University of Queensland

COMP7702 Assigment 3 - Report

Grid World and Pacman

Author:
Duy Long Tran

Student Number:
44992305

November 1, 2019

Question 1

The MDP problem for Grid World is defined as follows:

1. **State Space:** $\mathbf{S} = \{(x, y) \mid x \in \{0, 1, 2, 3\}, y \in \{0, 1, 2\}\}$
2. **Action Space** $\mathbf{A} =$ { 'north', 'south', 'east', 'west', 'exit' }
3. **Transition Function:**

$$\mathbf{T}(s, a, s') = \mathbb{P}(s_{t+1} = s' \mid s_t = s, a_t = a) \quad \forall\ s, s' \in \mathbf{S}, a \in \mathbf{A}, t > 0$$

4. **Reward Function:**

$$\mathbf{R}(s, a, s') = \begin{cases} 1 & \text{if } s' = (3, 2) \\ -1 & \text{if } s' = (3, 1) \\ 0 & \text{otherwise} \end{cases}$$

The Bellman equation used for this problem is:

$$V^*(s) = \max_{a \in \mathbf{A}} \left(\sum_{s' \in \mathbf{S}} \mathbf{T}(s, a, s') \left(\mathbf{R}(s, a, s') + \gamma V^*(s') \right) \right)$$

Question 2

For batch value iteration, the agent always prefers the terminal state with higher reward, unless we have a very small discount rate, and a big noise parameter. The small discount rate will eliminate the effect of high reward as it is discounted to the start state, and the big noise also do the same in the sense that the next action is highly unintended, that means the difference of reward between the states has no effect.
Thus, to cross the `Bridge Grid`, we should use big discount rate (close to 1), and small noise parameter. Here, an instance setup is `discount` $= 0.9$ and `noise` $= 0.0$.

Question 3

a) Prefer the close exit (+1), risking the cliff (-10)

To get the close path, we should use small discount rate which brings down the weights of distant states. Also, the noise and the living reward should be zero since we don't want to explore the distant path as soon as the close path is already found. An instance answer is `discount` = 0.2, `noise` = 0.0, and `living reward` = 0.0.

b) Prefer the close exit (+1), avoiding the cliff (-10)

Similarly to question a, we should keep the discount rate small. Now we need to explore the distant states even if the close path has been found. Thus, the noise parameter here should be greater than zero, but also not too big since if a move from a state is highly unintended, it might be illegal. An instance answer is `discount` = 0.2, `noise` = 0.2, and `living reward` = 0.0.

c) Prefer the distant exit (+10), risking the cliff (-10)

From question a, we keep noise and the living reward unchanged to have a path beside the cliff. The discount rate should be set bigger since we need to bring up the weights of distant states. An instance answer is `discount` = 0.9, `noise` = 0.0, and `living reward` = 0.0.

d) Prefer the distant exit (+10), avoiding the cliff (-10)

This option is the combination of question b and c, where we need to increase both discount rate and noise parameter. In addition, the living reward should be increased so that the agent will prefer the distant exit, rather than the close exit. If the living reward is greater than the reward at the close exit (1.0), then the agent never ends up with the close exit. Thus, an instance answer is `discount` = 0.9, `noise` = 0.2, and `living reward` = 1.0.

e) Avoiding both exits, avoiding the cliff

Similar to question d, if we set the living reward greater than the reward of the distant exit, then the agent will never visit that exit. Thus, we can avoid both exits, and the cliff by taking `living reward` = 10.0. However, as running the trained agent, we need to visit a `TERMINAL STATE` to stop 1 episode of running. Therefore, the agent will run forever in this setup. An instance for discount rate and noise is `discount` = 0.2, `noise` = 0.0.

Question 4

Q-learning is one type of Temporal Difference Learning, which is a **model-free method**. In Q-learning, the agent initializes 0 to Q-value of all pairs `(state,`

`action)`, then updates the Q-value of the current state after each steps of an episode. The episode finishes when the agent reaches a `TERMINAL STATE`.

Q-learning is **active learning** where given the current state, the agent decide the data to use by choosing the next action from an algorithm. For example, ε-greedy chooses the action that gives maximum Q-value with (1-ε) chance, otherwise randomly chooses among the legal actions of the current state.

After a pre-specified number of episodes, the learning is finished and the optimal policy is extracted from the Q-value table, where we prefer choosing an action with higher Q-value.

Question 5

ε-greedy algorithm chooses the action which gives highest Q-value (best action) with $(1-\varepsilon)$ chance, and randomly chooses among the legal actions of the current state with the remaining chance. This is one method for the agent to decide which data to use when it stands in the current state. The first option of ε-greedy is exploitation where the agent choose to use the (previous) best action, and the second option is exploration where the agent have some chance to use a new action. If $\varepsilon = 0.5$, the exploitation and exploration is balanced. If we want to achieve a distant `TERMINATE STATE`, ε should be increased so that the agent have more chance to get to that distance. Otherwise, for a close `TERMINATE STATE`, a small ε is more consistent.

An alternative is Upper Confidence Bound (UCB), which choose the action that gives the highest sum of exploitation and exploration value.

$$\pi(s) = \underset{a \in \mathbf{A}}{\operatorname{argmax}}\ Q(s,a) + c\sqrt{\frac{\log(n(s))}{n(s,a)}}$$

The exploitation value is just Q-value of the pair `(state,action)`. The exploration value is a function of the number of times the `state` visited, the number of pairs `(state,action)` visited, and a constant parameter c, which has a similar role as ε.

Question 6

For `Bridge Grid`, the agent need to explore as most as possible to cross the bridge. Thus, the best ε we can have is 1.0. The learning rate has an effect only if the agent can reach the distant exit. However, the exploration seems

to be not powerful enough to reach the distant exit (+10) within 50 episodes. Upon the trial of 10 runs, the furthest state the agent can reach is (3,1) which is at the middle of the bridge. Therefore, the answer for this question is `'NOT POSSIBLE'`.

Question 8

The state space in `PacmanQAgent` is the set of all possible configurations of the board, including the positions of ghosts and the positions of existing foods. The size of state space will increase exponentially as increasing the dimensions of the board. For a reduction on the space, `ApproximateQAgent` represents a single state by 4 main features: "#-of-ghosts-1-step-away", "closest-food", "eats-food", and "bias". By doing this, many configurations of the board which share the same "feature state" will lead to the same best action from the agent.

For example, suppose we have a 9x9 square board with the food at the center of the square, and one ghost moving around. Considering 2 configurations: (i) the ghost at the top-left corner, and (ii) the ghost at the bottom-right corner. The `PacmanQAgent` will consider them as 2 separate configurations, which is unnecessary since the agent is not in danger if it is close to the food at the center. In `ApproximateQAgent`, the 2 configurations have the same feature extraction, then they can be treated as one state and the agent can inherit the training results from one to each other.

At the decision stage, the agent will choose the best action that maximize the Q-value which is computed as a weighted sum of the four features. By adjusting the weights. the Q-value of the explored states will be updated so as to take the negative rewards of "#-of-ghosts-1-step-away" and positive rewards of "eats-food".