CS 7641 Machine Learning
Assignment 4

Philip Bale
pbale3

Due Sunday April 22nd, 2018 11:59pm

Introduction

This assignment explores reinforcement learning. It begins by choosing two Markov Decision Procceses (MDPs). Both MDPs are solved using both value iteration and policy iteration, and the results are compared against each other. Afterwards, Q-learning is applied to both problems and those results are then compared to the original results.

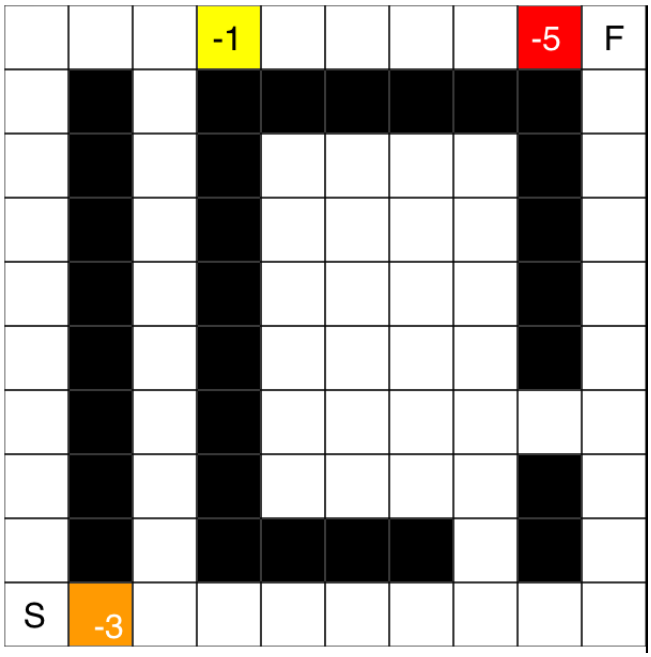
Problems Chosen

Gridworld path finding was chosen as the basis for the two Markov Decision Processes. This was done for a few different reasons. First, the ease in visual representation and understanding. Second, path finding is a common problem in real-life that has applications from walking to work, to driving cross-country, etc. Third, it is easy to scale the complexity of a grid as well as increase in the number of states. Fourth, for a solveable grid, one or more optimal solutions must exist. All-in-all, gridworld is great for demonstration reinforcement learning.

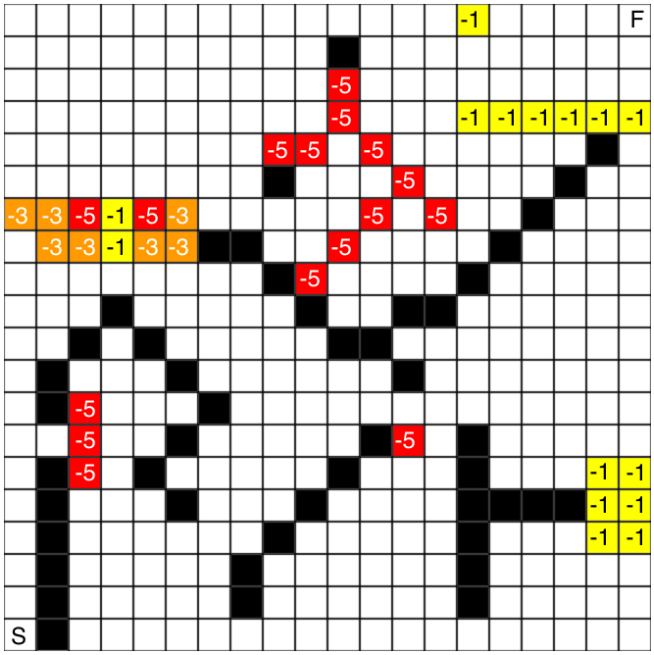
The first gridworld problem is an "easy" configuration. It has a 10x10 shape with 70 possible states.

The second gridworld problem is a "hard" configuration. It has a 20x20 shape with 353 possible states.

Each gridworld has its start state is in the bottom left corner and its end state is in the top right corner. Various "walls" exist that force the solution to optimize around. Various states have negative rewards assosciated with them and are color-coated in the state graphs below.



