Autonomous Networking - Homework 1

A.Y. 2021-2022

Andrea Gasparini Edoardo Di Paolo Riccardo La Marca

1. Introduction

In this homework we addressed a drone routing problem using Reinforcement Learning techniques. We worked in a setting with n drones, 3 *ferry* ones and $n - 3$ sensing ones. A sensing drone is able to collect information (i.e. temperature and humidity) and it can transfer the packets to other drones, while a ferry is only able to receive and deliver packets. Each drone has its own path to follow, that is different from the others and each packet is identified by an *event_id*. We had to implement two functions: *relay_selection* and *feedback*. The former is called at each *timestep* of the simulation and here the drone decides whether to pass or not the packets; while the latter is called by each drone when a packet gets lost (in this case we have an *outcome* equals to *-1*) or when a drone delivers packets to the depot (with *outcome* equals to *1*). We are going to explain in details these functions in subsection 2.1.

2. Approach

In the general scenario a drone can perform two possible actions:

1. Transmit: the drone decides to transmit the packets to a *drone x* among its neighbours;

2. Not transmit: the drone decides to keep the packets.

How a drone decides to which neighbour transmit the packets differs depending on the implemented algorithm (subsection 2.1).

2.1. Implemented algorithms

We implemented different algorithms but we always handled the *Q-table* as a floats' list of length *n*, differently initialized depending on the algorithm.

2.1.1. AI Routing

In this algorithm the *relay_selection* function performs *exploration* or *exploitation*. Exploration is always chosen at the first iteration of the simulation and then again based on a probability ϵ. In the *exploration* case the drone selects randomly one of its neighbours, including itself, while in the *exploitation* case it selects a drone d based on a d_s (namely the *drone score*) defined as follow:

$$d_s = \begin{cases} +\infty & \text{if } d_d \leq dp_{cr} \\ \alpha \cdot \frac{d_r}{d_m \cdot d_d} \cdot Q[d_{ID}] & \text{otherwise} \end{cases} \tag{1}$$

where dp_{cr} is the depot communication range, α is a constant with value 1.5, d_r is the residual energy of the drone, d_m the maximum energy of the drone, d_d the distance between the drone and the depot and, finally, $Q[d_{ID}]$ the action value in the Q-table for the drone. According with this equation, the drone that perform the relay selection choose a new drone, from its neighbours, that is closest to the depot and that maximize the d_s. In the case there is no such drone, it simply keeps the packet.

About the *feedback* function, we decided to give a negative reward equals to -2 when the *outcome* is -1, otherwise the reward is:

$$r = 2 + \frac{2 \cdot r_e}{m_e \cdot d} \tag{2}$$

where r_e is the residual energy, m_e is the max energy and d is the delay. In this way, the reward is directly related to the drone's energy and to the interval of time between the creation of the event and the current timestamp.

At the end we update the Q-table as follow:

$$Q[d_{ID}] = Q[d_{ID}] + \frac{1}{N[d_{ID}]} \cdot (r - Q[d_{ID}]) \tag{3}$$

where $N[d_{ID}]$ is the number of times that the drone d has been selected as the best.

Autonomous Networking, Sapienza University of Rome, 1st semester A.Y. 2021-2022.

2.1.2. Optimistic AI Routing

This algorithm uses the same formulas of the AI Routing but in this case we initialized the Q-table with a positive value equals to 5.

2.1.3. Double AI Routing

In this algorithm we decided to update also the action value of the drone that is passed as parameter in the feedback function. In this case we added this update to the feedback:

$$Q[d_{ID}] = \begin{cases} r & \text{if } N[d_{ID}] = 0 \\ \textit{Equation 3} & \text{otherwise} \end{cases} \tag{4}$$

where r is the reward and, in this case, d is related to the drone in input. The *relay_selection* is the same as in the subsubsection 2.1.1. This aims to make a drone more or less reliable, in order to take better decisions in future timesteps.

