Homework 1 - Autonomous Networking

Giovanni Pica[1], Andrea Bernini[2], and Donato Francesco Pio Stanco[3]

[1] Student Identification Number: 1816394
[2] Student Identification Number: 2021867
[3] Student Identification Number: 2027523

1 Introduction

The problem that we are going to discuss in this report is the *Patrolling Scenario Problem*, based on creating a *Reinforcement Learning Routing Protocol* for deciding to keep or send a packet to other neighbours drones. In this problem we have a squad of drones (N $\geq$ 5):

- 3 drones work as ferries, they move back-and-forth to the depot, to deliver data;
- N-3 drones are performing monitor tasks.

We need to deliver as many packets to the repository as possible for the main task and as a secondary task to reduce packet latency by minimizing the final score.

2 Methodology

2.1 Reinforcement Learning Approach

In this section we discuss the approaches to solve the Patrolling Scenario Problem. For creating this algorithm we used the Bandit Algorithm and the action choice mechanism is based on ϵ-Greedy selection, we generate a random number between 0 and 1, if this number is lesser than epsilon we take a random action, otherwise we take the action with the greater Q_value. The actions in this algorithm are described as follows:

- **1**: keep the packet;
- **0**: send the packet to other drones by using a ModifiedGeoRouting algorithm.

So we need to store Q_values and n associated to each drone, for this we use two arrays of two values (we got two actions) that store Q_value associated to each action and which times we call that action. When we choose the action we save that and we upgrade the Q_value with also the reward which is computed in this way:

$$reward = event\_duration - delay$$

that maximizes our algorithm, because we give a bigger reward for the drones that have delivered packets and a smaller reward for the drones that keep the packet.

2.2 AI Routing based on paths

Initially we had thought of managing the reward based on the "state" of the drone's path (AIP). Indeed, our idea was to check if the next waypoint was the depot, and give higher rewards based on the fewest number of waypoints to cross before returning to the depot. But as you can see from the graphs below, the performances were not very good, so we chose the previous idea.

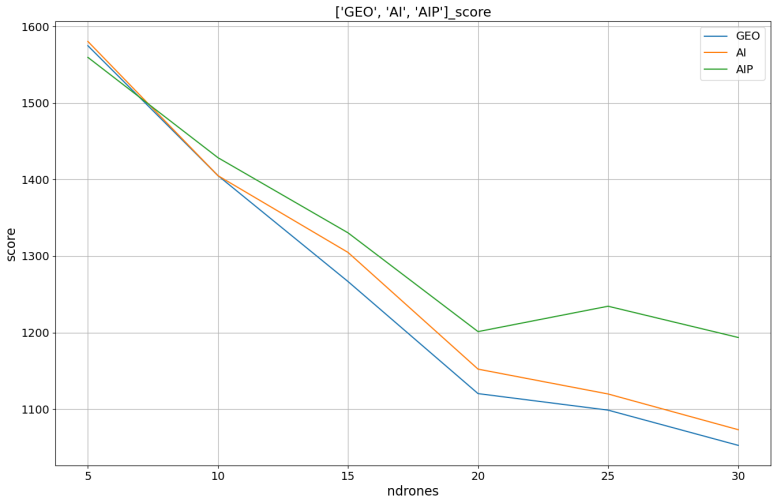

Fig. 1: Score for AI, AIP, GEO algorithms with seed 1 to 10

2.3 ModifiedGeoRouting function

Feedback function is used to return a possible feedback if the destination drone has received the packet, we also use this function to upgrade Q_values and also to fill the array for the drones that lost packets if the outcome equals to -1. We also managed the possibility to receive multiple feedbacks for the same packet with the use of a dictionary with key the id of the event and values an array of drones that have received feedback for that event.

2.4 Issues

We had some issues including the fact of not being able to overcome the GeoRouting score and so we had to find a better approach (handle expired packets and check if the drone is close to the depot) and another problem was to handle

repeated feedback due to network errors. Another problem involved `numpy` function `argmax()`, used for the selection of q_value, in fact the latter returns the leftmost index in the case of equal values. Initially we had set the exploit corresponding to index 1, while the exploration to 0, however this approach in case of equal values preferred the explore action, which worsened the algorithm, so we decided to invert the meaning of the indices, improving performance.

3 Experimental study

In this section we show performances of each approaches:

- RND: Random Routing,
- GEO: Geographical Routing,
- **AI**: Reinforcement Learning Routing based on ϵ greedy,
- **Greedy**: A modified version of AI with ϵ equal to 0.
- **UCB**: Upper Confidence Bound.
- **OPT**: Optimistic initial values.

The highlighted algorithms have been developed ourselves.

