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 : Rahul Reddy V
ROLL NUMBER : ME16B171

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: Instead of solving the POMDP, we can solve an MDP with the belief state space of the POMDP as the new state space. Since we are now working with belief states we don't have to remember the histories of states. Instead we can just store the ¡b, a, r, b'¿ tuples in experience replay. However since the belief state space is continous and very high-dimensional based on the state space of the POMDP this approach would be computationally expensive.

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: The high dimensional state space can be discretized with the help a hash function and then the visitation counts for each of these hash codes can be maintained.
However the performance of this exploration method highly depends on the hash function used and the granularity of it. A popular hash function class for this task is locality-sensitive hashing(LSH) which groups similar states into one hash code and keeps distant states seperate. So, this method generalises well to states unseen before if a good hash function is chosen.

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: False. If the belief state is randomly initialized and the environment reset state(goal state) is reached within k steps, the belief state won't localize to the true state with probability 1 as the states before the reset state are useless. And also the observations have to be error free for the belief to localize to true state with probability 1.

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-MDP approach to solving the problem of behaving in a POMDP, we assume that there is an underlying MDP and we know its optimal state-action value function(Q^*), and we estimate the Q function for the belief states by taking a weighted average of the MDP state-action values.

$$Q(b, a) = \sum_s b(s) Q^*(s, a) \tag{1}$$

Though the Q function used is optimal for the fully known MDP, it's not optimal for the POMDP as we don't know the state we are in exactly and Q-MDP approach doesn't take that into consideration. The optimal policy would consider the partial observability.
Q-MDP would be optimal under the circumstances of all belief states being very close to the true state.

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:

- **Adavantages of A3C over DQN:**
  - A3C uses multiple workers sampling data simultaneously using different CPU threads, so it can get more data in less time compared to DQN(less training time).
  - Using multiple agents exploring different parts of the environment with possibly different exploration strategies which produce de-correlated samples. This eliminates need of experience replay which is present in DQN and is computationally expensive.
  - A3C can use both on-policy and off policy methods whereas DQN can only use off policy methods.
- **Disadvantages of A3C over DQN:**
  - Because the updates are done in asynchronous manner, different workers would update the same global network at different times, some workers might be computing the updates using an older version of global network but the global network was already changed by other workers. This could lead to reduce in performance. This is a potential issue due to asynchronous updates in A3C

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: The objective of this problem is to minimize the overall cost(time taken or distance covered) while delivering mail to the office rooms in a multi storey building without visiting any room more than once. One approach to solve thhis problem is to set sub tasks such as "Go to nth Floor", "Go to office room A" etc and learn option policies to these sub tasks. After learning these options, approriate positive rewards can be set for delivering mail and negative rewards for re-visiting an office room and a small negative reward for each step.

With this all setup, now we can train the agent to solve the problem using Intra Option Q-Learning or SMPD Q-learning.

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

Solution:

7. (6 marks) This question may require you to refer to https://link.springer.com/content/pdf/10.1007/BF 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: The different deterministic uni-chain policies present are:

1. $\pi_1(A) = a_1;\ \pi_1(B) = a_1;\ \pi_1(C) = a_2;$
   recurrent class of $\pi_1 : \{A, B\}$
2. $\pi_2(A) = a_2;\ \pi_1(B) = a_1;\ \pi_1(C) = a_2;$
   recurrent class of $\pi_2 : \{A, C\}$
3. $\pi_3(A) = a_3;\ \pi_1(B) = a_1;\ \pi_1(C) = a_2;$
   recurrent class of $\pi_3 : \{A, C\}$
4. $\pi_4(A) = a_2;\ \pi_1(B) = a_1;\ \pi_1(C) = a_3;$
   recurrent class of $\pi_4 : \{C\}$
5. $\pi_5(A) = a_3;\ \pi_1(B) = a_1;\ \pi_1(C) = a_3;$
   recurrent class of $\pi_5 : \{C\}$

(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:

1. $\rho^{\pi_1} = 1$
2. $\rho^{\pi_2} = 0.5$
3. $\rho^{\pi_3} = 0$
4. $\rho^{\pi_4} = 1$
5. $\rho^{\pi_5} = 1$

The gain optimal policies are π_1, π_4, π_5

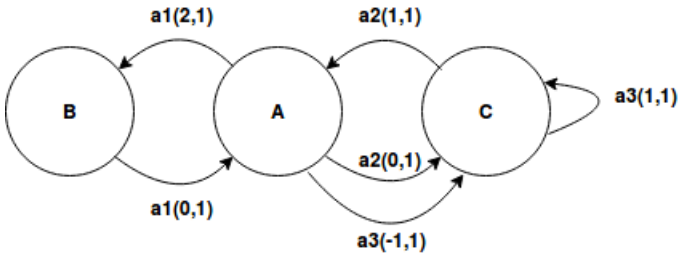

Figure 1: Notation : action(reward, transition probability). Example : a1(3, 1) refers to action a1 which results in a transition with reward +3 and probability 1