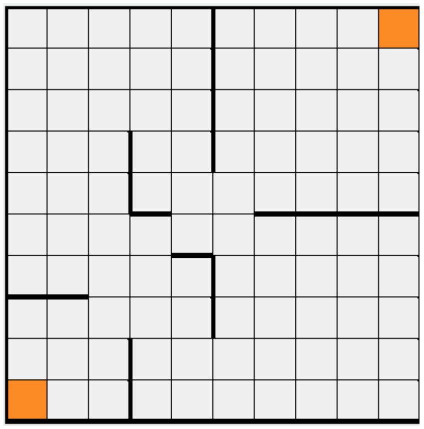
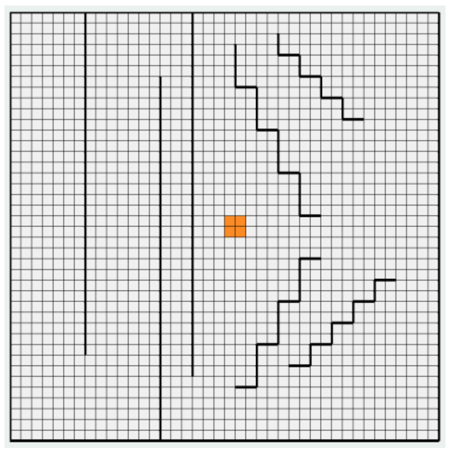
Pranshav Thakkar

GT username: pthakkar7

Markov Decision Processes

For this assignment, we were tasked with coming up with two interesting MDPs and solving each MDP using value iteration, policy iteration and a reinforcement learning algorithm of our choice, for which I decided to use Q-learning. My MDPs are both GridWorld problems, with the "small" one being a 10x10 grid and the "large" one being a 40x40 grid. The walls for the small gird had a penalty of 50 and the walls for the large grid had a penalty of 100.



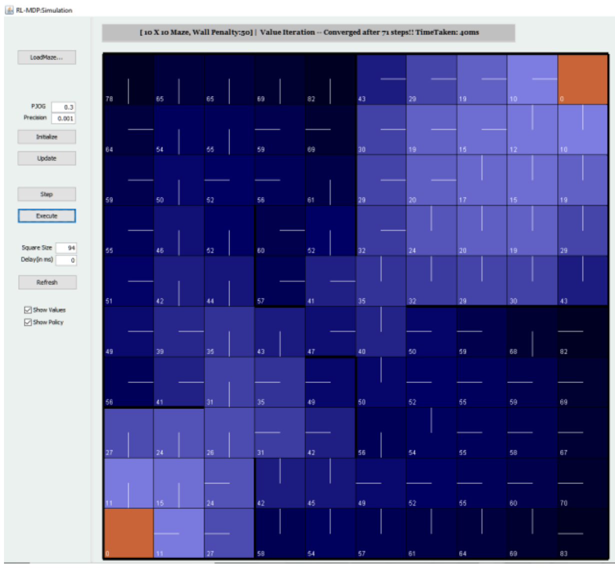


I believe the small grid to be an interesting problem because I thought it would be intriguing to see what decision would be made in terms of going towards the goal with the less compact area, but with more walls around it, or to go towards the goal that is more compact but has a less number of walls in total. The large grid problem is particularly interesting to me because I tried to make the problem similar to an arena that my robot would have to navigate through in one of my other classes. I wanted to see what kind of decisions would be made in a situation where there is a clear path on the left side of the problem versus the right side of the problem where there are "curved" obstacles. The walls had a higher penaly for this problem to emphasize the fact that we don't want the robot to hit the walls.

