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 : Dodiya Keval.
ROLL NUMBER : CS19M023

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: We can use RNN(Recurrant Neural Network) or LSTM(Long Short Term Memory) to train network .

Replay Buffer Content: Sequence of experiences.
e.x.$\{(s_j, a_j, r_j, s'_j), (s_{j+1}, a_{j+1}, r_{j+1}, s'_{j+1}), \ldots, (s_{j+\tau}, a_{j+\tau}, r_{j+\tau}, s'_{j+\tau})\}$
here size of input length is fixed(some k-length),
and we can tune this hyper-parameter(k).
Note : here $s'_j = s_{j+1}$

Input to RNN: Sequence of experiences(k-experiences) with previous output.
e.x. take all experiences of one sequence and feed into RNN and previous output will be $h_k = Q(s_k, h_{k-1}|\theta_i)$
here size of input length is fixed(some k-length),
and we can tune this hyper-parameter(k).

Output from RNN: $Q(s_{j+\tau}, a', h_{j+\tau}|\theta)$.
Q value for last state's all actions of experience sequence.

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: we can use pseudocount approach.
let $P_\theta(s)$ is parameterized visitation density. we will represent discrete count as a density value. so probability will be high if we visited similar states frequently and for less visitation of state it'll be very small
so approximation goes like this..

- fit model $P_\theta(s)$ to all states D seen so far
- take a step i and observe s_i
- fit new model $P_{\theta'}(s)$ to $D \cup s_i$
- use $P_\theta(s)$ and $P_{\theta'}(s)$ to approximate $N(s)$.

here $P_\theta(s) = \frac{N(s)}{\eta}$, η represents total count so far.

and $P_{\theta'}(s) = \frac{N(s)+1}{\eta+1}$

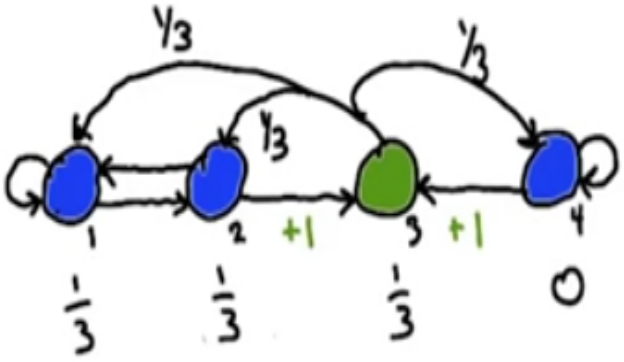
so $N(s) = \eta * P_\theta(s) = \frac{1 - P_{\theta'}(s)}{P_{\theta'}(s) - P_\theta(s)} P_\theta(s)$

here we can have 3 cases..

- $N(s) \geq 0$ iff P is learning Positive
- $N(s) = 0$ iff $P_\theta(s) = 0$
- $N(s) = \infty$ iff $P_\theta(s) = P_{\theta'}(s)$

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: Answer is False if underlying World is stochastic.we can predict next observation perfectly(with probability 1 to right local observation) but not state.



Consider above MDP where 2 observation(blue,green)and 4 states are there. blue observation consist states <1,2,3> and green observation has state <4>. transition from green observation to blue observation has 1 probability but state 3 to other states have 1/3 probability. furthermore +1 is the reward. actions space includes two actions "Left"and"Right". this MDP is 1st order MDP because one previous observation is enough to predict future(next observation) provided world is stochastic.
Suppose initial belief distribution = < 1/3, 1/3 , 1/3 , 0>.
now suppose <Left,B> has observed.
so updated distribution = < 2/9 , 1/9 , 0 , 1/9> .
so in above example with one update belief distribution doesn't converge to right state but it does converge to right observation . so it happens because one observation might have more than one state and that creates problem to decide local state and since MDP defined on observation we can predict next observation perfectly but not state.

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 POMDP we only have partial information of states .we don't have perfect state information.and that leads to give saddle point or local optima as a solution. so if we don't use perfect solving method e.x. kth-order MDP for sloving POMPDP with perfectly tuned 'k' then it is very likely to not get optimal approximation.

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: .

Advantages of A3C over DQN: As we know DQN is computationally costly and consumes more memory. so in place of history saving in A3C multiple agent walk around different part of environment asynchronously hence training data will have variety of experiences furthermore they are uncorrelated hence distribution become stationary. and another advantage is A3C allows on-policy methods unlike DQN that allows only off-policy methods. .