2.1.4. UCB Routing

In this algorithm the reward function is the same as in the 2.1.1 but we changed the *relay_selection* function; in particular we select the best action according to the Equation 5 and we do not have neither ϵ nor α.

$$\text{score} = \begin{cases} Q[d_{ID}] + c \cdot \sqrt{\frac{\log t_s}{N[d_{ID}]}} & \text{if } N[d_{ID}] > 0 \\ +\infty & \text{otherwise} \end{cases} \tag{5}$$

where c is a constant and $\log t_s$ is the logarithm of the current timestamp. The feedback function is the same as in the subsubsection 2.1.1. Note that in the case a drone has never been selected by another drone, we give it a priority by setting its score to infinity.

3. Experiments and results

In order to obtain the results shown in section 4 we performed hyperparameters tuning. In this section we are going to see the performed experiments and the final values we used. All the algorithms use the same values for the positive and negative rewards, respectively set to -2 and as defined in Equation 2.

The first three AI Routing algorithms (subsubsection 2.1.1, 2.1.2 and 2.1.3) have the same common hyperparameters: ϵ and α, which have been respectively set to 0.02 and 1.5 to obtain the best results. Figure 9 shows the tuning of ϵ and the score obtained with the different values.

For the UCB Routing (subsubsection 2.1.4) we tried multiple c values but we did not obtain interesting results; it was slightly better than the random routing though.

We performed several experiments with 5, 10, 15, 25, 30 and 35 drones, using 10 different random seeds. We tried seeds with values ranging from 1 to 10 and from 20 to 30 and since we obtained the same results in all the cases, we reported only the plots related to the seeds $1 - 10$. As we can see in Figure 1 all the AI approaches reach a great number of events and the AI routing appears to be the best one; instead, as shown in Figure 2, when we increase the number of drones the AI routing still performs very well but it does not always reach the highest number of events. In Figure 3 we can see that the AI routing does not have the lowest mean delivery time with a small amount of drones, but as soon as we provide a greater number of drones (Figure 4) it achieves the best performance. Regarding the score, our goal is to minimize this formula:

$$s = 1.5 \cdot |E| \cdot \text{TTL} + \sum_{p \in D} \delta_t \tag{6}$$

where E is the set of expired packets, TTL is the Time To Live, D is the set of the delivered packets and, finally, δ_t is the delivery time. In Figure 7 and 8 we can see that all the AI approaches are perform well and that the AI routing is one of the best ones. Since we are delivering the AI routing, in Figure 10 we can see the average rewards over the feedbacks in a simulation of 300k steps with 15 drones and it converges between 1.25 and 1.50.

4. Contributions

The algorithms' implementations have been developed jointly with meetings on a VoIP software. However, each of us contributed in the following way:

1. **Andrea Gasparini:** performed experiments with 25, 30 and 35 drones, Optimistic AI routing, average rewards implementation;
2. **Edoardo Di Paolo:** performed experiments with 5, 10 and 15 drones, Equation 1 and base AI Routing;
3. **Riccardo La Marca:** performed the hyperparameter tuning, UCB routing and Double AI.

Appendices

A Other approaches' source code

In the following the source code of the other approaches:

- **UCB Routing**;
- **DoubleAI Routing**;
- **OptimisticAI Routing**.

