ASEN 5519-003 Decision Making under Uncertainty
Homework 3: Online MDP Methods

February 4, 2021

1 Conceptual Questions

Question 1. (30 pts) In the proof for Lemma 5 of the Sparse Sampling paper by Kearns, Mansour, and Ng,[1], the authors claim that if a policy π satisfies $|Q^*(s, \pi^*(s)) - Q^*(s, \pi(s))| \leq \beta$ for all $s \in \mathcal{S}$, then it immediately follows that $|R(s, \pi^*(s)) - R(s, \pi(s))| \leq \beta$. This statement is mistaken. Provide an MDP and a policy that present a counterexample to this claim and demonstrate that the statement does not hold.

2 Exercises

Question 2. (30 pts) Monte Carlo Tree Search

Write code that performs 7 iterations of Monte Carlo Tree Search for an MDP created with `HW3.DenseGridWorld()` starting at state $(19, 19)$. You will need to produce three dictionaries:

- `Q` maps (s, a) tuples to Q value estimates.
- `N` maps (s, a) tuples to N, the number of times the node has been tried.
- `t` maps (s, a, s') tuples to the number of times that transition was generated during construction of the tree.

Then visualize the resulting tree with `HW3.visualize_tree(Q, N, t, SA[19, 19])`[2]. **Submit an image of the tree and the code used to generate it**.

You will need to use the following functions from POMDPs.jl for the problem:

- `actions(m)`
- `@gen(:sp, :r)(m, s, a)`
- `isterminal(m, s)`
- `discount(m)`
- `statetype(m)`
- `actiontype(m)`

You may also wish to use `POMDPs.simulate` and `POMDPSimulators.RolloutSimulator` for the rollouts.

`HW3.DenseGridWorld()` randomly generates a 100x100 grid world problem. There is a reward of +100 every 20 cells, i.e. at [20,20], [20,40], [40,20], etc. Once the agent reaches one of these reward cells, the problem terminates. All cells also have a randomly generated cost.

[1] `https://www.cis.upenn.edu/~mkearns/papers/sparsesampling-journal.pdf`; Note: you do not need to read the paper to complete the problem.

[2] `SA` is from the `StaticArrays.jl` package.

3 Challenge Problem

Question 3. (20 pts code, 20 pts score) Fast Online Planning

Create a function `select_action(m,s)` that takes in a `DenseGridWorld`, `m`, and a state `s`, and returns a near-optimal action within 50ms. You may wish to base this code on the MCTS code that you wrote for Question 2. Evaluate this function with `HW3.evaluate` and **submit the resulting `json` file along with the code and a short description of your approach**. A score of 50 will receive full credit. There are no restrictions on this problem - you may wish to use a different algorithm, multithreading, etc.