Wroclaw University of Science and Technology

Artificial Intelligence and Machine Learning

Politechnika
Wrocławska

Markov Decision Problems and Reinforcement Learning Project

Ines de Oliveira Soares, 256652

Part I - Markov Decision Problems

In this part of the project, to solve the MDP problems, I developed the Value Iteration Algorithm.

Question 1 - Gridworld 4x3

Firstly, to see if the developed algorithm is working correctly, I runned the program in the gridworld 4x3 like in the lecture class and compared the results. With discount factor of 0.99, the reward of the winning state being +1, a losing state with -1 as a reward and the rest of the states with reward -0.04. The motion uncertainty model being 0.8, 0.1, 0.1 and the condition to stop computing the utilities is when the reduction of all the differences between successive iterations is below 0.0001.

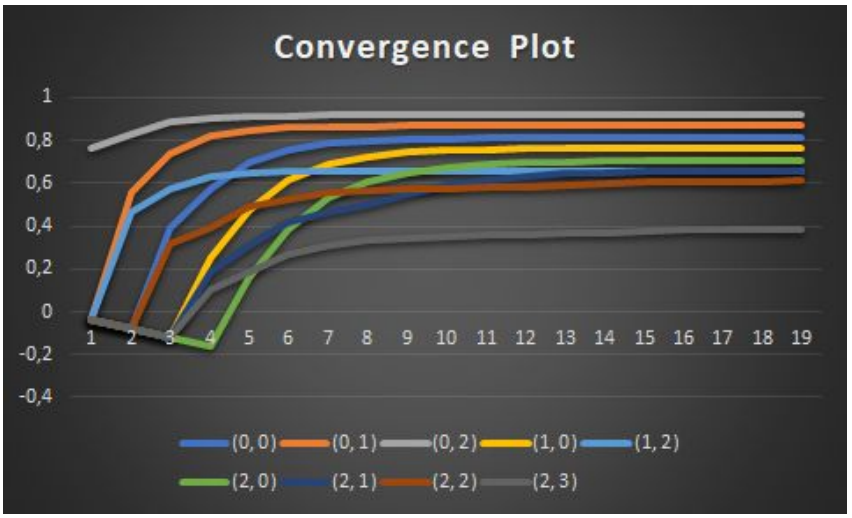
In the picture below are represented the final utilities for each state. As we can conclude, the values obtained are the same as in the slides.

0.8115	0.8678	0.9178	1.0
0.7615	0.0	0.6603	-1.0
0.7052	0.6552	0.6111	0.3872

In the picture below is represented the optimal policy in each state, which gives the expected results. To reach these results, the algorithm did 19 iterations.

>	>	>	1
^	F	^	-1
^	<	<	<

In the picture below is shown the convergence plot graph. As expected, the utility values of the states start being zero and then with the increase of the number of iterations they start updating the values, until they reach a point where they stay stable.



Question 2 - Gridworld 4x4

After checking that the developed program is working correctly, it was time to implement it on a different gridworld. Now it is a 4 by 4 world, with discount factor of 0.99, the reward of the winning state being +100, a special state with +20 as a reward and the rest of the state with reward -1. The motion uncertainty model stays the same (0.8, 0.1, 0.1). And the condition to stop computing the utilities is when the reduction of all the differences between successive iterations is below 0.0001.