B Figures

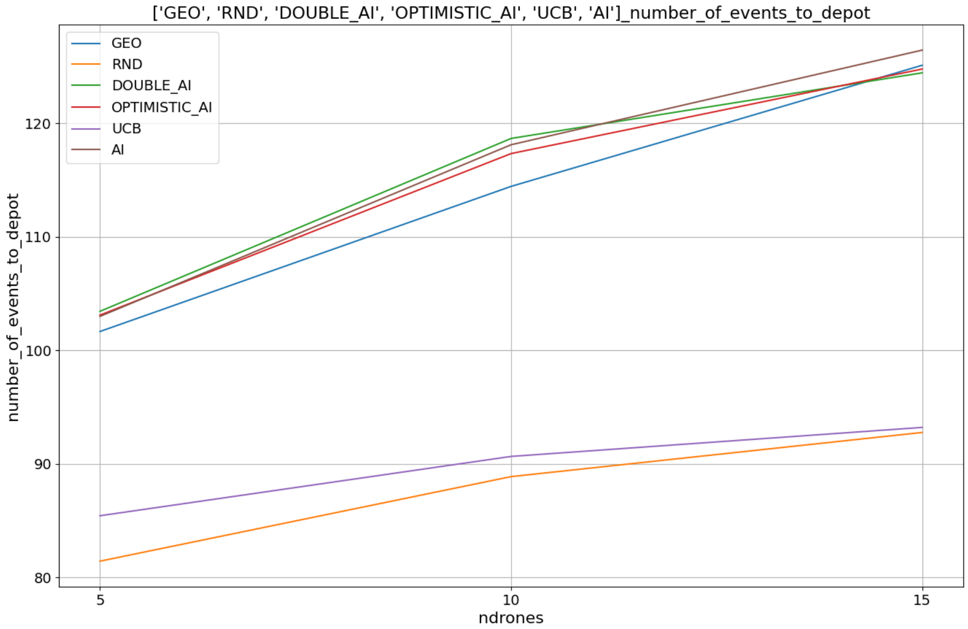

Figure 1. Number of events to the depot for simulation with 5, 10 and 15 drones.

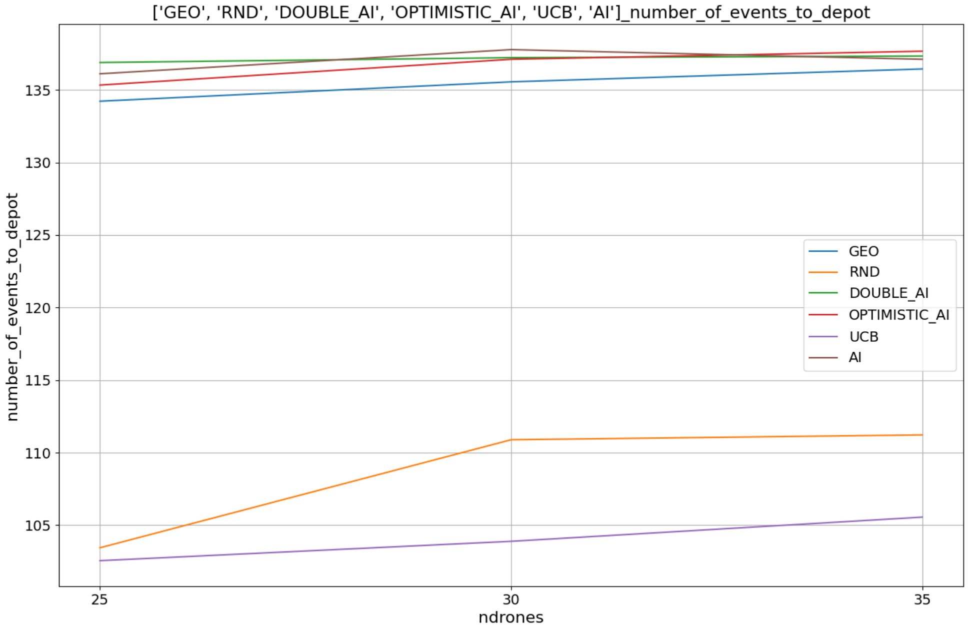

Figure 2. Number of events to the depot for simulation with 25, 30 and 35 drones.

