the UAV using frequency division multiple access (FDMA) over orthogonal channels with equal bandwidth B and there is no interference. For the UAV-aided networks, as the altitude of the UAV is much higher than that of the ground users, the line-of-sight channels of the UAV communication links are assumed to be dominant. Therefore, the channel gain from UE k to UAV at time t can be described as the free-space path loss model

$$g_k(t) = g_0 d_k(t)^{-2} = \frac{g_0}{h^2 + ||a_u(t) - a_k||^2}, \tag{1}$$

where g_0 represents the channel gain at the reference distance $d_0 = 1$ m and d_k is the distance between user k and the UAV m. Assume that the channel capacity is fully utilized. Then the transmission rate of UE k is expressed as

$$R_k = B \log_2(1 + \frac{P_k g_k}{B\sigma^2}) \tag{2}$$

where P_k is the transmit power of UE k, B is the bandwidth and σ^2 is the noise power spectral density.

2.2 UAV's Mobility Model

The location of UAV at time t is denoted as $a_u(t) = (x(t), y(t), h)$ with time-varying horizontal coordinates and fixed height. The trajectory of UAV are subject to the maximum speed constraint within each period T:

$$\dot{a}_u(t) \leq V_{max}, 0 \leq t \leq T, \tag{3}$$

where V_{max} is the maximum UAV speed in meter/second (m/s). The period T is discretized into N equal time slots by $n \in \mathcal{N} = \{1, ..., N\}$. The time slot length $\Delta = \frac{T}{N}$ is chosen to be sufficiently small such that the location of a UAV is considered as unchanged within each time slot even at the maximum speed. Consequently, the trajectory constraint can be rewritten as:

$$||a_u[n+1] - a_u[n]||^2 \leq S_{max}^2, n = 1, ..., N-1, \tag{4}$$

where $S_{max} \equiv V_{max}\Delta$ is the maximum horizontal placement that the UAV can travel in each time slot.

2.3 Problem Formulation

The objective is to maximize the sum of UEs' data rates $\sum_{k=1}^{K} R_k$ over the whole epoch. The optimization problem can be presented as:

$$\max_{\mathbf{a}} \quad \sum_{n=1}^{N} \sum_{k=1}^{K} B \log_2(1 + \frac{P_k g_k}{BN_0}), \tag{5}$$

$$\text{s.t.} \quad v[n] \leq V_{max}, n = 1, ..., N-1, \tag{6}$$

3 Environment

We do the implementation based on system model. Figure 2 shows the learning framework for our environment.

We define the whole graph as $1000m \times 1000m$ square. The UAV cannot move outside this graph. In the graph, there are limit number of UEs. The objective of UAV agent is maximizing the cumulative transmission rate for all UEs. Here let us define the environment:

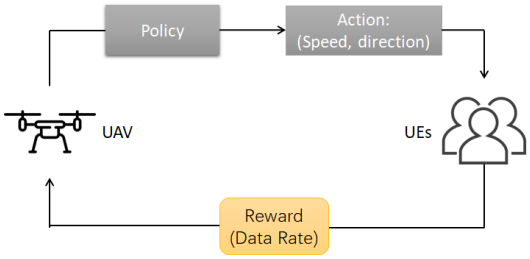


Figure 2: Environment Framework

- **State:** Current position of UAV in the graph
- **Action space:** ["left", "right", "front", "behind"]. 4 discrete action. At any time, UAV agent need to choose one of them.
- **Speed**: Using fixed maximum speed, which is *20m* or *50m* per second
- **Transition:** Using fixed transition, after UAV agent takes an action, the reward and next state is deterministic.
- **Limit Total step:** Rather than take infinite step to find the optimal point for UEs geographic distribution, UAV agent need to maximize its total reward alongside the path. So we have maximum step constrain. It makes different UAV initial position can affect the training result.
- **Users:** Totally K users in different position of the graph. UEs' position will affect the reward
- **Reward:** The reward at time n is define as the summation of all UEs' transmission rates: $\sum_{k=1}^{K} R_k[n]$

We set $g_0 = 10^{-5}$, $B = 10^6$, $P_k = 0.1$, $noise = 10^{-9}$ for the simulation.

4 Methodology

4.1 Value iteration

First, we try value iteration to provide a bench mark for our environment. Because the UEs position will not change over time, so there must exist an optimal point. UAV can stay at the optimal point to achieve maximum data rate. Theoretically, if the user position is known as prior, we can solve the environment as an optimization problem. So value iteration is a reinforcement learning solution which does same thing as solve optimization equation.

In this implementation, we do not actually provide the UEs' position. Alternatively, we provide the next state's reward. This implementation can also satisfy the value iteration's requirement. So we write the "get transition" function and implement value iteration.

