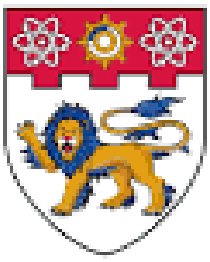


NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

Masters in science (Artificial Intelligence) (MSAI)

AI-610

Assignment 1

Name of Student:
Teo Lim Fong

Matriculation No: G2101964G

Contents

Q learning

Q learning is an off policy that updates its Q values by using the Q values of the next state and select the maximum action in that state. In off policy, the target policy is not equal to the behaviour policy. It evaluates and improve a different policy from that policy that is used for the action.

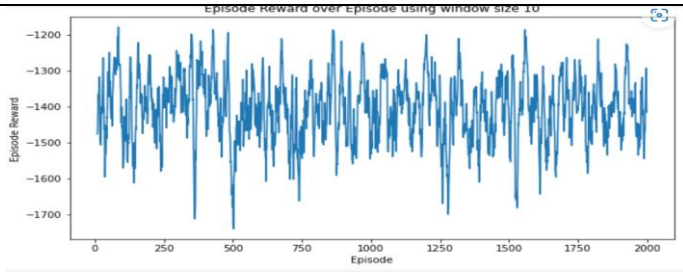
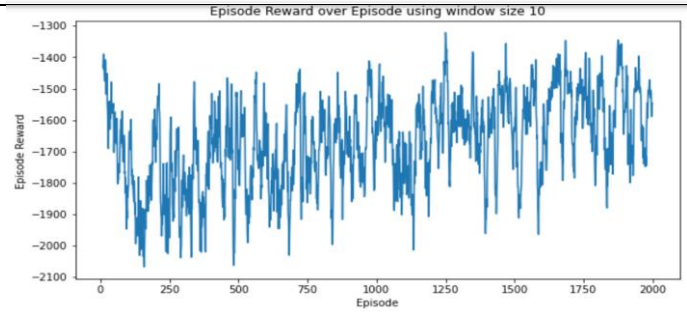
Nevertheless, we need to finetune the parameters such as the discounted value, the learning rate, and the epsilon. Different environments have different values of these parameters. For discounted value, gamma, it is normally the range of 0.9 to 0.99. For this assignment, I will assign the discounted value as 0.98 and experiment the other two parameters – learning rate and the epsilon.

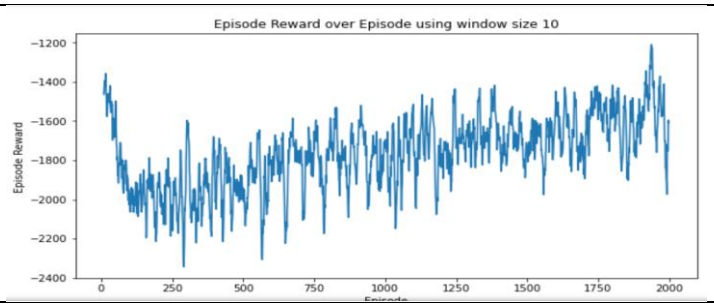
Learning rate can be defined as how fast you want the function to learn the slope. In deep learning, this learning rate is normally set to 0.01 – 0.001 because if the learning rate is too high, this mean it will cause overfitting and if the learning rate is too low it will cause underfitting. However, Q learning does not involve any deep learning. Therefore, we can try to set a range of value such as 0.9 and 0.5.

Epsilon is important in reinforcement learning. It is a trade-off between exploration and exploitation. Ideally, we need to explore the environment at the beginning which normally the epsilon will be high to choose random action then after a few hundred episodes the epsilon will decay and exploited by choosing the maximum action in that state. However, I want to experiment a static epsilon instead of decay epsilon to see how different epsilon performed. I will be using 0.5, 0.1 and 0.01.