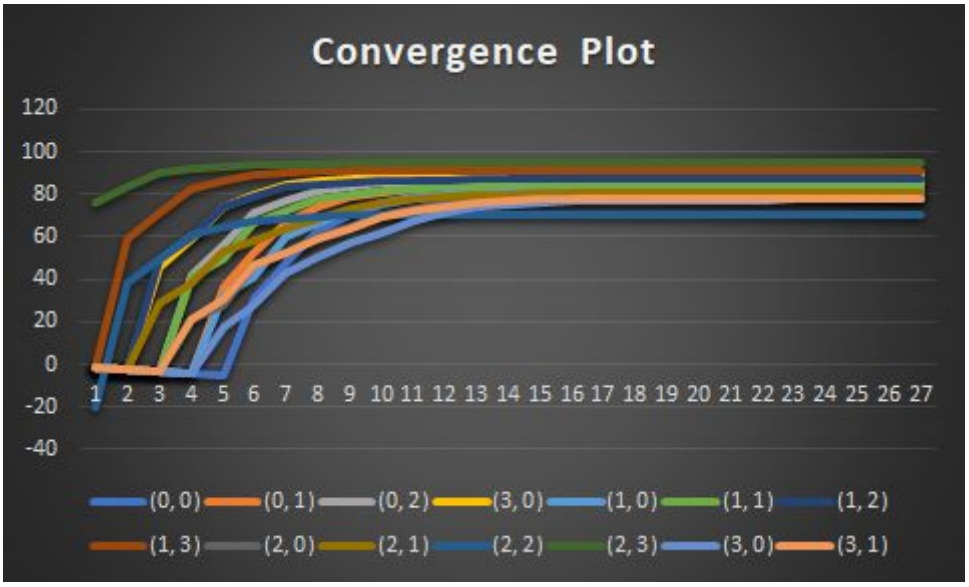
As a result we get the utilities shown in the picture below.

81.9382	84.2609	86.586	88.8827
81.7351	84.2723	87.0596	91.5547
79.5929	80.5995	70.467	94.5352
77.4515	78.2487	0.0	100.0

In the picture below is shown the full policy obtained. To reach these results, the algorithm did 27 iterations.

>	>	>	v
>	>	>	v
^	^	>	v
^	^	F	100

In the picture below is shown the convergence plot graph.



Question 3

In this part, the parameters are the same as in the previous question with the exception that the reward of the special state is -70.

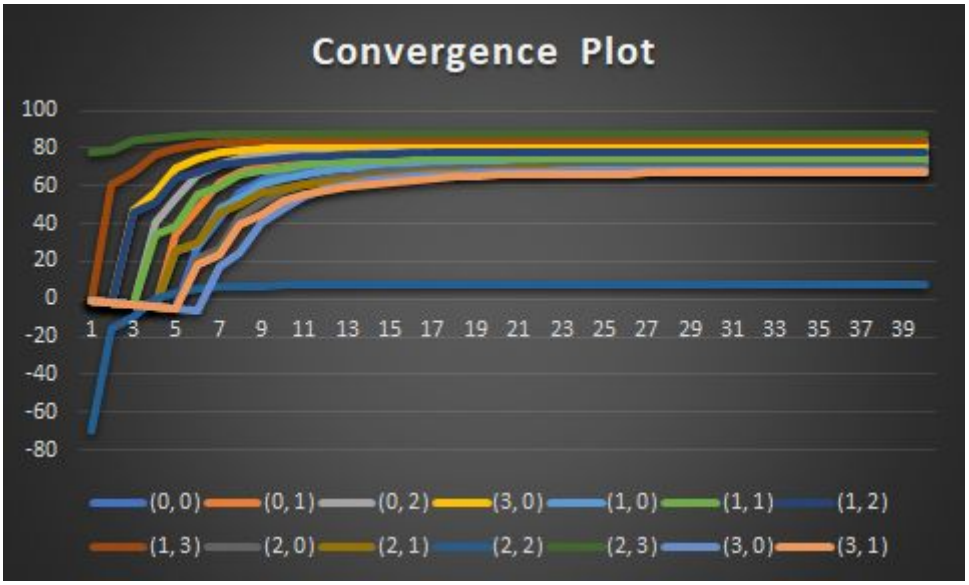
The picture below shows the resulting utilities for each state.

74.4792	76.908	79.39	81.8716
72.6736	74.9222	77.6575	84.4782
70.3226	68.7231	7.8967	87.6601
68.0419	66.7749	0.0	100.0

In the picture below is shown the full policy obtained. To reach these results, the algorithm did 40 iterations.