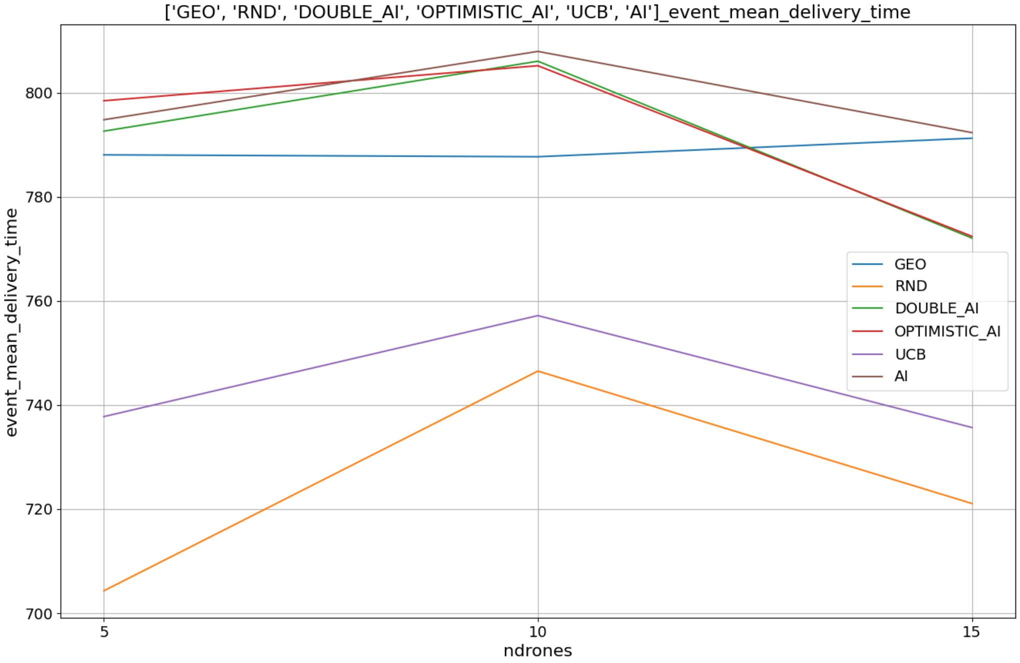

Figure 3. Mean of delivery time for simulation with 5, 10 and 15 drones.

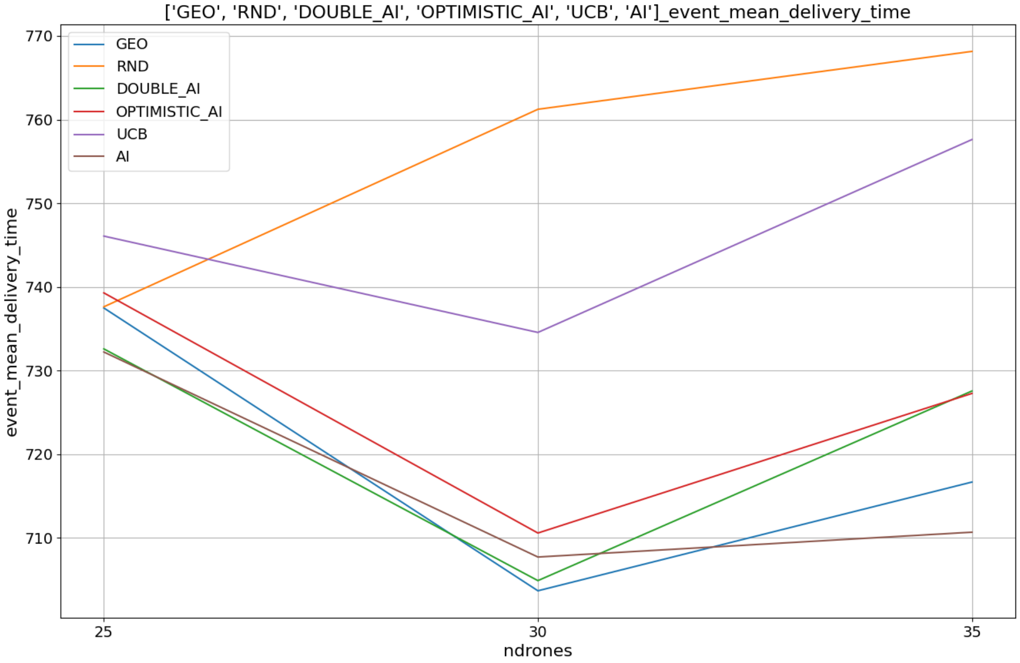

Figure 4. Mean of delivery time for simulation with 25, 30 and 35 drones.