Plotting of Episode rewards vs. Episodes with learning rate of 0.5

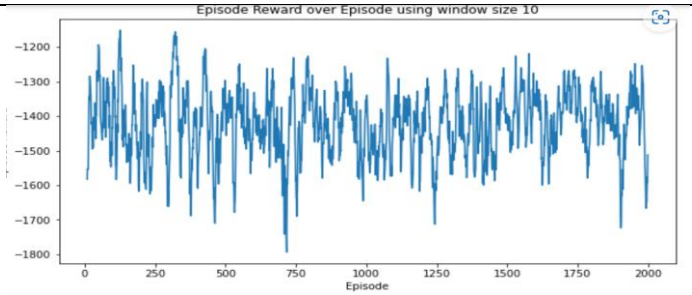
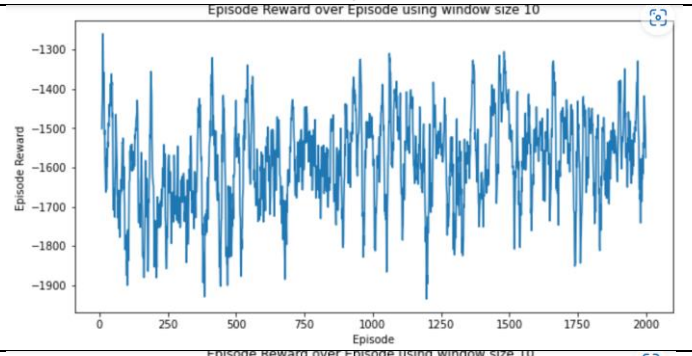
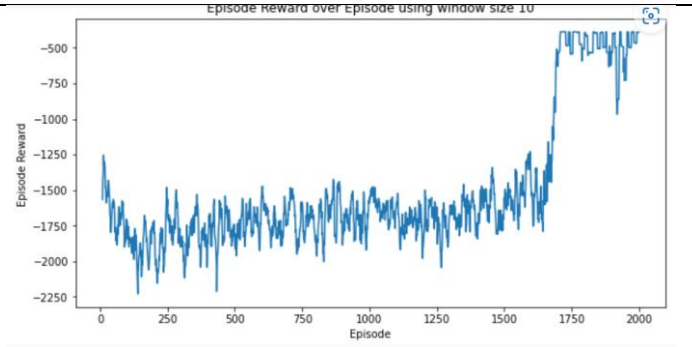
Table 1: Plotting Episode Reward over Episodes with learning rate of 0.5 and different epsilon values

	Epsilon	Episode Reward over Episodes	Push Box to Goal	Total Reward
Learning rate = 0.5	0.5		No	NIL
	0.1		No	NIL

	0.01		Yes	~ -1200

Plotting of Episode rewards vs. Episodes with learning rate of 0.9

Table 2: Plotting Episode Reward over Episodes with learning rate of 0.5 and different epsilon values

	Epsilon	Episode Reward over Episodes	Push Box to Goal	Total Reward
Learning rate = 0.9	0.5		No	NIL
	0.1		No	NIL
	0.01		Yes	-388

From both Table1 and 2, we can clearing see that epsilon 0.01 are able to make the agent to push the box to the desire state (Goal). However, I choose the Table 2 epsilon 0.01 (highlighted in red) because from the plot we can analysis that in Table 1 the highest total reward of epsilon 0.01 is around -1200 compared to the epsilon 0.01 in Table 2 is -388 to be exact. If the total reward is low but still managed to reach the destination, this means two things. 1) The solution is still not coverage and can be improve or 2) It has many repeated routes that cause the total reward to be low.

Value Table

The ideal path generated using Q-learning are shown below:

```
[4], [1], [1], [1], [3], [1], [4], [4], [4], [4], [1], [4], [2], [2], [2],
[3], [2], [4], [4], [4], [2], [4], [1], [3], [1], [4], [4], [4], [2], [4],
[1], [1], [3], [1], [4], [4], [1], [4], [2], [2], [2]]
```

Figure 1: Policy for Q learning

The optimal action is highlighted in red for each state.