Value Iteration vs. Policy Iteration

Small Grid

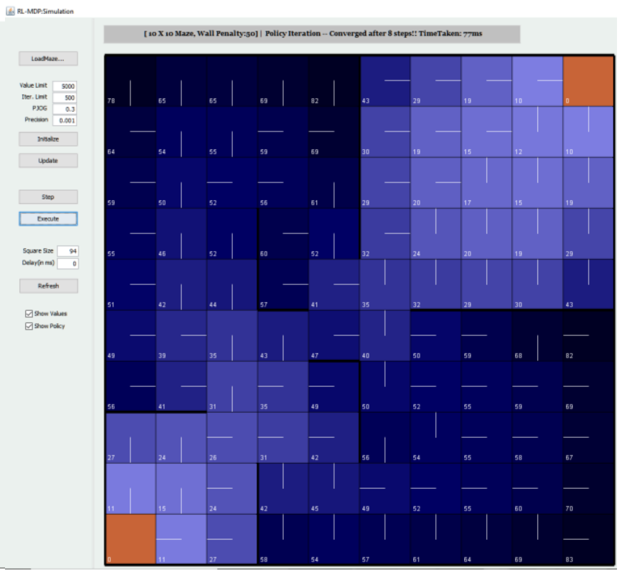
PJOG: 0.3



[10 X 10 Maze, Wall Penalty:50] | Value Iteration -- Converged after 71 steps!! TimeTaken: 40ms

PJOG 0.3
Precision 0.001
Square Size 94
Delay(in ms) 0
☑ Show Values
☑ Show Policy

78	65	65	69	82	43	29	19	10	0
64	54	55	59	69	30	19	15	12	10
59	50	52	56	61	29	20	17	15	19
55	46	52	60	52	32	24	20	19	29
51	42	44	57	41	35	32	29	30	43
49	39	35	43	47	40	50	59	68	82
56	41	31	35	49	50	52	55	59	69
27	24	26	31	42	56	54	55	58	67
11	15	24	42	45	49	52	55	60	70
0	11	27	58	54	57	61	64	69	83



[10 X 10 Maze, Wall Penalty:50] | Policy Iteration -- Converged after 8 steps!! TimeTaken: 77ms

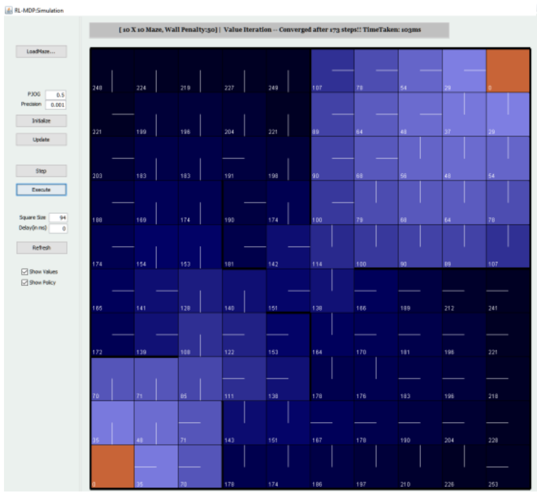
Value Limit 5000
Iter. Limit 500
PJOG 0.3
Precision 0.001
Square Size 94
Delay(in ms) 0
☑ Show Values
☑ Show Policy

78	65	65	69	82	43	29	19	10	0
64	54	55	59	69	30	19	15	12	10
59	50	52	56	61	29	20	17	15	19
55	46	52	60	52	32	24	20	19	29
51	42	44	57	41	35	32	29	30	43
49	39	35	43	47	40	50	59	68	82
56	41	31	35	49	50	52	55	59	69
27	24	26	31	42	56	54	55	58	67
11	15	24	42	45	49	52	55	60	70
0	11	27	58	54	57	61	64	69	83

With the small grid, Value Iteration and Policy Iteration returned the same solutions, but had differences in the number of iterations and the time it took. Value Iteration took 71 steps to converge while Policy Iteration only took 8 steps to converge. However, Value Iteration took 40ms to converge which is about half of what it took Policy Iteration to converge (77ms).

It seems that with the small grid size, it is faster to just go through Value Iteration to find the optimal solution, with it going through 71 steps in 40 ms, than to try and update a random policy, which took less steps (8) but required more time overall (77ms).

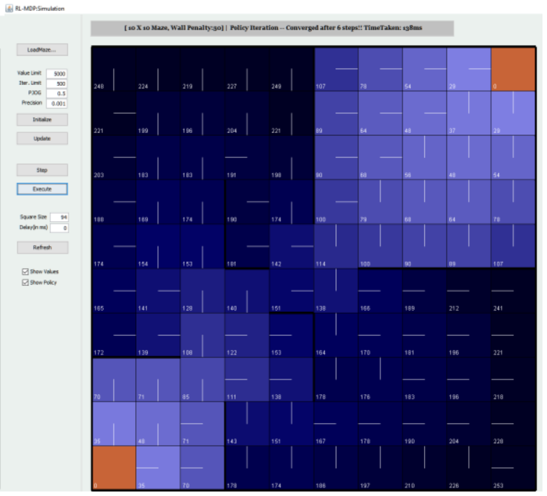
PJOG: 0.5



[10 X 10 Maze, Wall Penalty:50] | Value Iteration -- Converged after 173 steps!! TimeTaken: 103ms