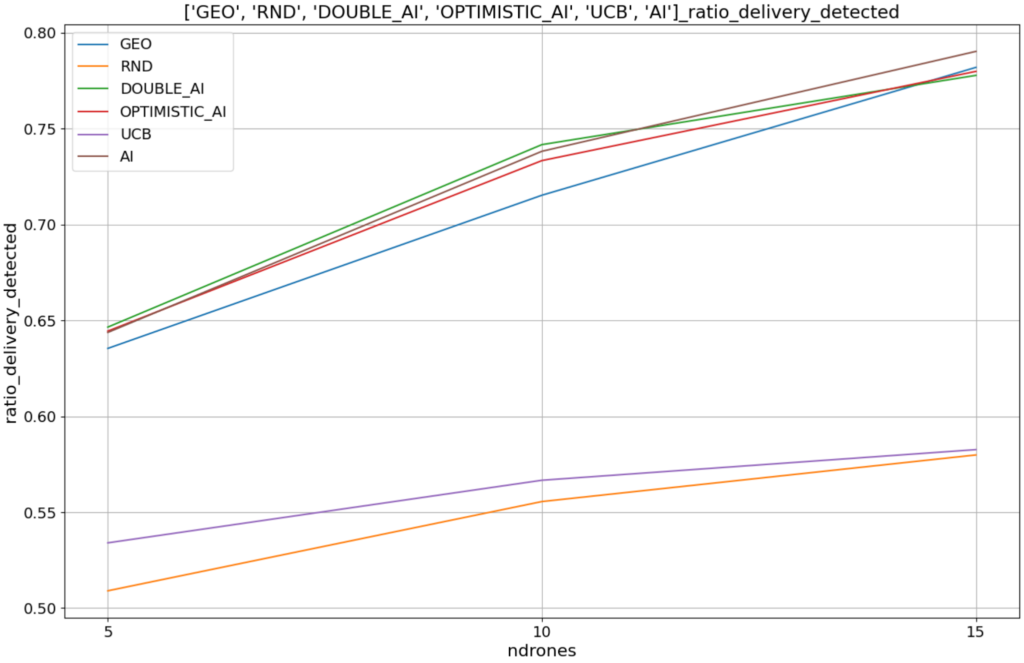

Figure 5. Packet delivery ratio for simulation with 5, 10 and 15 drones.

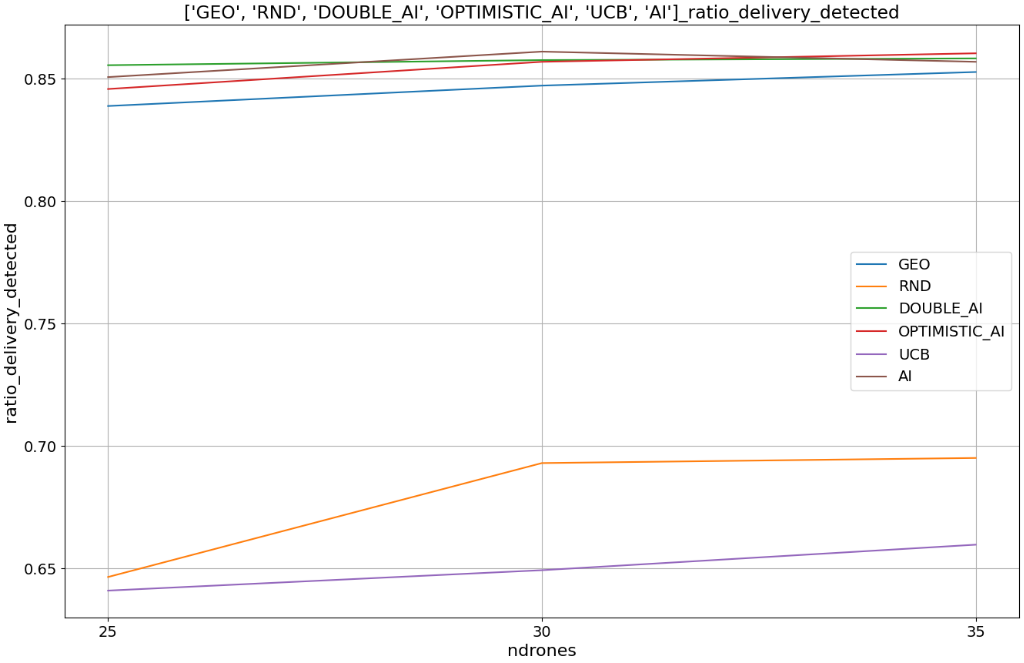

Figure 6. Packet delivery ratio for simulation with 35, 30 and 35 drones.