Disadvantages of A3C over DQN: In A3C multiple agents have same objective function but they does-not do simultaneous update and they asynchronously accessing different part of environment hence objective function keeps changing according to those agent updates.hence it might possible that earlier update got replaced by new update. so those older updates have no use! .

Issue caused by asynchronous updates: since newer asynchronous update replaces older update that leads to performance degradation because now there is zero contribution of older update.

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:

suppose we have n different office rooms. and we will take different MDPs for each different office Rooms. so Total MDPs = $\{M_1, M_2, \ldots, M_n\}$,and our Task multi storey building represent by M. these MDPs(M_i) can be solve using intra-option learning or we can implement well defined policy. Now our final Goal is to find out optimal policy for M. we can assume all these MDPs as nodes and distance between rooms represented by edges in graph and our goal is to solve travelling salesman problem for above graph. now for above configuration reward function can be defined as
$R_M = \sum_{i=1}^{n} R_i$ where R_M is the total reward of different local MDPs reward. likewise for local reward we can write as
$R_i(s, a) = 1\ if\ s = s_i, a = a_i$ else 0.

now our task is to combine all nodes so we need to find shortest path reward R_i to R_j for all (i, j).now we can use a heuristic like simulated annealing (which is explained below) to find shortest path. this algorithm would choose option with highest estimated value. so by SMDP as mentioned above where states have initial states well as reward states. and action space contains set of options we can solve problem.

In 'simulated annealing' we have one target policy(Nearest neighbour in our case) and one random action picking policy. as time progress random policy trying to be inclined with target policy keeping mind a reward.this procedure stop until reward start to degrade.we can use this heuristic here

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

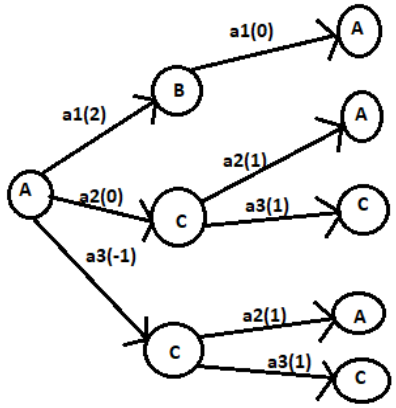
Solution:

it might possible we can not achieve optimal solution as flat optimal solution or global optimal solution. because in our solution we optimize smaller options(sub-problems) to reduce computation hence options internal policy would be optimal, but overall policies from all options might not result in optimal solution.

7. (6 marks) This question may require you to refer to https://link.springer.com/content/pdf/10.100 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:



so Above branching indicates uni-chain policies e.x. A ->B ->A. actions and reward mentioned with transition arrow. so 5 unichain policies possible by given MDP

(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 ?

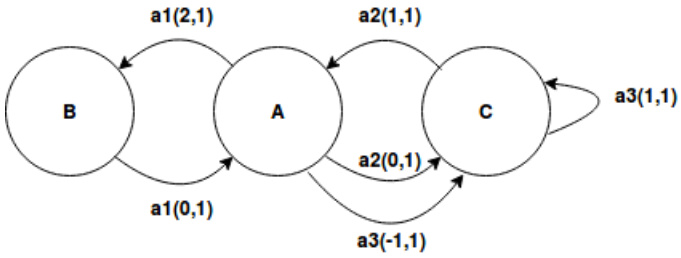
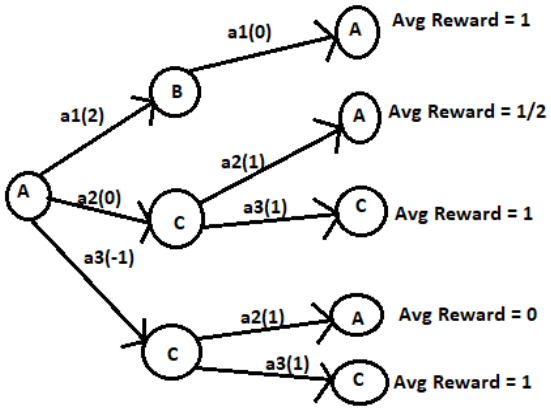


Figure 1: Notation : action(reward, transition probability). Example : a1(3, 1) refers to action a1 which results in a transition with reward +3 and probability 1

Solution:



for all unichain policies corresponding ρ^{π} mentioned in above image. and we found 3 policies gain optimal(1) which is

1) A->B>A with action (a1,a1) ,
2) A->C->C with action (a3,a3),
3) A -> C -> C with actions(a2,a3).