>	>	>	v
>	>	^	v
^	<	>	v
^	^	F	100

In the picture below is shown the convergence plot graph.



As we can observe by assigning a very low reward to the special state (-70), it resulted in a notable change in the full policy and in the utility value of this state (it decreased a lot). For example in states (2,2) and (1,2), which are neighbors of the special state, the best action is to move in the opposite direction of the special state. This doesn't happen with the other neighbor of the special state (2,3) because it is also neighbor with the winning state which as a reward of +100, therefor the best action in this state is to go in the direction of the winning state, although there is a slight possibility (0.1) that it can go to the special state.

Question 4

In this part, the parameters are the same as in the question 2 with the exception that the motion uncertainty model which now has the values of 0.4, 0.3, 0.3. This change increases the possibility of not follow the action we want, which takes more in consideration if the neighbors states are good or not.

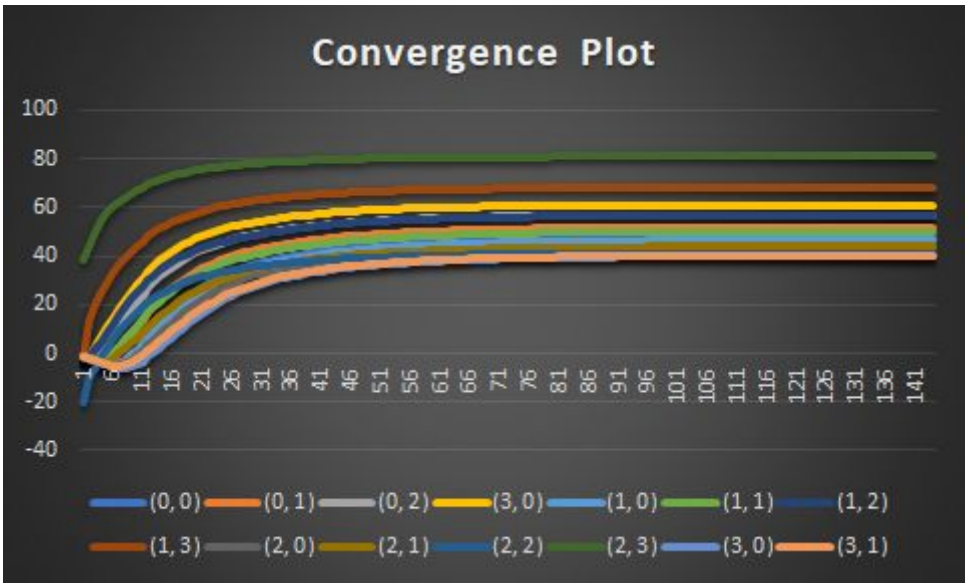
The picture below shows the resulting utilities for each state.

47.4309	51.8557	56.9415	61.0715
46.4981	50.1902	56.7213	68.2379
43.3489	43.9796	41.1801	81.0702
39.9984	40.2483	0.0	100.0

In the picture below is shown the full policy obtained. To reach these results, the algorithm did 144 iterations.

>	>	>	v
^	^	^	>
^	^	>	>
^	^	F	100

In the picture below is shown the convergence plot graph.



One notable change to the full policy comparing to the original one is the policy on state (2,3) where the agent prefers to go right because has a possibility of 0.3 of getting to the winning state and zero chance to get to the special state. If the agent would chose to go down, there would be a 0.4 chance to reach the winning state, but a 0.3 chance to get to the special state with a -20 reward.

It is also notable that the utilities values in all the states are considerably lower because there is no certainty that the agent will follow the action desired.

Question 5

In this part, the parameters are the same as in the question 2 with the exception that the discount factor now if 0.90.

