CS6700 : Reinforcement Learning
Written Assignment #3

Deadline: ??

- This is an individual assignment. Collaborations and discussions are strictly prohibited.
- Be precise with your explanations. Unnecessary verbosity will be penalized.
- Check the Moodle discussion forums regularly for updates regarding the assignment.
- **Please start early.**

AUTHOR : Arabhi Subhash
ROLL NUMBER : cs17b005

1. (3 marks) Consider the problem of solving POMDPs using Deep Reinforcement Learning. Can you think of ways to modify the standard DQN architecture to ensure it can remember histories of states. Does the experience replay also need to be modified? Explain.

Solution: Firstly, as we don't know states, the state part in DQN will be replaced by history from the start of trajectory, for q update we should be storing transitions of a trajectory in a buffer even in the normal case. We can discard it after the trajectory is completed. If we intend to do experience replay then chunks of such trajectories are maintained in a replay buffer, for update we would be selecting a transition and its history from starting of trajectory.

2. (4 marks) Exploration is often ameliorated by the use of counts over the various states. For example, one could maintain a visitation count $N(s)$, for every state and use the same to generate an intrinsic reward ($r_i(s)$) for visiting that state.

$$r_i(s) = \tau \times \frac{1}{N(s)}$$

However, it is intractable to maintain these counts in high-dimensional spaces, since the count values will be zero for a large fraction of the states. Can you suggest a solution(s) to handle this scenario? How can you maintain an approximation of the counts for a large number of states and possibly generalize to unseen states?

Solution: Having large state spaces wherein most of intrinsic reward values are zero is a problem with this mode of representation. This problem still persists in Bayesian approach or density models. To solve this problem we have a quantity in literature called Pseudo count. It generalizes across states and has the ideas of intrinsic reward. It goes on describing a density ρ over state space, a recording probability $\rho'(s) = \rho(s; s_{1:n}s)$, a pseudo-count $\hat{N}_n(s)$ function and a pseudo-count total $\hat{n}$.

$$\rho(n) = \frac{\hat{N}_n(s)}{\hat{n}}, \rho'(n) = \frac{\hat{N}_n(s)+1}{\hat{n}+1} \tag{1}$$

Here the $\hat{N}_n(s)$ is equivalent to that of our count N(s) with a density underneath it. The large state spaces with zeros in prior will have a small negligible value while retaining the values for more explored states.

3. (5 marks) Suppose that the MDP defined on the observation space is k-th order Markov, i.e. remembering the last k observations is enough to predict the future. Consider using a belief state based approach for solving this problem. For any starting state and initial belief, the belief distribution will localize to the right state after k updates, i.e., the true state the agent is in will have a probability of 1 and the other states will have a probability of 0. Is this statement true or false? Explain your answer.

Solution: Yes it's true. Having a belief state distribution for current state is equivalent to maintaining the history of the state from the start of journey. Let us say our belief distribution updated k times that means we have its history of k or more states (the length of journey is >= k). As this is a k-th order Markov, with last k observations the current state is deterministic and hence the belief state probabilities i.e. 1 for right state and 0 for others.

4. (3 marks) Q-MDPs are a technique for solving the problem of behaving in POMDPs. The behavior produced by this approximation would not be optimal. In what sense is it not optimal? Are there circumstances under which it can be optimal?

Solution: In Q-MDPs we assume we know belief state probabilities based on our near surroundings and calculate scores for actions. If we have a symmetric environment with terminal state in the center then the symmetric states (both have same belief state probabilities) on left and right will have same scores for opposite actions. We wont be able to choose the action for π from $max_{a*}Score(a)$. It can be optimal if the surrounding knowledge enough to choose action i.e. every state with similar surroundings have same action as optimal. In the previous line I am not saying if the problem is MDP, its not fixation on state based on surroundings but the constrain that different states with similar surroundings will have same optimal action.

5. (3 marks) What are some advantages and disadvantages of A3C over DQN? What are some potential issues that can be caused by asynchronous updates in A3C?

Solution: In A3C we would be performing A2C with n workers updating parameters asynchronously. Here we can use multiple CPU threads, each worker can explore different parts of environment. Not just parts each worker can explore with different strategies because of this exploration, in A3C we do not need a replay buffer to reduce variance. These are some advantages of A3C over DQN. In A3C we are not using our data efficiently but instead we are using multiple learners to consume more date within small clock time whereas replay buffers in DQN use data efficiently. This could be a problem in environments where interaction is more expensive than updating.