PJOG 0.5
Precision 0.001
Square Size 94
Delay(in ms) 0
☑ Show Values
☑ Show Policy

248	224	219	227	249	107	78	54	29	0
221	199	196	204	221	89	64	48	37	29
203	183	183	191	198	90	68	56	48	54
188	169	174	190	174	100	79	68	64	78
174	154	153	181	142	114	100	90	89	107
165	141	128	140	151	138	166	189	212	241
172	139	108	122	153	164	170	181	196	221
70	71	85	111	138	178	176	183	196	218
35	48	71	143	151	167	178	190	204	228
0	35	70	178	174	186	197	210	226	253



[10 X 10 Maze, Wall Penalty:50] | Policy Iteration -- Converged after 6 steps!! TimeTaken: 138ms

Value Limit 5000
Iter. Limit 500
PJOG 0.5
Precision 0.001
Square Size 94
Delay(in ms) 0
☑ Show Values
☑ Show Policy

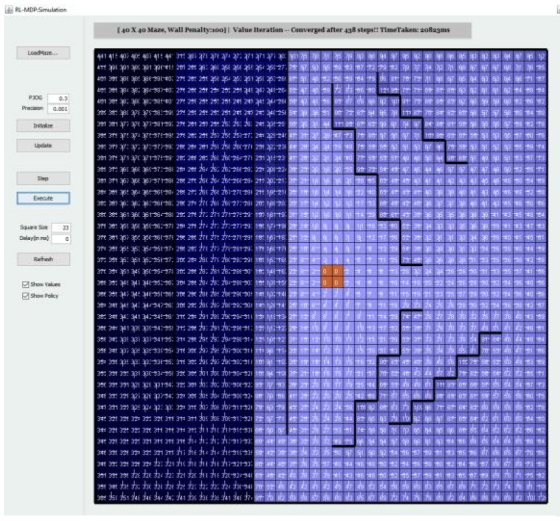
248	224	219	227	249	107	78	54	29	0
221	199	196	204	221	89	64	48	37	29
203	183	183	191	198	90	68	56	48	54
188	169	174	190	174	100	79	68	64	78
174	154	153	181	142	114	100	90	89	107
165	141	128	140	151	138	166	189	212	241
172	139	108	122	153	164	170	181	196	221
70	71	85	111	138	178	176	183	196	218
35	48	71	143	151	167	178	190	204	228
0	35	70	178	174	186	197	210	226	253

Increasing the PJOG value (which relates to the probability on which actions are taken in the environment) to 0.5 changed the reward values for each position slightly but the two solutions are still the same. The number of steps for Value Iteration went up from 71 to 173, however, they went down from 8 steps to 6 steps for Policy Iteration. Time increased for both algorithms, with VI going from 40 ms to 103ms and PI going from 77ms to 138ms.

Based on this, it seems like increasing the PJOG value will eventually favor Policy Iteration as it will become harder to find exact values that converge when the probabilities to determine actions get closer to 50-50.

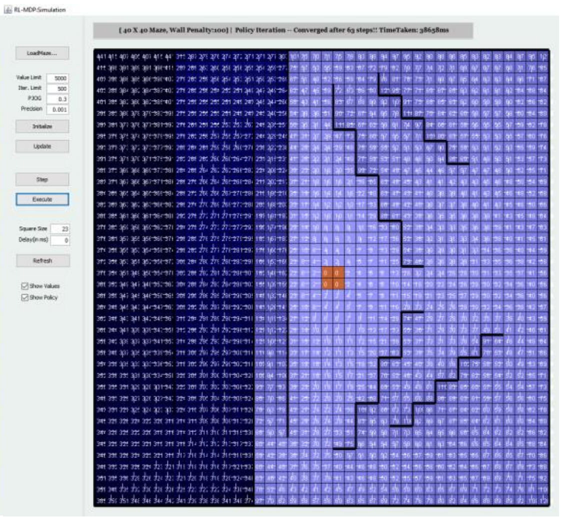
<u>Large Grid</u>

PJOG: 0.3


RL-MDP:Simulation

[40 X 40 Maze, Wall Penalty:100] | Value Iteration -- Converged after 438 steps!! TimeTaken: 20823ms

