Reinforcement Learning Algorithms

Trevor Grabham
CMPT 310
Spring 2021

Value Iteration

Policy - `[3 3 3 1 0 0 3 1 0 3 3 0 3 3 3 0 0]`
Value Function - `[88.99681008 91.63600829 94.29053752 96.44331847`
`86.98123484 89.75895848 96.6938905  99.19277736`
`78.11229865 94.87777492 98.97432773  0.`
`83.86809416 86.91004242 96.98621272 99.22489977`
`0.        ]`
Iterations - `22`

Policy Iteration

Policy - `[3 3 3 1 0 0 3 1 0 3 3 0 3 3 3 0 0]`
Value Function - `[88.99681008 91.63600829 94.29053752 96.44331847`
`86.98123484 89.75895848 96.6938905  99.19277736`
`78.11229865 94.87777492 98.97432773  0.`
`83.86809416 86.91004242 96.98621272 99.22489977`
`0.        ]`
Iterations - `5`

Results

The policy and value function results match perfectly between the value iteration, and policy iteration functions. This is to be expected as they both use very similar algorithms for computing the policy and value functions. The only difference between the two is that the policy iteration algorithm takes less iterations to compute the optimal policy and value functions. This is due to the fact that we are actually iterating over the value function during the policy evaluation phase in the policy iteration algorithm. This greatly reduces the number of iterations needed, because we are actually doing all of the converging in the policy evaluation phase and this doesn't show up in the policy iteration, iteration count. If we took this into account, we would probably see an interaction count that is much more similar between the two, likely an increase in iteration count for policy iteration, but the iteration loops would be more simple to calculate for policy iteration. We see the effects of this in the runtimes of the two algorithms, value iteration running in 1.696s and policy iteration running in 0.881s.