COMPSCI 570: Artificial Intelligence

Homework 6: Markov Decision Processes (due Nov. 23 10:29pm)

Please read the rules for assignments on the course web page (http://www2.cs.duke.edu/courses/fall21/compsci570/). In particular, you must **write your own code (or whatever you use) from scratch** for this assignment. Please let us know if you have questions about what you can use. Please use Ed Discussion for questions (use private questions if your question is likely to reveal part of the answer to others). Use Gradescope to turn this in.

In this assignment, you will solve for the optimal policy and values of a small MDP, given by the following story. You will solve using both value iteration and policy iteration, which you are allowed to do in any way you like (writing your own code, a spreadsheet, by hand (!)), but you should show all your work (including your source code or whatever else you use).

A little autonomous rover on Mars depends on its solar panels for energy. Its goal is to collect as much energy as possible. It is close to a small hill; it collects the most energy when it is at the top of the hill, but it has a tendency to roll off the hill, and it takes energy to get back up the hill.

Specifically, the rover can be in one of three states: on top of the hill, rolling down the hill, or at the bottom of the hill. There are two actions that it can take in each state: drive or don't drive. Driving always costs 1 unit of energy. When it is at the top of the hill, it collects 3 units of energy; in the other two states, it collects 1 unit of energy. For example, if the rover is at the top of the hill and is driving (to stay on top of the hill), its reward is $3 - 1 = 2$.

If the rover is at the top of the hill and drives, then it is still at the top of the hill in the next period with probability .9, and rolling down in the next period with probability .1. If it is at the top of the hill and does not drive, these probabilities are .7 and .3, respectively.

If it is rolling down and drives, then with probability .3 it is at the top of the hill in the next period, with probability .6 it is still rolling down in the next period, and with probability .1 it is at the bottom in the next period. If it is rolling down and does not drive, then with probability 1 it is at the bottom in the next period.

Finally, if it is at the bottom of the hill and drives, then in the next period it is at the top of the hill with probability .6, and at the bottom with probability .4. If it does not drive, then with probability 1 it is at the bottom in the next period.

1 (25 points). Draw the MDP graphically, similarly to the machine example from class.

2 (25 points). Using a discount factor of .8, **solve** the MDP using value iteration. You should start with the values set to zero. You should do **at least 60 iterations and print out the values** at each state for each iteration (one line per iteration, in the order top, rolling, bottom). Also, **give** the policy that results at the end.

You presumably do not want to do all this by hand, but you can use whatever programming language you like (or a spreadsheet). **Submit** your code, spreadsheet, or whatever you used, separately.

3 (25 points). Using a discount factor of .8, **solve** the MDP using policy iteration (until you have complete convergence). You should start with the policy that never drives. **Show** the policy and values at each state and each iteration. (Of course, at the end they should be (approximately) the same as in **2**...).

You may do this by hand or not, but if you don't do it by hand, you should **submit** what you used separately.

4 (25 points). Change the MDP in **three** different ways: by changing the discount factor, changing the transition probabilities for a single action from a single state, and by changing a reward for a single action at a single state. Each of these changes should be performed separately starting at the original MDP, resulting in three new MDPs (which

you do not have to draw), each of which is different from the original MDP in a single way. In each case, the change should be so that the optimal policy changes, and you should **state what the optimal policy becomes** and **give a short intuitive argument for this**. *Hint:* It is fine, in fact recommended, to give the parameter that you're changing some extreme value, so that it becomes easy to see that the policy will change.