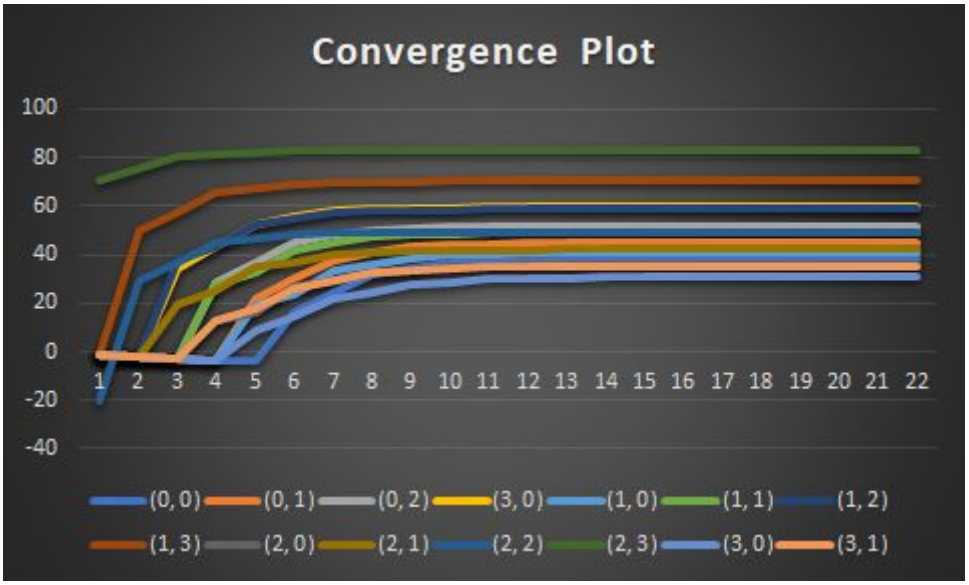
The picture below shows the resulting utilities for each state.

38.4388	44.8393	51.9197	59.6665
41.0582	49.1295	58.7454	70.3108
35.8408	42.1917	49.4317	82.9108
30.7519	35.3248	0.0	100.0

In the picture below is shown the full policy obtained. To reach these results, the algorithm did 22 iterations.

>	>	>	v
>	>	>	v
>	>	>	v
^	^	F	100

In the picture below is shown the convergence plot graph.



In this case, the full policy becomes more reckless. As we can see, the states (2,0) and (2,1) don’t prefer to avoid the special state with the -20 reward. In fact, the policy is to go in the direction of the special state, just to be closer to the winning state.

Part II - Reinforcement Learning

In the second part of the assignment, the objective was to compute the optimal agent's policy using the Q-learning method with exploration. The world used was the previous one (question 2), with the same basic parameters. But now, the motion uncertainty model and the rewards are unknown to the agent, they will only be used to generate the trials.
The parameters are the same as before: motion uncertainty model being 0.8, 0.1, 0.1 and a discount factor of 0.99. With a learning parameter α=1/N(s,a) where N(s,a) is the number of times the action 'a' has been selected in state 's' in previous trials.
The exploration strategy should select the optimal move (exploitation) with probability 1-ε, and a random move (exploration) with probability ε. For this assignment, I will use to values of exploration: ε=0.05 and ε=0.2.

Exploration: ε=0.05

In this part the value of exploration was 0.05, which means that with a probability of 0.05, the agent will take a random action instead of following the rules.

10000 Trials

In the picture below it is shown the results of the Q-values for 10000 trials with exploration of 0.05.

