Lab Report A8
CS4300
Derek Heldt-Werle 2,4,6
Matthew Lemon 1,3,5

12/1/16

1. Introduction

Create a MDP policy iteration algorithm to generate an optimal policy for navigating the stochastic Wumpus World environment given below:

0	0	0	G
0	0	W	P
0	0	P	0
0	0	0	0

We are allowed the actions A = { UP, LEFT, DOWN, RIGHT } where these are movements with probabilistic outcomes as described in the text (i.e., 0.8 probability of going the direction selected, 0.1 of going to either side).

Does starting out with a policy of all UPs (or any other action) rather than a random policy affect the final policy at all?

Does increasing K for the number of iterations in the Policy evaluation function help generate a better policy?

2. Method

We began by creating a runner function was created in order to properly setup the arguments to properly run our policy iteration function. In here we create the MDP, as well as specify gamma, to pass down to our policy iteration function.

The policy iteration algorithm is essentially a mirror of the policy_iteration algorithm found on page 657 of Peter Norvig and Stuart J. Russell's book Artificial Intelligence: A Modern Approach. The algorithm begins by setting our known utility values. We then populate an array with 16 random policies. The algorithm will then call our policy evaluation function, which will return the evaluated utilities using our transition model. We then calculate the action with the maximum value returned by finding the dot product of our current utility and the transition model. We continue to repeat and update our policy as long as changes are made to it.

3. Verification of Program

Running our CS4300_MDP_policy_iteration function with:
S = [1,2,...,16];
A = [UP, LEFT, DOWN, RIGHT]
P = the probabilities of getting to any other state given a state and an action
R = -1 everywhere except -1000 where there is a pit or wumpus and 1000 where there is gold
K = 10
Gamma = 0.999999
And the wumpus world described in the introduction we produce the policy below:

→	→	↑	1000
↑	←	-1000	-1000
↑	←	-1000	↓
↑	←	↓	←

Table 0: Optimal policy for stochastic environment with k = 10 and a random initial policy

Which is in fact the optimal policy given the parameters listed above.

0.8116	0.8678	0.9178	1.0000
0.7616	0	0.6603	-1.0000
0.7053	0.6553	0.6114	0.3879

Table 1: Results of Running Policy Iteration on 4x3 World with γ=.99999

4. Data and Analysis

→	→	↑	1000
↑	←	-1000	-1000
↑	←	-1000	↓
↑	←	↓	←

Table 2: Optimal policy for stochastic environment with R(s) = -1

→	→	→	1000
↑	↑	-1000	-1000
↑	↑	-1000	↓
↑	↑	←	←

Table 3: Optimal policy for stochastic environment with R(s) = -100

→	→	→	1000
↑	↑	-1000	-1000
→	→	-1000	↑
↑	↑	↑	↑

Table 4: Optimal policy for stochastic environment with R(s) = -500

↓	←	↑	1000
→	←	-1000	-1000
↓	←	-1000	↓
↑	←	↓	↓

Table 5: Optimal policy for stochastic environment with R(s) = 1

→	→	↑	1000
↑	←	-1000	-1000
↑	←	-1000	↓
↑	←	↓	↓

Table 6: Optimal policy for stochastic environment with k = 100

→	→	↑	1000

↑	←	-1000	-1000
↑	←	-1000	↓
↑	←	←	←

Table 7: Optimal policy for stochastic environment with k = 1

5. Interpretation

To determine if initial policy of the algorithm made a difference in the final policy generated we went into the algorithm and replaced the randomized initial policy with one where the policy was for any state go up. The final policy generated is the exact same as the one shown in Table 1. We also conducted the same experiment with other initial policy states and the outcome was always the same thus I do not believe the initial policy plays a factor into the final generated policy.

Results of running the MDP_policy_iteration function with k = 100 produced what is shown in Table 6 and running the MDP_policy_iteration function with k = 1 produces what is shown in Table 7. As can be seen and as is expected with a greater number of iterations in the policy evaluation we get a better final generated policy (one closer to the optimal policy).

6. Critique

Since this assignment was quite similar to the previous we were able to not only get started easier, but hack up and answer quicker as well. This assignment did a great job at not only introducing us to policy iteration, but helping us obtain a greater understanding of how it actually works. By having access to the algorithm both in the book, and through lectures made it incredibly easy to take it step by step and implement it in MATLAB without too many issues.

7. Log

Author Matthew Lemon & Derek Heldt-Werle
Coding Portion (Worked together): 4
Report (Derek): 2
Report (Matt): 2