Assignment 5 – Markov Decision Processes
CS 2400 – Fall 2020
Due: December 11

In this assignment, you will write code to solve the small grid world problem we looked at in class using Value Iteration. You will also need to perform a few experiments using your code and report the results. You may work in pairs.

Code

Solve this problem using synchronous, in-place Value Iteration. In synchronous Value Iteration, you calculate new utilities for all the states on every iteration. "In-place" means that you calculate the utilities at iteration $(i+1)$ using both the utilities from iteration i and any new utilities that have already been calculated during iteration $(i+1)$. Here is Value Iteration pseudocode, from the textbook :

function VALUE-ITERATION(mdp, ϵ) **returns** a utility function
  inputs: mdp, an MDP with states S, actions $A(s)$, transition model $P(s' \mid s, a)$,
    rewards $R(s)$, discount γ
    ϵ, the maximum error allowed in the utility of any state
  local variables: U, U', vectors of utilities for states in S, initially zero
    δ, the maximum change in the utility of any state in an iteration

  repeat
    $U \leftarrow U'$; $\delta \leftarrow 0$
    for each state s **in** S **do**
      $U'[s] \leftarrow R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a)\, U[s']$
      if $|U'[s] - U[s]| > \delta$ **then** $\delta \leftarrow |U'[s] - U[s]|$
  until $\delta < \epsilon(1-\gamma)/\gamma$
  return U

You can use this as a guide, but note that it is *not* in-place Value Iteration. The change to make it so is trivial.

Your implementation should allow the user to specify a discount factor γ and a maximum allowable error ϵ in the state utilities. You should also allow the user to specify the positive terminal state reward, the negative terminal state reward, and the step cost. All of this must be specifiable on the command line. Write your program so that the arguments are given in that order. For example, in Java, assuming that your main class is `SmallGrid.java`:

```
java SmallGrid 0.99 1e-6 1 -1 -0.04
```

specifies a discount rate of 0.99, a maximum state error of 0.000001, a positive terminal state reward of +1, a negative terminal state reward of −1, and a step cost of −0.04. Your program should stop iterating when the error target is met.

NOTE: Any program that does not allow me to specify these parameters on the command line will get no credit. In fact, I won't even look at it.

You will need to code the transition function and the reward function for the problem. Arrays are the most straightforward approach, using a 3-dimensional array for the transition function and a 1-dimensional array for the reward function. I have provided Java code (`SmallGrid.java`) that takes care of initializing these. Feel free to use any or all of this code, including using it as a base and just adding your code to it.

Your program should print out:

1. all of the parameters used for that run,
2. the number of iterations in that run,
3. the final state utilities, and
4. the policy implied by these utilities.

The last two items should be in a format like the following:

```
0.776   0.844   0.905   1.000
  E       E       E       N

0.717   XXXXX   0.641  -1.000
  N     XXXXX     N       N

0.651   0.593   0.560   0.338
  N       W       N       W
```

This, by the way, was the result of solving the problem with the parameters specified in the Java command above, i.e. a discount rate of 0.99, a maximum state error of $1e-6$, a positive terminal state reward of +1, a negative terminal state reward of −1, and a step cost of −0.04.

Experiments

Run experiments to answer the following questions and describe your results in a brief report.

1. How many iterations does Value Iteration need to solve this problem? Use a discount factor of 0.99, a maximum state error of 1e-6, a positive terminal state reward of $+1$, a negative terminal state reward of -1, and a step cost of -0.04.

2. The number of iterations in the previous question is what was needed to reduce the state error to the specified level. How many iterations does it actually take before the policy is optimal?

3. Measure the effect of the following factors on the policy arrived at. Test the values suggested. When testing a given parameter, hold the other parameters constant at the values used in the previous experiment. For each parameter, describe its impact on the policy. Note that different values of a given parameter may produce the same policy. You may describe the policy produced for each value, but it is not necessary. I am interested in the overall impact of the parameter on the policy produced.

   (a) the discount factor
   { 0.1, 0.4, 0.7, 1.0 }

   (b) the step cost
   { -10.0, -0.04, 0.0, 0.04 }

   (c) the negative reward
   { -100.0, -10.0, -1.0 }

   (d) the positive reward
   { 1.0, 10.0, 100.0 }