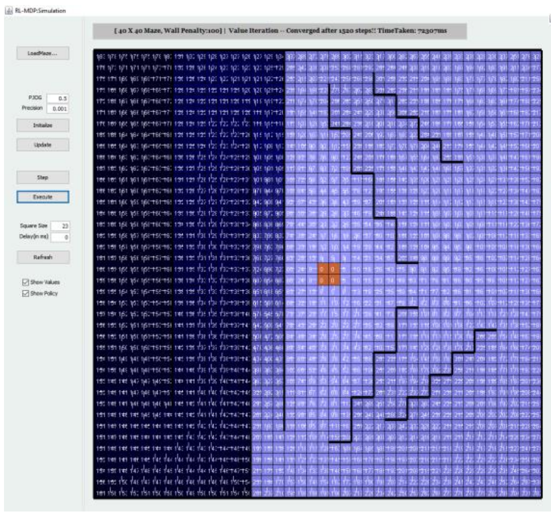
PJOG 0.3
Precision 0.001
Square Size 23
Delay(in ms) 0


RL-MDP:Simulation

[40 X 40 Maze, Wall Penalty:100] | Policy Iteration -- Converged after 63 steps!! TimeTaken: 38658ms

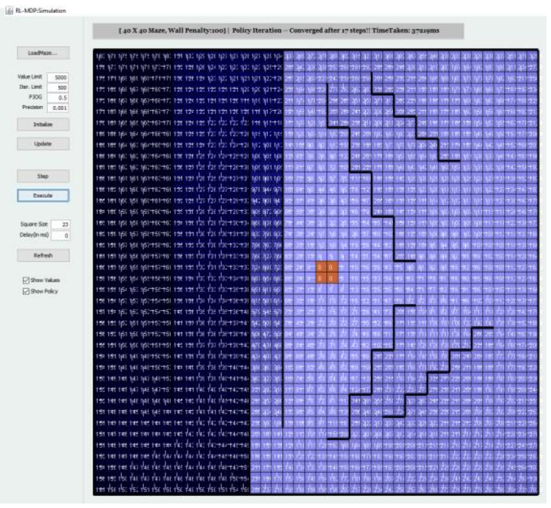
Value Limit 5000
Iter. Limit 500
PJOG 0.3
Precision 0.001
Square Size 23
Delay(in ms) 0

PJOG: 0.5


RL-MDP:Simulation

[40 X 40 Maze, Wall Penalty:100] | Value Iteration -- Converged after 1520 steps!! TimeTaken: 72307ms

PJOG 0.5
Precision 0.001
Square Size 23
Delay(in ms) 0


RL-MDP:Simulation

[40 X 40 Maze, Wall Penalty:100] | Policy Iteration -- Converged after 17 steps!! TimeTaken: 37219ms

