Assignment 4: Reinforcement Learning

Mhanley6

MDP Overview:

I tried to be super unique, so I chose Gridworld from Sutton Chapter 4 and Forest Management for my Markov Decision Processes.

In the exciting world of Gridworld, the agent tries to reach either the top left corner of the map or the bottom right corner of the map. I liked this MDP because I too wish that life presented two obvious competing goals, instead of one boring goal state like in Frozen Lake. I looked at many different sizes of Gridworld ranging from 5x5s all the way up to 50x50 monsters. I settled on 20x20 for the bulk of my investigations. In gridworld, it is impossible to navigate outside of the borders of the world, instead the agent stays in place. It's literally thinking inside the box, so it is very representational of the corporate world.

Forest Management is more of a real world scenario, instead of the standard grid-world. Two actions can be taken to the forest map: Wait and do nothing, or Cut. The chance of a fire in the world is represented by a probability. Habitat suitability for the population we want to conserve has rewards that increase by .2 every year until 5 years without a fire, peaking at 1, and then declining by .1 every year until hitting a terminal reward state in year 10 of .5.

RL Algorithms

Each iteration of Policy Iteration requires policy evaluation, which itself may require multiple steps through state space. For PI, I consider it converged once there are 0 changes to the previous policy.

Value Iteration is a special case of PI that terminates after one pass of policy evaluation. VI technically only converges in the limit, but in practice we consider it converged below a certain tolerance level. I used a convergence tolerance threshold of 0.0001. This means that if the current value function didn't improve by at least 0.0001 from the previous iteration, I consider it converged and terminate.

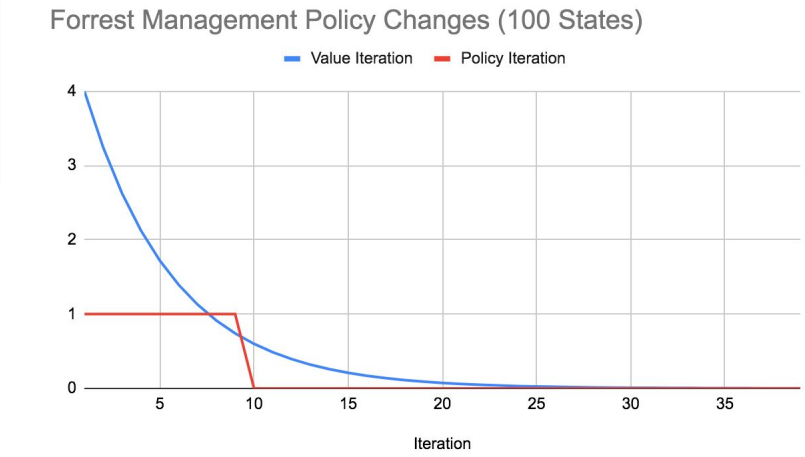
One drawback of PI / VI is that they are computed through dynamic programming style tables, which does not lend itself well to exact solutions on environments with continuous states and actions.

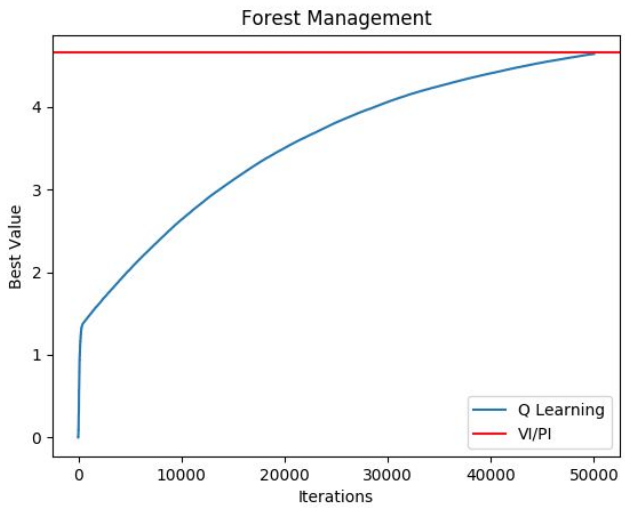
For my last RL algorithm, I chose Q learning. QL is similar to VI, except it is model-free and doesn't need to know the transition probabilities or rewards upfront. If we already knew those, VI would be much more efficient, but in general, Q learning is more practical because you often wouldn't have perfect information.