4.2 Deep Q-learning

We implement Deep Q learning using PyTorch. The loss function is MSE loss, using Adam optimizer. We also apply the clip gradient method to make the training process more stable. Figure 3 shows the DQN structure.

5 Experiment Result

Here we compare value iteration results and deep Q-learning results. We focus on the effect of the predefined environment parameter "height". In reality, the height of UAV cannot be too low to avoid

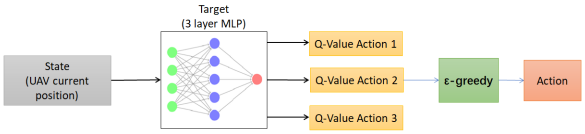
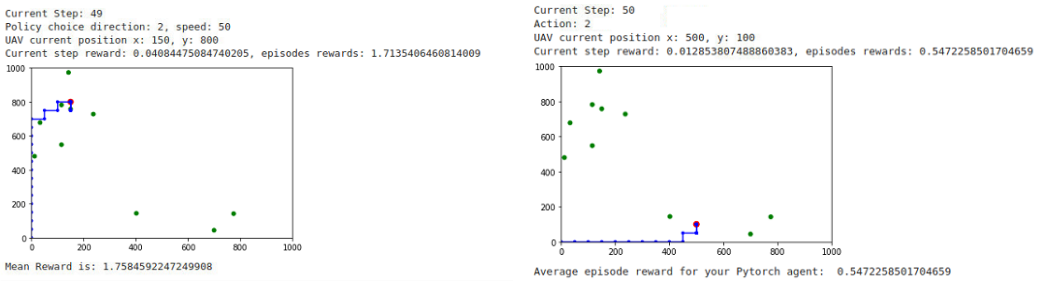


Figure 3: DQN Structure

collision. In contradiction, the height also cannot be too high, since it will affect the data rate, and it may cause conflict with other flight object. Actually in realistic, not only the height will affect the data rate, but the angle, obstacle like building and mountain, will also affect the data rate. So here we try the simulation in different height, and try to find some difference.

5.1 Lower height

We set the $height = 100m$. Since the graph is 1000×1000, $100m$ is 10% of the whole graph. As



(a) Value Iteration

(b) DQN

Figure 4: Height = 100m, Path, converge point comparison for value iteration and DQN

shown in figure 4. The green points are the UEs' position. Red point is the final point for the UAV. Blue path is the trajectory path. We set the initial point at the left bottom position $(0, 0)$. Here is screen capture for one of the random environment. Here are our observations:

- Value iteration and DQN can have different path for same environment
- For most of the testing environment, DQN can have 40-80% performance for value iteration
- When height is $100m$, the optimal converge point is around a particular user. Because lower height can make one user achieve dominant transmission rate.

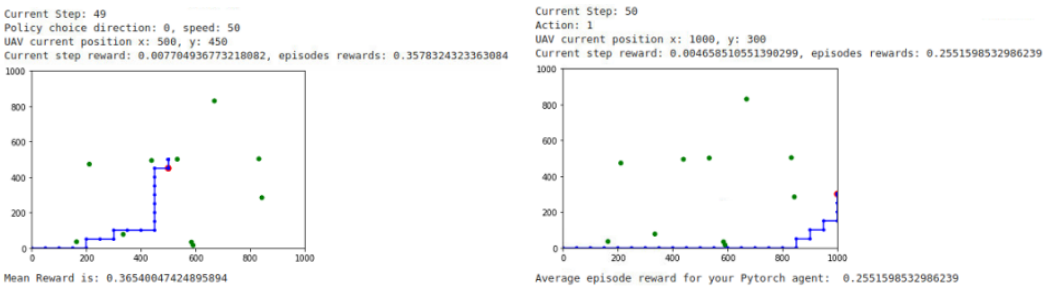
5.2 Higher height

We set $height = 300$ to simulate the higher height situation. The result is shown in figure 5. We observe the following results:

- Value iteration and DQN can have different path for same environment
- Sometimes DQN cannot converge to the optimal point, but most of the time it can.
- When height is $300m$, the optimal point will converge to the mean value point among all UEs.

6 Conclusion

In this project, we create a simulation environment for UAV wireless service reinforcement learning. We are trying to apply DQN to solve the optimal path of a UAV which service particular group of users. From the experiment result, we find that DQN can get good result for this simple environment.



(a) Value Iteration

(b) DQN

Figure 5: Height = 300m, Path, converge point comparison for value iteration and DQN

Next step of apply reinforcement learning to our research is make this environment more complex. For example, introduce continuous action space, or make the direction be angle. Also we can add variant speed and give UAV maximum speed contains. Moreover, we can let the UEs move among time, this will make the task very hard to solve.