3.1 Setup

All experiments in this section, unless otherwise specified, 48000 steps (len_test), which correspond to three hours of mission. The libraries used by the project are the python3 standard library and Numpy.

3.2 ϵ Choice

As a first step we had run several tests on the AI algorithm, to decide which is the best choice for the ϵ value. To reduce the number of graph plotting runs we decided to modify the simulator, and print multiple epsilon values at the same time. We have chosen to print epsilon values quite different from each other to have a better view (0 which is Greedy, 0.01, 0.02, 0.06, 0.07, 0.09). From the graph in Figure 2 it turns out that the best choice is to set epsilon equal to 0.02.

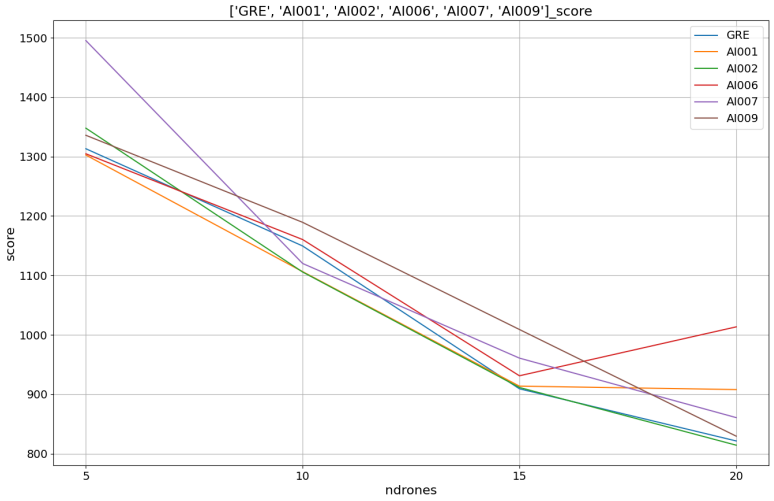

Fig. 2: All ϵ score

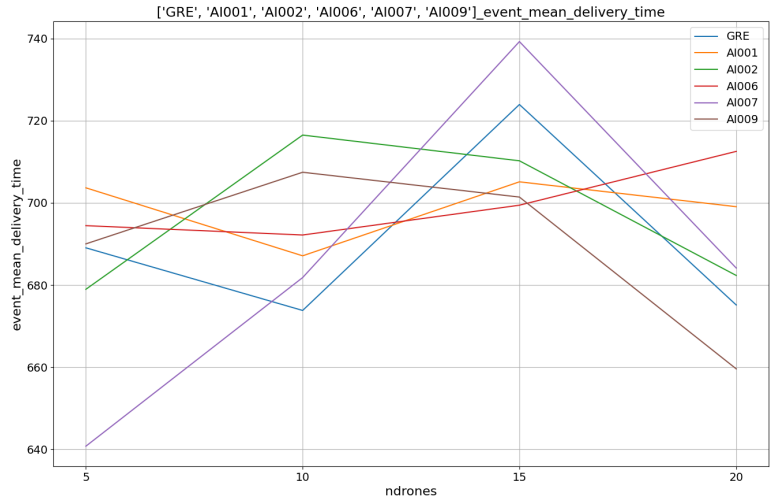

Fig. 3: All ϵ event mean delivery time

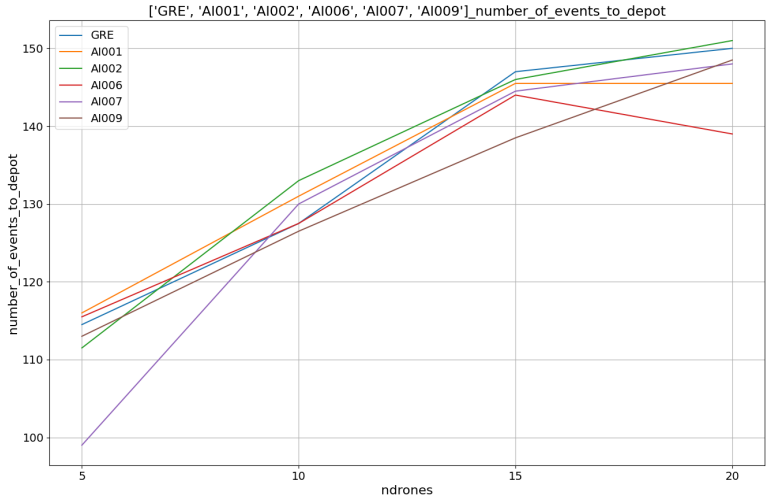

Fig. 4: All ϵ number of events to the depot

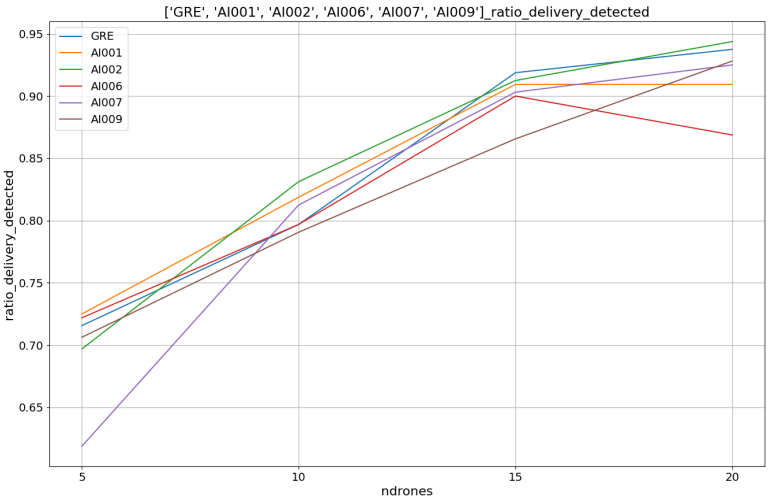

Fig. 5: All ϵ ratio delivery detected

4 Testing

In this section we show the plots on the performance of the various algorithms (GEO, RND, AI, UCB, OPT). For the plots we use 30 drones and seed 1 to 30 (produces an average from 1 to 30 for the seed).