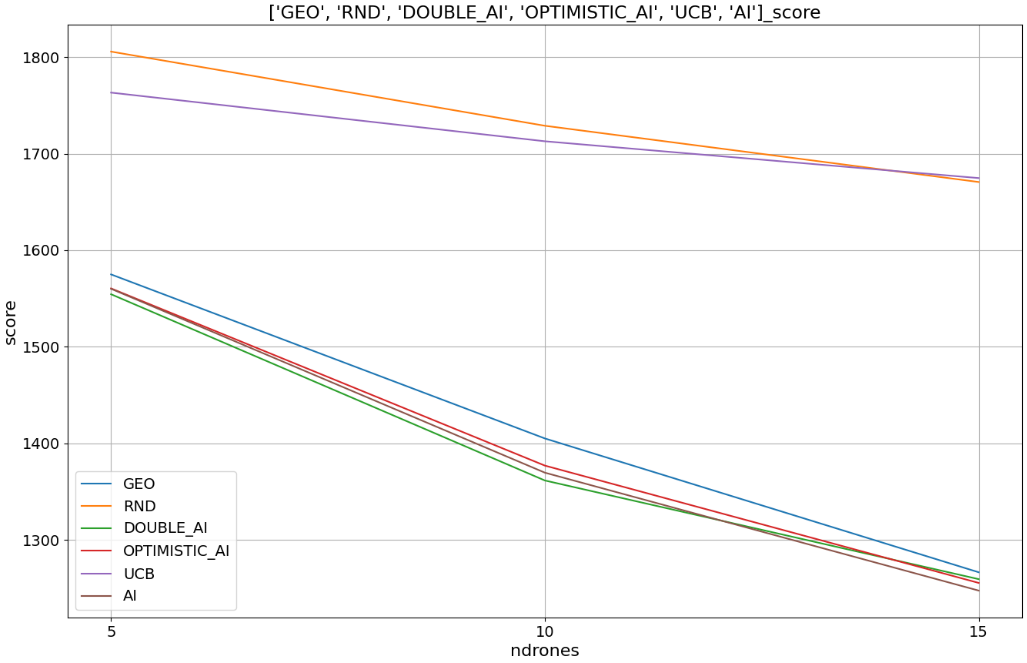

Figure 7. Score for simulation with 5, 10 and 15 drones.

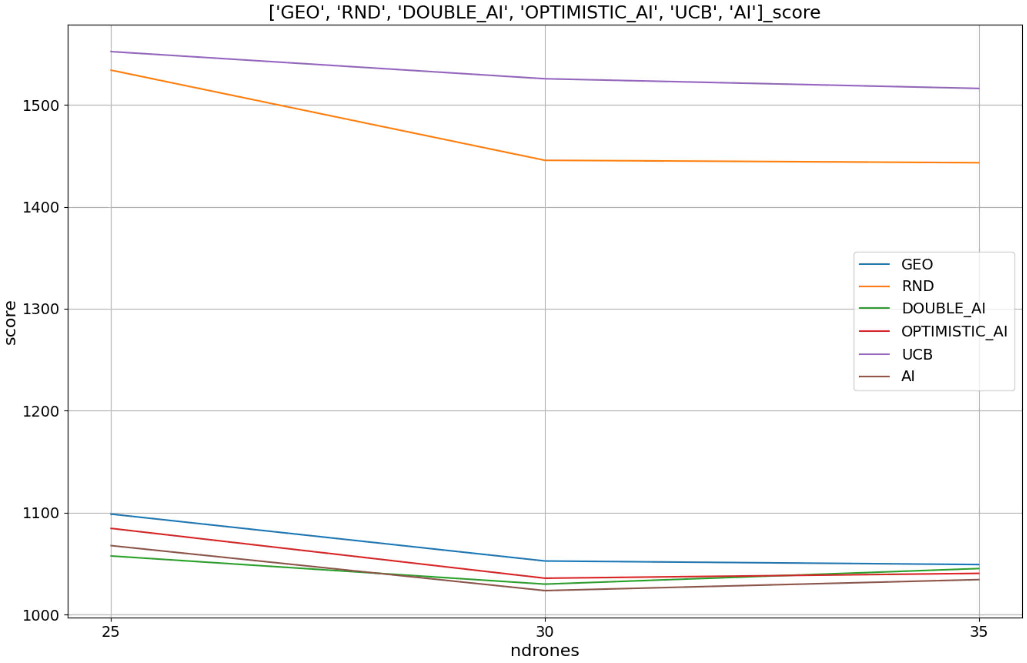

Figure 8. Score for simulation with 25, 30 and 35 drones.