Table 3: Value Table for Q learning

State	Actions				
	UP	Down	Left	Right	Best Action
((5,0), (4,1))	-294.956	-300.025	-298.063	-293.473	Right
((5,1), (4,1))	-285.176	-291.020	-298.886	-293.343	Up
((4,1), (3,1))	-275.691	-286.992	-278.913	-283.235	Up
((3,1), (2,1))	-264.9905	-270.643	-268.347	-274.807	Up
((2,1), (1,1))	-259.448	-264.598	-253.051	-253.628	Left
((2,0), (1,1))	-239.84	-257.511	-248.422	-258.853	Up
((1,0), (1,1))	-245.253	-251.098	-239.715	-227.396	Right
((1,1), (1,2))	-217.267	-233.183	-230.825	-215.710	Right
((1,2), (1,3))	-212.516	-211.737	-208.240	-204.808	Right
((1,3), (1,4))	-195.845	-1013.018	-208.518	-194.700	Right
((1,4), (1,5))	-185.408	-197.956	-191.669	-885.5	Up
((0,4), (1,5))	-185.340	-192.723	-182.698	-174.907	Right
((0,5), (1,5))	-174.634	-165.211	-177.613	-950.625	Down
((1,5), (2,5))	-165.358	-156.338	-165.441	-981.343	Down
((2,5), (3,5))	-157.951	-148.304	-151.924	-1011.012	Down
((3,5), (4,5))	-146.525	-145.437	-141.126	-1003.102	Left
((3,4), (4,5))	-138.946	-132.782	-948.75	-144.794	Down
((4,4), (4, 5))	-138.246	-128.892	-995.203	-125.288	Right
((4,5), (4,6))	-122.729	-126.403	-122.038	-118.661	Right
((4,6), (4,7))	-1005.058	-113.016	-117.016	-112.919	Right
((4,7), (4,8))	-114.787	-108.0813	-112.605	-880.25	Down
((5,7), (4,8))	-108.147	-106.849	-109.003	-102.128	Right
((5,8), (4,8))	-97.065	-102.072	-107.167	-945	Up
((4,8), (3,8))	-91.781	-97.590	-90.882	-882.875	Left
((4,7), (3,8))	-83.554	-93.735	-94.031	-94.470	Up
((3,7), (3,8))	-78.507	-79.417	-882.875	-77.095	Right
((3,8), (3,9))	-75.762	-74.8	-80.300	-71.526	Right
((3,9), (3,10))	-975.531	-755.25	-67.758	-66.864	Right
((3,10), (3,11))	-63.971	-63.121	-64.325	-988.313	Down
((4,10), (3,11))	-63.688	-64.170	-755.25	-58.291	Right
((4,11), (3,11))	-54.379	-61.689	-58.825	-880.25	Up
((3,11), (2,11))	-49.366	-55.762	-52.251	-944.063	Up
((2,11), (1,11))	-47.652	-44.769	-43.231	-882	Left
((2,10), (1,11))	-35.950	-41.396	-756.75	-39.827	Up
((1,10), (1,11))	-30.908	-31.814	-1000.125	-29.541	Right
((1,11), (1,12))	-24.806	-27.657	-27.751	-24.022	Right
((1,12), (1,13))	-19.401	-880.25	-20.654	-23.421	Up
((0,12), (1,13))	-17.056	-15.702	-23.721	-13.683	Right
((0,13), (1,13))	-9.851	-8.8608	-18.777	-11.767	Down
((1,13), (2,13))	-5.913	-4.96	-6.158	-5.940	Down
((2,13), (3,13))	-3	-2	-753	-2.985	Down
((3,13), (4,13))	GOAL				

The figures below demonstrate a step-by-step agent location to the box location. The full step-by-step policy will be placed under Annex A. The state contains two different coordinates, the agent coordinates and box coordinates. The agent needs to reach the box location to update the next location for the box. For example, in Figure 2, the agent location is represented as A and Box location is represented as B. The initial agent location is (5,0) and the Box location is (4,1). Figure 2 show the correct direction to move from agent (5,0) to Box (4,1) by first moving → to (5,1) then move ↑ to reach Box (4,1). I also added different colours for each action for better visualization.

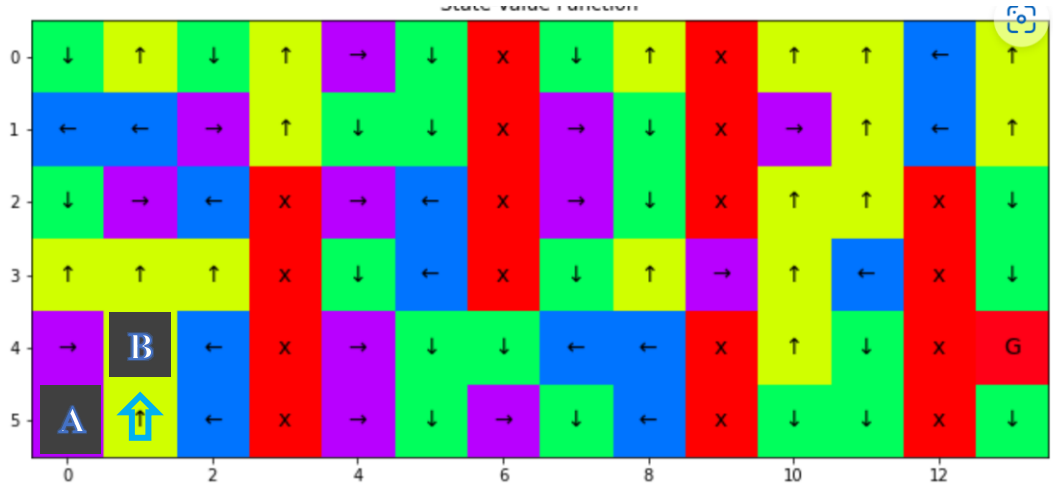


Figure 2: The direction of Agent (5,0) toward the Box location (4,1)

Next the agent (4,1) will move ↑ to Box (3,1).