6. (6 marks) There are a variety of very efficient heuristics available for solving deterministic travelling salesman problems. We would like to take advantage of such heuristics in solving certain classes of large scale navigational problems in stochastic domains. These problems involve navigating from one well demarcated region to another. For e.g., consider the problem of delivering mail to the office rooms in a multi storey building.

   (a) (4 marks) Outline a method to achieve this, using concepts from hierarchical RL.

   Solution: Consider the given example, using concepts of hierarchical RL, we can solve the TSP for each floor, considering floor as a sub-task and assigning an appropriate distance factor for floor to floor transitions. Now solving for floors will solve our problem. This approach is computationally less expensive than solving with every office in the entire building. Similarly for any large scale navigational problems we need to identify the links between one demarcated region to other i.e. bottlenecks and solve TSP for that regions first. Now solving for a super TSP by putting these regions as single entities will solve our problem. This heuristic is analogous to option framework in hierarchical RL. Similarly we can also draw analogies from HAM, maxQ etc to solve this problem.

   (b) (2 marks) What problems would such an approach encounter?

   Solution: The main problem is to find the bottlenecks. Sometimes they may not be present or cannot yield optimal solutions. Consider the given example, if there are elevators at either ends of floors. Even then the floor is a demarcated region joined to other by 2 links but the optimal solution can be like delivering to some offices one one side go up and come back on other side for delivering the rest. This transition is not present in present in our approach.

7. (6 marks) This question may require you to refer to this paper on average reward RL. Consider the 3 state MDP shown in Figure 1. Mention the recurrent class for each such policies. In the average reward setting, what are the corresponding ρ^π for each such policy ? Furthermore, which of these policies are gain optimal ?

   (a) (3 marks) What are the different deterministic uni-chain policies present ?

   Solution: There are a total of 6 deterministic policies possible
   π_1 : A - a1, B - a1, c - a2 - it is uni-chain with AB recurrent class and C is transient state
   π_2 : A - a2, B - a1, c - a2 - it is uni-chain with AC recurrent class and B is transient state
   π_3 : A - a3, B - a1, c - a2 - it is uni-chain with AC recurrent class and B is transient state
   π_4 : A - a1, B - a1, c - a3 - it is multi-chain with AB and C as two recurrent classes
   π_5 : A - a2, B - a1, c - a3 - it is uni-chain with C recurrent class and A,B are transient state
   π_6 : A - a3, B - a1, c - a3 - it is uni-chain with C recurrent class and A,B are transient state

   (b) (3 marks) In the average reward setting, what are the corresponding ρ^π for each such policy ? Furthermore, which of these policies are gain optimal ?

   Solution: In uni-chain case the average reward is independent of start state and is equal to reward accumulated in recurrent class per step. For policies stated above -
   $\rho^{\pi 1} = 2/2 = 1$
   $\rho^{\pi 2} = 1/2 = 0.5$
   $\rho^{\pi 3} = 0/2 = 0$
   $\rho^{\pi 4}(A) = \rho^{\pi 4}(B) = 2/2 = 1, \rho^{\pi 4}(C) = 1/1 = 1$
   $\rho^{\pi 5} = 1/1 = 1$
   $\rho^{\pi 6} = 1/1 = 1$
   The optimal policies π^* are the one that are gain optimal, here $\pi 1, \pi 4, \pi 5$ and $\pi 6$ are gain optimal.

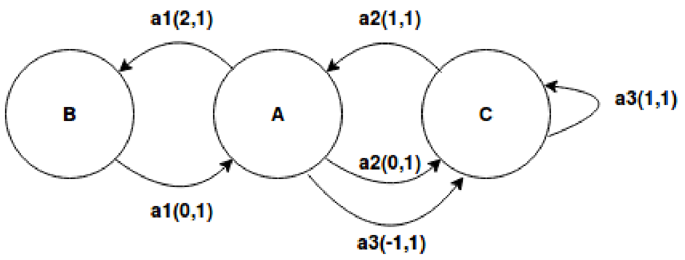

Figure 1: Notation : action(reward, transition probability). Example : a1(3, 1) refers to action a1 which results in a transition with reward +3 and probability 1