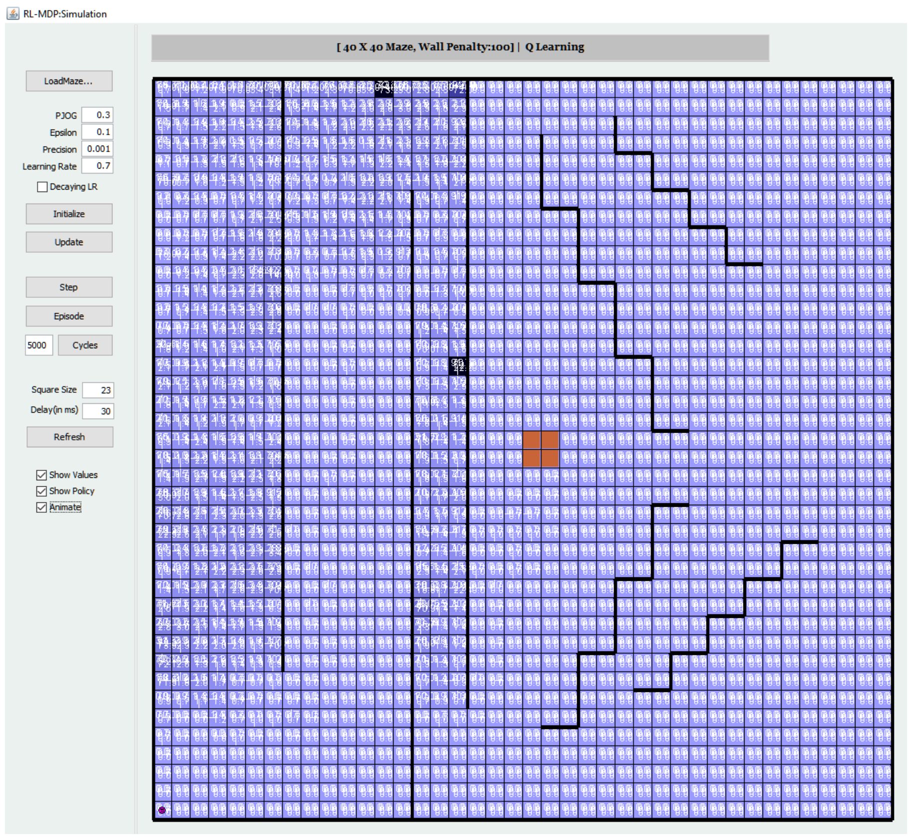
Value Limit 5000
Iter. Limit 500
PJOG 0.5
Precision 0.001
Square Size 23
Delay(in ms) 0

For the larger grid, both algorithms had the same optimal solutions again, and similar to the smaller grid, it took Policy Iteration (38658ms) almost double the time as Value Iteration (20823ms). It still took Value Iteration (438 steps) a lot more steps than Policy Iteration (63 steps). So, even with the larger grid, it is still more time efficient to go through Value Iteration than Policy Iteration.

However, when the PJOG is changed to 0.5, things get interesting. It takes Value Iteration 1520 steps which is more then 3 times as much as when PJOG was 0.3 and it took 72307ms, which is also more than 3 times as long as it took for PJOG = 0.3. Policy Iteration is a completely different story, taking only 17 steps in 37219ms. Both of those values are smaller than when PJOG was 0.3. Thus, my intuition proved to be somewhat correct with Policy Iteration performing much better than Value Iteration when the PJOG was moved closer to 0.5. The larger grid showed us the difference significantly better than the smaller grid.

Again, I believe this is because it becomes easier to take a random policy and attempt to update it to make it work when the grids get larger with the probabilities of the actions getting close to 50-50. It is just more efficient to do this than to go through and assign a value for each and every box, especially when the grids are larger.

Q-Learning


RL-MDP:Simulation

[40 X 40 Maze, Wall Penalty:100] | Q Learning

PJOG 0.3
Epsilon 0.1
Precision 0.001
Learning Rate 0.7
☐ Decaying LR
5000 Cycles
Square Size 23
Delay(in ms) 30
☑ Show Values
☑ Show Policy
☑ Animate

I set the PJOG to 0.3, and the learning rate to 0.7 for this Q-learning situation. I set the number of cycles for each episode to be 5000. This algorithm started in the bottom left corner and it took over 3 minutes for it to find its way to a goal state. It explored going straight up and then made its way back down and around the corner. Similarly, it took a while for the algorithm to make its way through the narrow path but didn't take long to reach the goal after that.

I believe that it took the q-learner this long because of the specific layout of the grid and because it started in the bottom left corner. It takes longer because it doesn't know the rewards and has to go figure that out by itself. It has to explore and determine where to go.

I attempted to explore using the same q-learner, except with a decaying learning rate. However, that version was not close to converging even after it had been running for 10 minutes. I assume that the difficult situation was too tough to overcome with the learning rate declining and no clear path available to the goal.

It definitely took Q-Learning longer and more steps to converge (in the scenario in which it did converge) than either Value Iteration or Policy Iteration. In this case, I believe it was definitely due to the specific grid world problem that was attempted to be solved. I believe that the environment is the deciding factor in general for which algorithm will work better. In this case, the environment of the problem is the most important piece of domain knowledge. For example, a grid world problem in which there are not many obstacles could be solved faster by a Q-Learner, where it explores and could find a path to the goal. VI and PI may take longer in that situation since they have to find a policy for every box in the grid.