RBE 595 — Reinforcement Learning
Assignment #6
Model-Based Reinforcement Learning

Arjan Gupta

Problem 1

What is "planning" in the context of Reinforcement Learning?

Answer

In the context of Reinforcement Learning, planning is the process of using a model of the environment to improve the policy.

A model of the environment is a representation of the environment that is used to predict what next state and reward will be given a current state and action. Once we have a model, we can use it to simulate the environment and produce a simulated experience, in the form of an episode. Then we can use this simulated experience to improve the policy. This is the process of **planning**.

Problem 2

What is the difference between Dyna-Q and Dyna-Q+ algorithms?

Answer

The problem with Dyna-Q is that it does not balance exploration and exploitation well in the planning phase. This is because the planning phase is greedy. Only the learning phase is ϵ-greedy. The reason that we want to also explore in the planning phase is that the model may need to change if the environment changes over time.

Dyna-Q+ is like Dyna-Q, except that it adds an exploration 'bonus' to the planning phase. What this means is that we essentially provide a bonus reward in the planning phase for states that have not been visited in a long time. This encourages the agent to explore in the planning phase. Specifically, the bonus reward is given by:

$$R = r + \kappa\sqrt{\tau(s,a)}$$

Where r is the reward received, κ is a constant of our choice, and $\tau(s,a)$ is the number of time steps since the last visit to state s after taking action a. It is important to note that the bonus reward is only given in the planning phase, and not during regular interaction with the real environment.

Problem 3

Model-based RL methods suffer more bias than model-free methods. Is this statement correct? Why or why not?

Answer

This statement is correct. Model-based RL methods suffer more bias than model-free methods because the design of the model introduces bias. The model is a representation of the environment, and it is not possible to represent the environment perfectly. Therefore, the model will always introduce some bias.

Problem 4

Model-based RL methods are more sample efficient. Is this statement correct? Why or why not?

Answer

This statement is correct. Model-based RL methods can use a limited number of samples to learn a model of the environment. Given an episode from real-interaction, we can extract as much ‘juice’ as possible from it by using it to learn a model. Then, we can use the model to simulate more episodes, and use those episodes to improve the policy.

On the other hand model-free RL methods can only use the episode to improve the policy, so they require more ‘samples’ to learn the same amount.

Therefore, model-based RL methods are more sample efficient since they can learn more from the same number of samples.

Problem 5

What are the 4 steps of the MCTS algorithm? How does MCTS balance exploration and exploitation?

Answer

The 4 steps of the MCTS algorithm are:

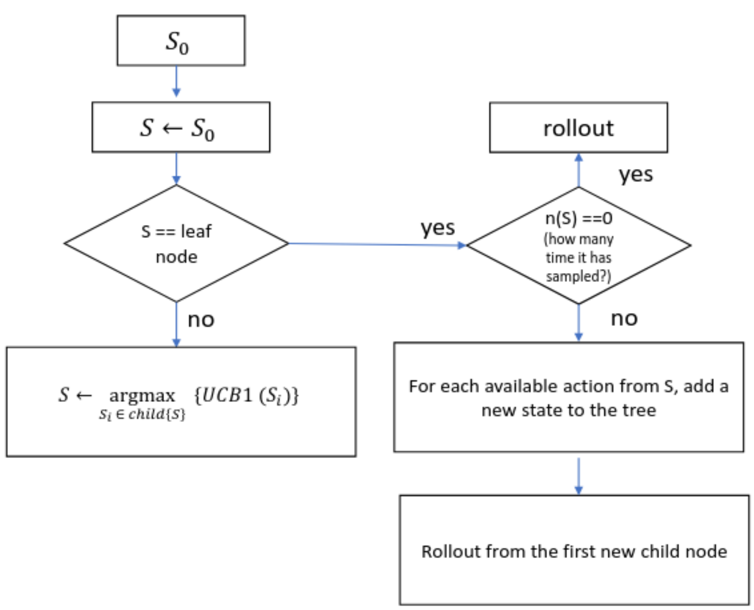
1. **Selection**: Starting from the root node, we select a leaf node with an exploration-exploitation trade-off criteria (UCB1).
2. **Expansion**: We expand the leaf node by adding a child node for each possible action.
3. **Simulation**: We simulate an episode starting from the newly added child node. We use the roll-out policy to select actions during the simulation.
4. **Back-up**: We store action-values for each node in the tree.

Specifically, in the selection step, for balancing exploration and exploitation, we use the UCB1 formula:

$$UCB1(S_i) = \frac{v_i}{n_i} + C\sqrt{\frac{\ln N_i}{n_i}}$$

Where,
n_i is the number of times we have visited node S_i,
N_i is the number of times we have visited the parent of node S_i,
v_i is the value of node S_i, and
C is a constant that we choose.

The flow chart for the MCTS algorithm (tree policy) is shown below:



Now, let us describe how we use the above flow chart to balance exploration and exploitation. At first, we just have the root node, S_0. This technically currently a leaf node, but since it is the root-node, we make an exception and immediately add new child nodes to it, and rollout from the first new child node. Based on the results of the rollout, we update the value of the first new child node and its parent.

Then, we restart the algorithm from the root node. This time, since the root node is no longer a leaf node, we need to use UCB1 to select a leaf node. Now this is where the exploration-exploitation trade-off comes in. We will find that even though the first new child node has a high value (since we rolled out from it previously), it would be the choice in case of greedy selection. However, since we are using UCB1, we will instead consider the second new child node, which has a lower value, because its exploration term is higher. Specifically, the exploration term is higher because n_i is 0, making the second term in the UCB1 for that node infinitely large. This shows that in the planning phase, MCTS balances exploration and exploitation by using the UCB1 formula.

Problem 6

(Exercise 8.1, page 166) The nonplanning method looks particularly poor in Figure 8.3 because it is a one-step method; a method using multi-step bootstrapping would do better. Do you think one of the multi-step bootstrapping methods from Chapter 7 could do as well as the Dyna method? Explain why or why not.

Answer

Let us analyze how the n-step TD method would be applied to this problem.

Firstly, in n-step TD, the $G_{t:t+n}$ is given as,

$$G_{t:t+n} = R_{t+1} + \gamma R_{t+2} + \ldots + \gamma^{n-1} R_{t+n} + \gamma^n Q(S_{t+n}, A_{t+n})$$

One thing to notice is that, as given in the problem prompt, the reward for all actions is 0 except for the goal state, where the reward is 1. Therefore, we have two cases, depending on whether the goal state is reached within n steps or not.

If $t < T$, then $G_{t:t+n}$ is given as,

$$G_{t:t+n} = 0.95^n Q(S_{t+n}, A_{t+n})$$

If $t \geq T$, then $G_{t:t+n}$ is given as,

$$G_{t:t+n} = 0.95^{k-1} + 0.95^k Q(S_{t+n}, A_{t+n})$$

Where k is the number of steps taken before reaching the goal state (i.e. $k = T - t$).

And $Q(S_t, A_t)$ is given as,

$$Q(S_t, A_t) = Q(S_t, A_t) + \alpha \left[G_{t:t+n} - Q(S_t, A_t)\right]$$