36.74	^	41.29	^	50.9	^	62.51	^
40.23	v	46.77	v	59.96	v	72.45	v
37.06	<	38.65	<	13.05	<	53.7	<
30.88	>	23.14	>	67.3	>	5.42	>
38.6	^	43.76	^	57.41	^	63.13	^
34.88	v	41.36	v	54.24	v	78.26	v
38.9	<	40.42	<	49.84	<	64.19	<
43.15	>	52.97	>	66.19	>	48.23	>
33.07	^	43.11	^	42.8	^	74.47	^
24.97	v	33.18	v	30.38	v	84.58	v
30.55	<	32.79	<	21.25	<	46.65	<
21.78	>	42.69	>	53.98	>	67.28	>
23.8	^	35.76	^	0	^	100	^
26.39	v	32.31	v	0	v	100	v
23.88	<	29.84	<	0	<	100	<
30.61	>	27.96	>	0	>	100	>

The picture below shows the resulting utilities for each state.

40.23	46.77	67.3	72.45
43.15	52.97	66.19	78.26
33.07	43.11	53.98	84.58
30.61	35.76	0	100

In the picture below is shown the full policy obtained.

v	v	>	v
>	>	>	v
^	^	>	v
>	^	F	100

20000 Trials

The results with 10000 trials were not what I was expecting so I increased the number of trials. In the picture below it is shown the results of the Q-values for 20000 trials with exploration of 0.05.

30.91	^	52.91	^	61.1	^	43.04	^
40.93	v	55.89	v	67.22	v	66.66	v
38.73	<	35.01	<	40.46	<	10.31	<
19.21	>	42.45	>	28.34	>	-0.59	>
35.95	^	49.56	^	60.53	^	62.32	^
43.84	v	52.03	v	59.8	v	81.96	v
40.54	<	42.48	<	54.59	<	69.7	<
40.44	>	65.07	>	72.36	>	40.94	>
41.87	^	50.68	^	50.57	^	78.61	^
37.13	v	45.7	v	42.32	v	87.04	v
41.48	<	43.94	<	34.85	<	65.92	<
45.48	>	53.47	>	60.01	>	83.26	>
36.87	^	45.98	^	0	^	100	^
36.26	v	42.57	v	0	v	100	v
36.29	<	24.22	<	0	<	100	<
35.16	>	41.85	>	0	>	100	>

The picture below shows the resulting utilities for each state.

40.93	55.89	67.22	66.66
43.84	65.07	72.36	81.96
45.48	53.47	60.01	87.04
36.87	45.98	0	100

In the picture below is shown the full policy obtained.

v	v	v	v
v	>	>	v
>	>	>	v
^	^	F	100

30000 Trials

In the picture below it is shown the results of the Q-values for 30000 trials with exploration of 0.05.

27.7 ^	34.73 ^	61.15 ^	72.12 ^
38.63 v	55.86 v	64.91 v	75.93 v
23.21 <	23.45 <	54.7 <	28.46 <
43.06 >	61.52 >	57.96 >	29.18 >
22.79 ^	54.89 ^	64.14 ^	69.16 ^
30.9 v	51.49 v	60.18 v	82.63 v
32.24 <	50.76 <	61.4 <	70.07 <
56.42 >	64.19 >	72.14 >	53.81 >
47.31 ^	56.65 ^	52.58 ^	78.61 ^
38.82 v	46.39 v	41.11 v	87.21 v
41.23 <	44.23 <	36.45 <	65.06 <
44.54 >	54.74 >	60.4 >	78.19 >
33.55 ^	47.94 ^	0 ^	100 ^
34.91 v	42.45 v	0 v	100 v
39.19 <	43.22 <	0 <	100 <
43.19 >	46.04 >	0 >	100 >

The picture below shows the resulting utilities for each state.

43.06	61.52	64.91	75.93
56.42	64.19	72.14	82.63
47.31	56.65	60.4	87.21
43.19	47.94	0	100

In the picture below is shown the full policy obtained.

>	>	v	v
>	>	>	v
^	^	>	v
>	^	F	100

Satisfactory Results

For more than 30000 trials the utilities didn't have a significant change, so I can say that the number of trials necessary to achieve satisfactory results in this case is 30000. The results are not quite equal to the expected ones, although they are not far from reaching them.

Exploration: ε=0.2

In this part the value of exploration was 0.2, which means that with a probability of 0.2, the agent will take a random action instead of following the rules.