10x10 Easy Gridworld w/ 70 States

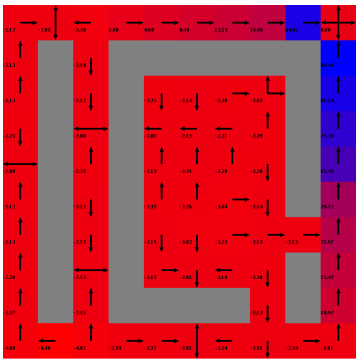


20x20 Hard Gridworld w/ 353 States

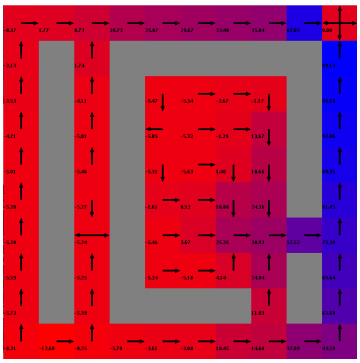
Value Iteration

Introduction

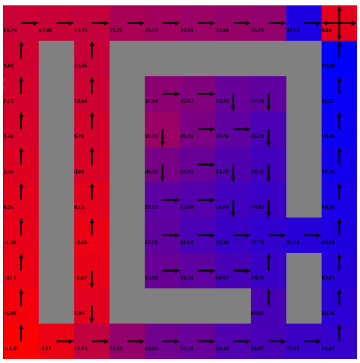
The first reinforcement learning algorithm used to optimally solve the MDPs is Value Iteration. Value iteration works by using the Bellman equation while moving from state to state in an attempt to reach convergence. Since the utility is not initially known, the algorithm begins with the reward at the final state and works backwards calculating utility for nearest states. This continues until all states are evaluated. After a number of iterations, the algorithm will eventually converge on an optimal solution.



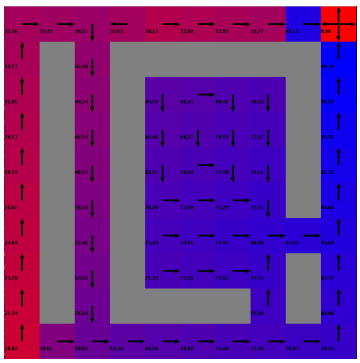
Easy GW Value Iteration #2



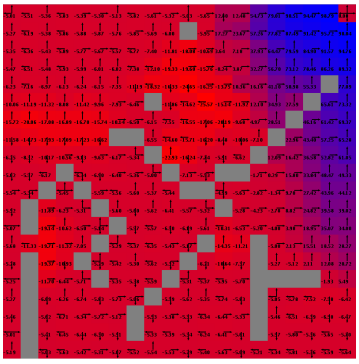
Easy GW Value Iteration #5



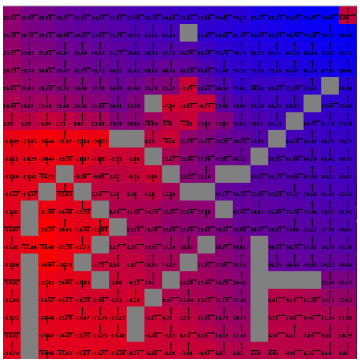
Easy GW Value Iteration #10



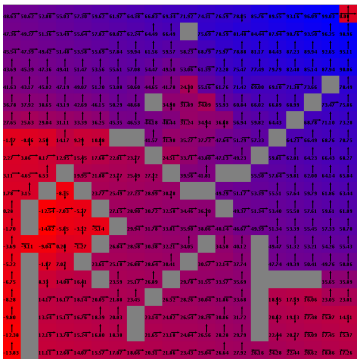
Easy GW Value Iteration #64



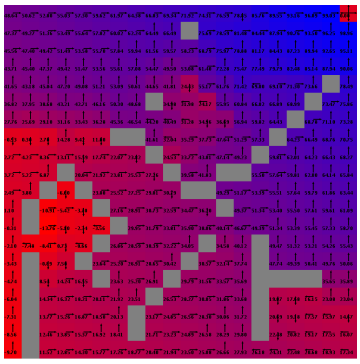
Hard GW Value Iteration #5



Hard GW Value Iteration #15



Hard GW Value Iteration #30

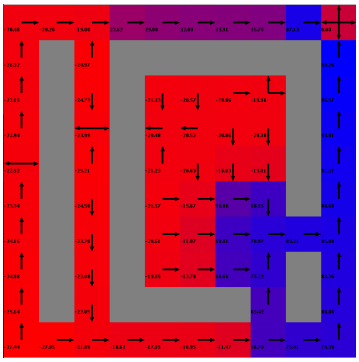


Hard GW Value Iteration #61

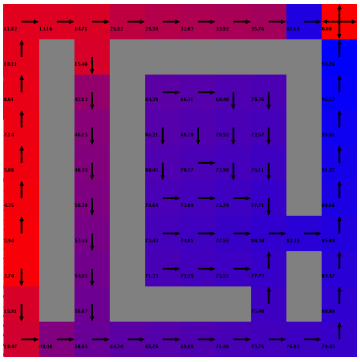
Policy Iteration

Introduction

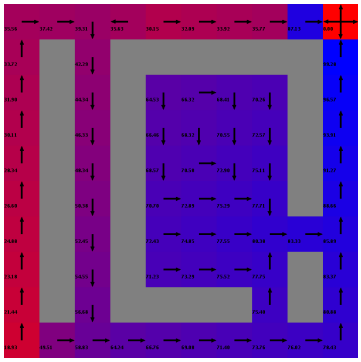
The second reinforcement learning algorithm used to optimally solve the MDPs is Policy Iteration. Policy iteration works by beginning with a random policy and then attemping to modify it state by state by taking different actions. This continues throughout each state of the policy in an attempt to find a better solution. When no further policy changes are found, the algorithm is finished. Policy Iteration tends to take longer than Value Iteration due to the additional number of explorations and calculations being done.