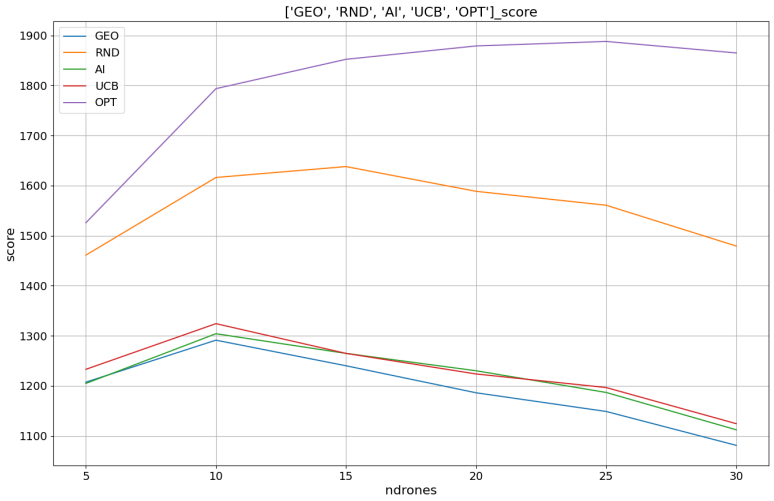

Fig. 6: Score for all algorithms with seed 1 to 30

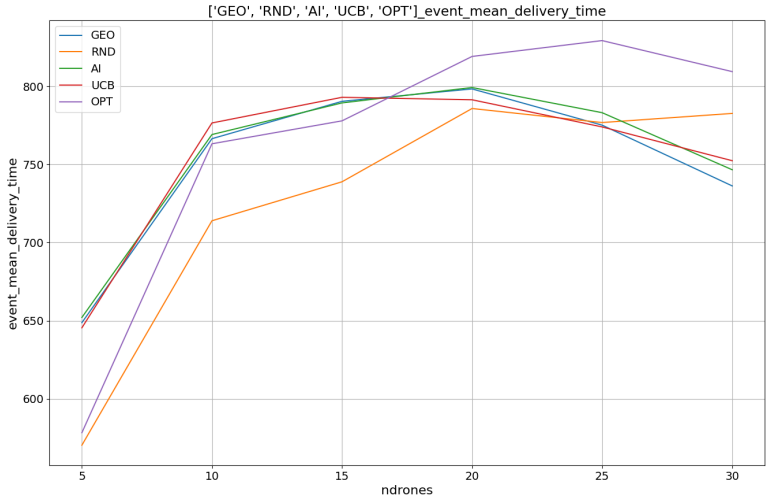

Fig. 7: Mean delivery time for all algorithms with seed 1 to 30

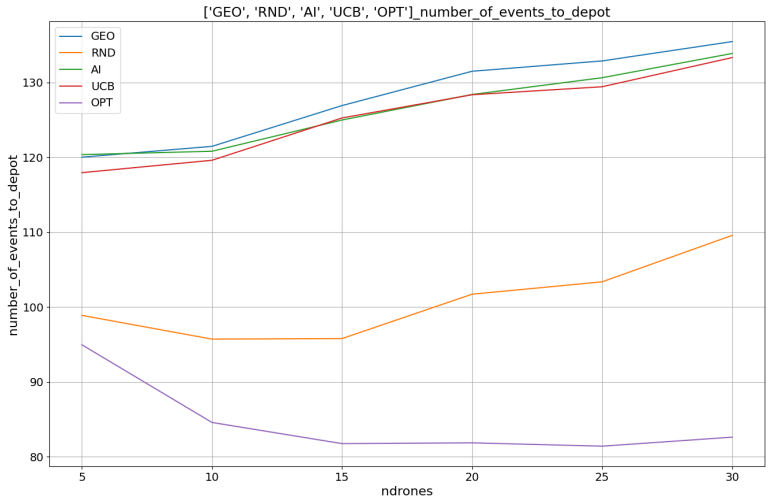

Fig. 8: Number events to depot for all algorithms with seed 1 to 30

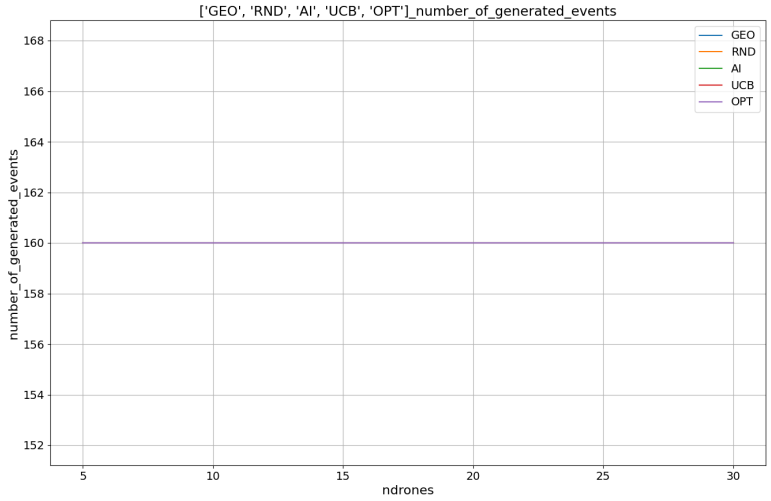

Fig. 9: Number events generated for all algorithms with seed 1 to 30

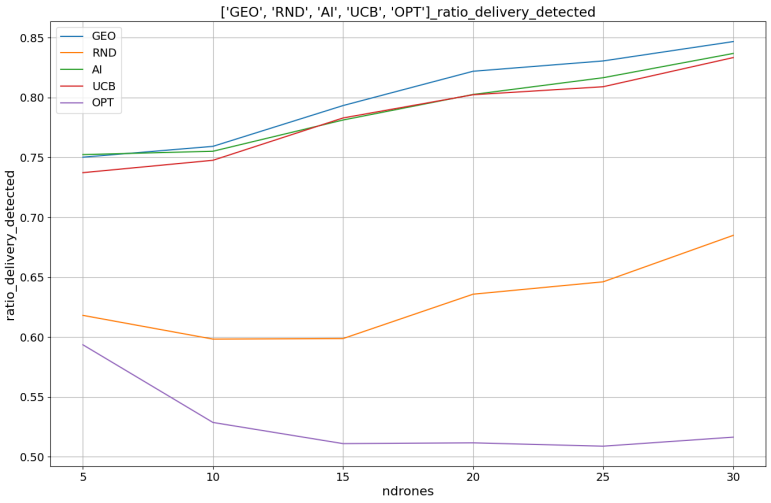

Fig. 10: Ratio delivery detected for all algorithms with seed 1 to 30

5 Conclusions

The approach used to carry out the homework is the following. Initially we collaborated through VoIP calls and screen sharing to define the algorithm to use and write an initial draft, discussing the problems encountered and proposing various solutions. Subsequently, each of us made an initial modification of the algorithm by running tests on it, in order to compare ourselves for any changes to be made to the main algorithm. Finally, the testing and plot of the various algorithms and epsilons used was equally divided to maximize times.

6 Parts developed

6.1 Giovanni Pica

We work together to the entire project and we divide things such plots, testing or modifying algorithm and in particular I was involved on the implementation of the ModifiedGeoRouting algorithm.

6.2 Andrea Bernini

We collaborated for most of the project, specifically I was involved in developing and testing a first version of the AI algorithm based on the "state" of the path, the modification of the simulator to be able to print multiple epsilons at the same time.

6.3 Donato Francesco Pio Stanco

We collaborated for most of the project, in particular I was involved in the development and testing of the UCB algorithm and of the tests and graphs of the various comparisons present in the document.

7 Code of algorithms

- **AI**: `https://pastebin.com/PpRbQgzQ`
- **Greedy**: `https://pastebin.com/CDVt96NL`
- **UCB**: `https://pastebin.com/v3H6djnS`
- **OPT**: `https://pastebin.com/Z8y7Y2jk`
- **AI Path**: `https://pastebin.com/92n3kFkL`