Exercise 2: A Reactive Agent for the Pickup and Delivery Problem

Group №54: Oriol Barbany Mayor, Natalie Bolón Brun

October 7, 2019

1 Problem Representation

1.1 Representation Description

The topology of the problem is defined by the weighted graph $\mathcal{G} = (V, E)$, where V is the set of cities and E represent the roads between them. The distance in kilometers between cities $s, s' \in V$ is denoted as $d(s, s')$.

States: Our state space is the set of cities in the problem, i.e. $\mathcal{S} = V$. The state of the agent at a given time step corresponds to the city where it is currently located.

Actions: The actions are represented by a tuple of two elements: Accept/Refuse and the destination city. Note that a given task can have any destination except the source, and if the task is refused, we restrict the next movement to neighbor cities. We apply this constraint in order to avoid skipping tasks found when going from the source to the destination without a task, thus leading to suboptimal results.

Formally, our action set for a city $s \in V$ is $\mathcal{A}(s) = \{Accept\} \times V \setminus s \cup \{Refuse\} \times \mathcal{N}(s)$, where $\mathcal{N}(s)$ is the set of neighbors of city s. We consider all the set of actions when computing the Q-values, but in the simulation, not all the actions are available in all cities. Indeed, there is at most one $\{Accept\}$ action, so the set of possible actions $\mathcal{A}(s)$ will be constrained. On the one hand, if there is no task available in s for a given time step, the possible actions will be limited to the subset $\{Refuse\} \times \mathcal{N}(s)$. We then define $Best(s)$ to be the best action of this restricted set.

On the other hand, if there is a task to city $s' \in V \setminus s$, then $\mathcal{A}(s)$ will be constrained to $\{Accept\} \times \{s'\} \cup \{Refuse\} \times \mathcal{N}(s)$. Hence, in this case best action will be $\arg\max\{Best(s), \{Accept\} \times \{s'\}\}$.

Reward table: The rewards are given by two factors: the actual reward of pursuing a task from i to j given by $r[i, j]$, a table accessed through the `TaskDistribution` object, and the cost of movement. Let c be the cost per kilometer, then we compute the reward for each state-action pair as follows:

$$R(s, a) = r[s, \pi_2(a)]\mathbb{1}_{\{\pi_1(a)=Accept\}} - c \cdot d((s, \pi_2(a))) \tag{1}$$

, where $\mathbb{1}_{\{\cdot\}}$ is the indicator function and π_i is the projection operator defined by $\pi_1(x, y) = x$, $\pi_2(x, y) = y$. In the case of an action a from an state s, $\pi_1 \in \{Accept, Reject\}$ and $\pi_2(a) \in V \setminus s$ (the destination city). Note that $d((s, \pi_2(a)))$ is the distance from city s to the destination city, i.e. the kilometers separating them.

Probability transition table: This table represents the probability to reach a city s' through action a starting at city s. In our case, $p[i, j]$, a table accessed through the `TaskDistribution` object, gives the probability of having a task from city i to city j.

$$T(s, a, s') = \begin{cases} p[s, s'] & \text{if } \pi_1(a) = Accept \\ \frac{1}{|\mathcal{N}(s)|}(1 - \sum_{s' \in V \setminus s} p[s, s']) & \text{otherwise} \end{cases} \tag{2}$$

Note that $\pi_1(a) = Refuse$ only for actions leading to neighbor states, so by our formulation, we make sure that

$$\sum_{a \in \mathcal{A}(s)} T(s, a, s' = \pi_2(a)) = 1 \quad \forall s$$

and when we don't have task, the probability to move to any neighbor city is uniform.

1.2 Implementation Details

The stopping criteria for the iterative computation of the Q-values and V-values is based on the absolute value of the difference between the previous and the updated Q-value, which we will denote $\Delta_q(t)$ for iteration t. We don't consider the difference of V-values since

$$\begin{aligned} |V_{t+1}(s) - V_t(s)| &= |\max_a Q_{t+1}(s, a) - \max_a Q_t(s, a)| \leq \max_a \{|Q_{t+1}(s, a) - Q_t(s, a)|\} \\ &\leq \max_{s,a} \{|Q_{t+1}(s, a) - Q_t(s, a)|\} := \Delta_q(t + 1) \end{aligned}$$

so the difference in Q-values will always be bigger than that of V-values. Hence, we take the maximum absolute difference between Q-values when updating all the values for all state-action pair, and if it's bigger than $\varepsilon = 10^{-5}$ we perform another iteration of all the values until the stopping criteria is satisfied, i.e. until iteration t such that $\Delta_q(t) \leq \varepsilon$.

2 Results

2.1 Experiment 1: Discount factor

Different discount factors have been used to understand its influence in the system.