10000 Trials

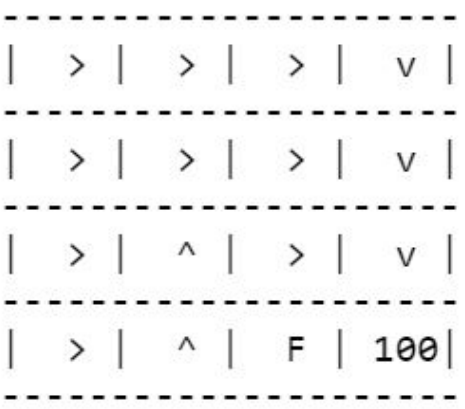
In the picture below it is shown the results of the Q-values for 10000 trials with exploration of 0.2.

47.99	^	54.77	^	61.9	^	69.45	^
45.16	v	56.72	v	63.39	v	73.13	v
47.82	<	49.54	<	56.6	<	66.0	<
50.88	>	56.91	>	68.32	>	70.25	>
46.85	^	55.13	^	64.26	^	69.58	^
41.31	v	50.93	v	56.6	v	79.48	v
45.49	<	50.4	<	57.28	<	68.87	<
54.46	>	61.58	>	70.57	>	74.85	>
41.91	^	53.48	^	48.92	^	74.29	^
36.39	v	41.8	v	36.68	v	85.53	v
41.56	<	44.19	<	32.65	<	59.79	<
46.39	>	51.5	>	55.69	>	84.65	>
34.58	^	45.2	^	0	^	100	^
35.88	v	41.59	v	0	v	100	v
35.87	<	38.41	<	0	<	100	<
39.79	>	42.32	>	0	>	100	>

The picture below shows the resulting utilities for each state.

50.88	56.91	68.32	73.13
54.46	61.58	70.57	79.48
46.39	53.48	55.69	85.53
39.79	45.2	0	100

In the picture below is shown the full policy obtained.



20000 Trials

In the picture below it is shown the results of the Q-values for 20000 trials with exploration of 0.2.

45.47	^	53.13	^	62.27	^	67.45	^
44.76	v	54.45	v	63.84	v	72.77	v
45.15	<	45.89	<	53.6	<	65.39	<
51.0	>	61.31	>	66.53	>	55.91	>
45.36	^	54.2	^	61.99	^	68.76	^
41.27	v	48.86	v	55.77	v	80.13	v
45.36	<	47.46	<	57.02	<	67.95	<
53.02	>	61.85	>	70.42	>	75.67	>
46.51	^	52.57	^	46.12	^	74.14	^
36.87	v	42.52	v	36.97	v	85.6	v
39.96	<	42.05	<	32.02	<	61.92	<
42.79	>	49.8	>	57.68	>	84.35	>
35.27	^	43.31	^	0	^	100	^
35.72	v	39.89	v	0	v	100	v
35.24	<	38.2	<	0	<	100	<
38.58	>	41.51	>	0	>	100	>

The picture below shows the resulting utilities for each state.

51.0	61.31	66.53	72.77
53.02	61.85	70.42	80.13
46.51	52.57	57.68	85.6
38.58	43.31	0	100

In the picture below is shown the full policy obtained.

>	>	>	v
>	>	>	v
^	^	>	v
>	^	F	100

Satisfactory Results

For more than 20000 trials the utilities didn't have a significant change, so I can say that the number of trials necessary to achieve satisfactory results in this case is 20000. The results are not quite equal to the expected ones, although they are not far from reaching them.

Conclusions

Comparing the results, we can conclude that with a higher rate of exploration, the agent will explore more the world. As we can see, with a exploration rate of 0.2, the Q-values converge faster to the final value. By taking more random actions (with probability of 0.2 instead of 0.05), the agent is able to discover more paths and then learn from them which ones are best, therefore the values converge faster.