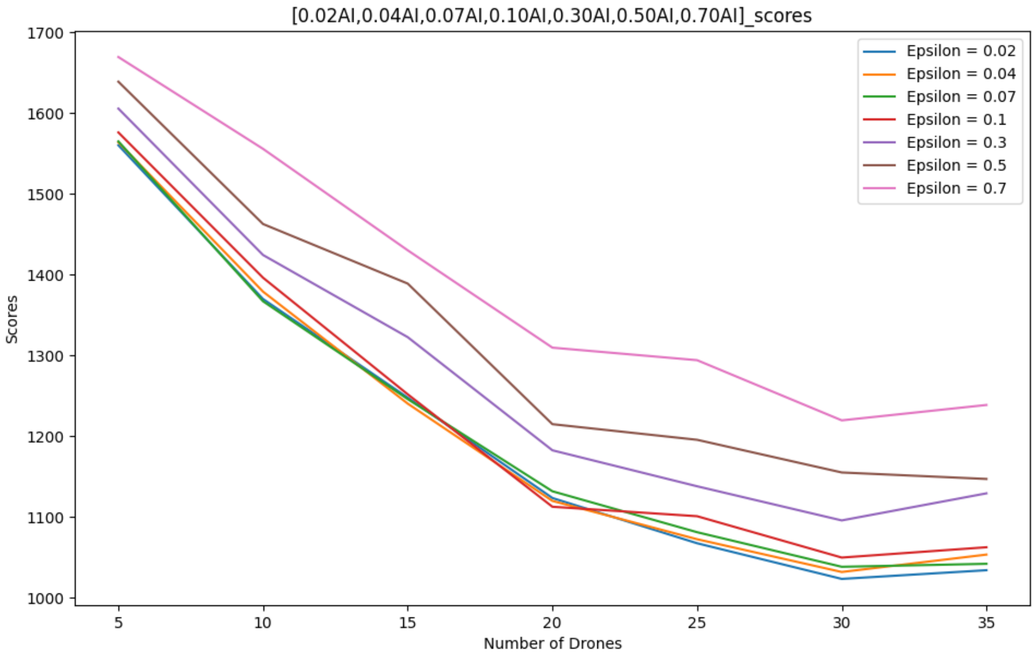

Figure 9. AI Routing ϵ tuning related to the score.

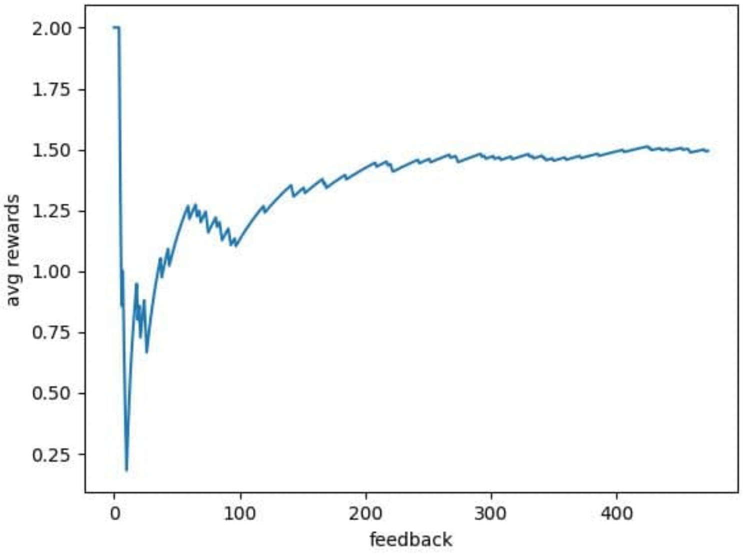

Figure 10. Average rewards with 300k steps and 15 drones on AI routing.