Forest Management (100 States)

	PI	VI	QL
Running Time	8.09 ms	4.54 ms	393 ms
Iterations	10	39	

Policy iteration converged in only 10 iterations, while taking VI 39 iterations. Despite that, VI still ran in only 56% of the time as PI. Q-learning took much longer and many more iterations than both of the model based learners, as expected.

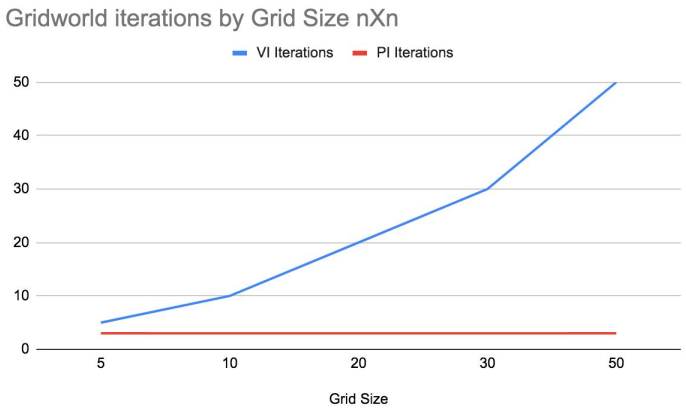


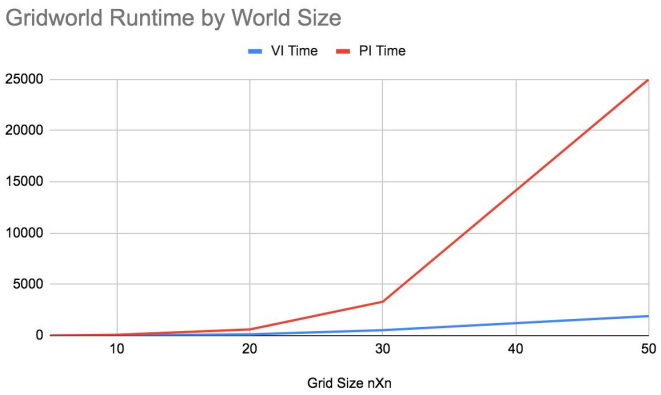


Gridworld

	PI	VI	QL
Running Time	590 ms	116 ms	9976 ms
Iterations	3	20	

For the more complex MDP, the speed of Value Iteration is even more apparent, running in 20% of the time as PI. In the simpler MDP, VI ran at 56% of the time as PI. This is despite the fact that VI takes a lot more iterations than PI. In fact, even as the size of the world grows, PI still converges in the same number of iterations, but it just takes increasingly longer to evaluate the policy. VI's runtime meanwhile grows in a much more manageable manner.





Optimal policies for VI/ PI are identical.

VI Policy (0=up, 1=right, 2=down, 3=left):

```
[[0 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 2]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 2]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 2]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 2]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 2]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 2]
 [0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 2]
 [0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 2]
 [0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 2]
 [0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 2]
 [0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 2]
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 2]
 [0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 2]
 [0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2]
 [0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2]
 [0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2]
 [0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2]
 [0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]]
```

PI Policy (0=up, 1=right, 2=down, 3=left):

```
[[0 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 2]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 2]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 2]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 2]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 2]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 2]
 [0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 2]
 [0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 2]
 [0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 2]
 [0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 2]
 [0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 2]
 [0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 2]
 [0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 2]
 [0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2]
 [0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2]
 [0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2]
 [0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2]
 [0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]]
```