2.1.1 Setting

The experiment has been conducted with 7 different agents. Each of them was assigned a different discount factor ranging from 0 to 0.95. Moreover, in order to enhance the influence of the discount factor in the cumulative reward, two different experiments were conducted: one with the initial cost of 5u/km and a second one with a cost of 50u/km.

2.1.2 Observations

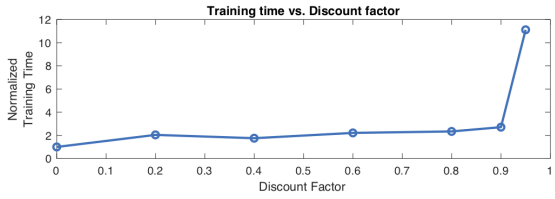

Figure 1: Evolution of training time for different discount factors

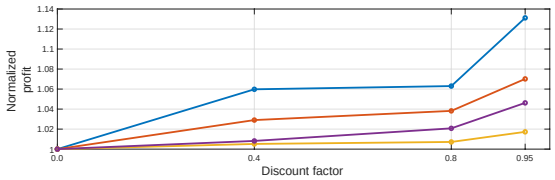

Figure 2: Evolution of profit for different discount factors. Each line represents a different seed

The first effect we can see is the influence in the training time. As we can see in Figure 1, the time required to converge when computing the Q values increases as we increase the discount factor. If we select as reference the time required for a discount factor of 0, we can see that for higher discounts up to 0.9, this time doubles. Once we overpass this late value, the required time for training increases much faster, resulting in 7 times more time than the base case for a discount of 0.95.

On the other hand, the discount value also has an impact in the average total revenue. This factor reflects how much importance we give the future possible rewards vs. the current one. Therefore, it determines which actions will our agent perform.

Nevertheless, we can see in Figure 2 that in this case, the variance produced by changing the seed is greater than the variance given by changing our discount factor, showing a low impact of this parameter in the final profit of the agent.

We have studied 2 different cases, varying the cost per km of the vehicles to point out the influence of the discount factor. For low values of cost, we can see that the difference between the policies performance is not very relevant and all agents achieve a similar reward. It is not the same case when we increase the cost per km. In this case, the gap in revenue achieved between large or small discount factors increases, pointing out the need of a non-zero value to improve performance.

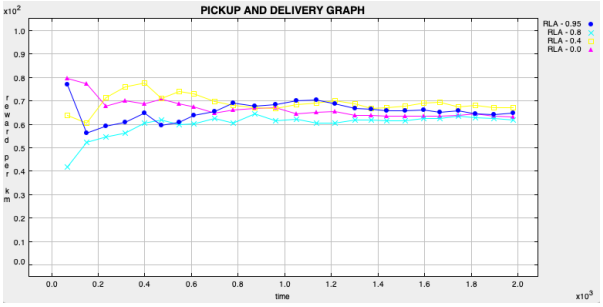

Figure 3: Evolution of reward per km with cost = 5u/km

Figure 4: Evolution of reward per km with cost = 50u/km

2.2 Experiment 2: Comparisons with dummy agents

2.2.1 Setting

To evaluate the performance of the reactive agent, it is compared with two other agents: the random agent already provided and the dummy agent. The dummy agent is such that it only moves to neighboring cities. When located in a city, the agent will accept a task only if its destination is a neighbor city. Otherwise, if there is no task or the destination is not among its neighbors, it move to a city selected randomly among its neighbors with equal probability.

The capacity of the vehicles is left much higher than the weight of the packets. The cost per km is left to 5u/km. For the dummy agent, the pickup probability is set to 0.85 while the discount factor for the reactive agent is set at 0.8.

2.2.2 Observations

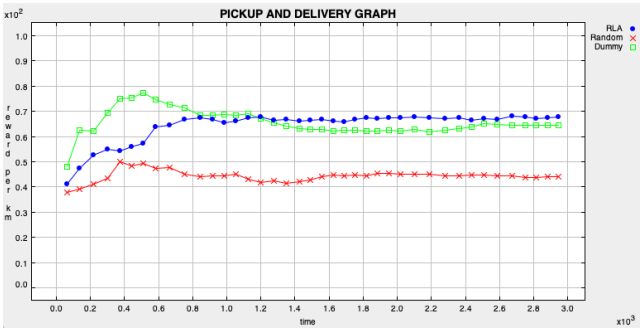

Figure 5: Evolution of reward per km for three different agents

Given the results on Figure 5, we can see that the worst performing agent is the random one. The other two agents are following a policy with the aim to obtain the maximum average accumulated reward. In this case, the reactive agent achieves a higher reward although the dummy agent is quite close in terms of performance. This can be explained since the dummy agent will reduce its travel cost by not accepting tasks implying long distance travels and still, given the size and connectivity of the graph, it has high probability to find tasks between neighbor cities. In this case, the strategy designed imposing an intuitive behaviour produces a similar behaviour to the one obtained by optimizing an average reward.