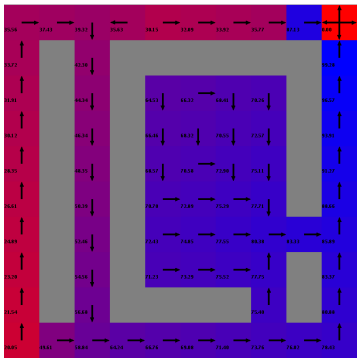
Easy GW Policy Iteration #2



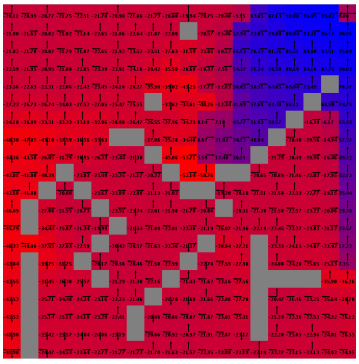
Easy GW Policy Iteration #5



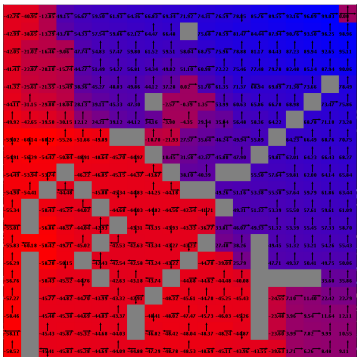
Easy GW Policy Iteration #10



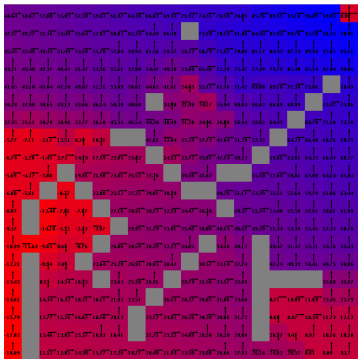
Easy GW Policy Iteration #33



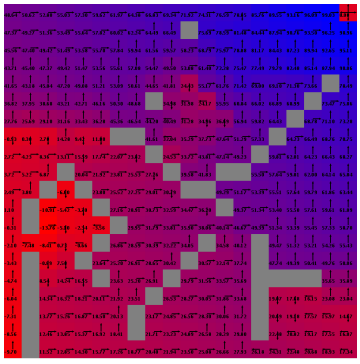
Hard GW Policy Iteration #2



Hard GW Policy Iteration #5



Hard GW Policy Iteration #10



Hard GW Policy Iteration #22

Q-learning

Introduction

The third reinforcement algorithm used to optimally solve the MDPs is Q-learning. Q-learning works by assigning q values to every state. At each state, the learner calculates new q values based on both immediate and future rewards. Initially, Q-learning will spend time exploring and learning, whereas it will eventually optimize based on knowledge learned and converge. While the first two algorithms have domain knowledge about the problem, Q-learning has no such information and learns as it goes.

Q-learning Params	Iterations	Time	Reward	Steps	Convergence
EASY Gridworld					
L0.1 q0.0 E0.1	158.0000	0.0896	50.4200	47.9200	5.3800
L0.9 q100.0 E0.3	294.0000	0.0571	49.9600	48.4200	21.9495
L0.9 q0.0 E0.3	156.0000	0.0416	48.4000	47.4000	53.3545
L0.9 q0.0 E0.5	19.0000	0.0093	45.9200	50.9800	63.1121
L0.9 q100.0 E0.5	615.0000	0.1257	44.5200	47.1200	25.2423
L0.9 q100.0 E0.1	372.0000	0.0806	44.0600	48.4800	22.5748
L0.9 q0.0 E0.1	902.0000	0.1461	43.7200	25.2200	26.0067
L0.9 q-100.0 E0.5	880.0000	0.1724	43.6400	25.3000	26.0176
L0.1 q100.0 E0.1	210.0000	0.0940	42.3800	52.1000	0.6710
L0.1 q0.0 E0.3	936.0000	0.1276	41.8000	25.2200	3.5595
HARD Gridworld					
L0.1 q0.0 E0.1	2900.0000	0.8542	19.6100	63.7700	2.1298
L0.1 q100.0 E0.5	1882.0000	0.8056	19.5100	63.8700	1.3662
L0.1 q0.0 E0.3	2411.0000	0.8169	19.0300	64.1900	2.9960
L0.1 q0.0 E0.5	1177.0000	0.5065	18.9700	62.9800	3.6562
L0.1 q100.0 E0.3	2958.0000	1.0309	18.4100	64.9200	1.8214
L0.1 q100.0 E0.1	2685.0000	0.9719	17.8000	63.1300	1.7738
L0.9 q0.0 E0.3	1728.0000	0.9165	5.7800	70.0500	64.5125
L0.9 q0.0 E0.1	754.0000	0.4340	-15.2900	63.1400	46.8360
L0.9 q100.0 E0.5	1891.0000	0.9692	-17.2500	89.3400	40.4389
L0.9 q100.0 E0.1	2791.0000	1.4653	-19.1800	62.6200	42.1840

Gridworld Q-learning results for various parameter configurations, sorted by max reward