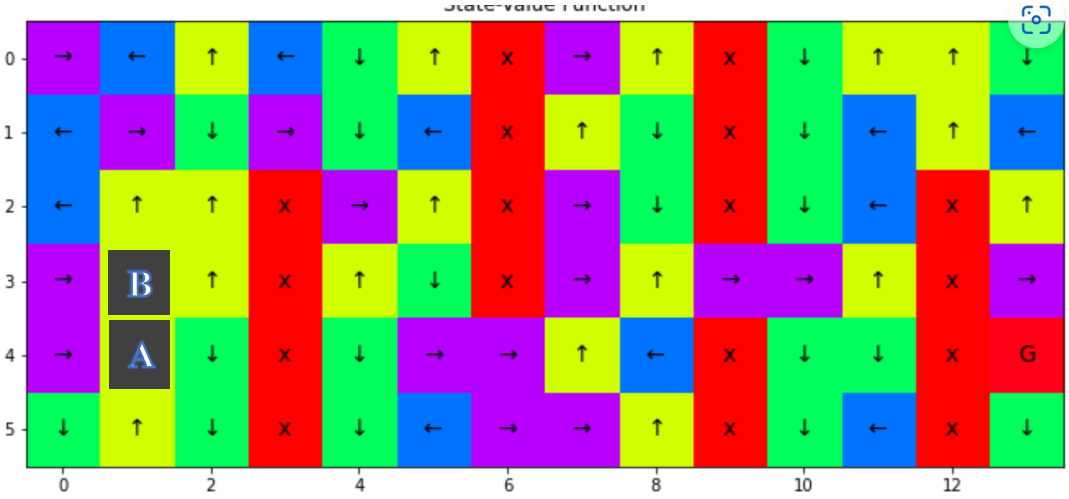


Figure 3: The direction of Agent (4,1) toward the Box location (3,1)

Policy for Q learning

This is the ideal policy for Q learning. I divide the trajectory into two sections because it bypasses the point (4,7) twice with different action. The red arrow in coordinate (4,8) in Figure 4 mean that it will continue the remaining trajectory at coordinate (4,8) in Figure 5.

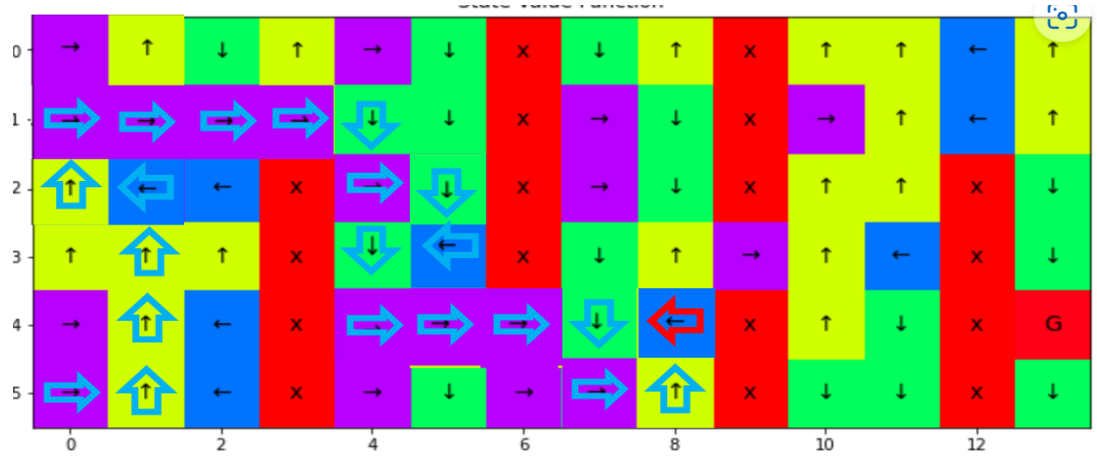


Figure 4: Policy for Q learning with arrows from (5,0) to (4.8)

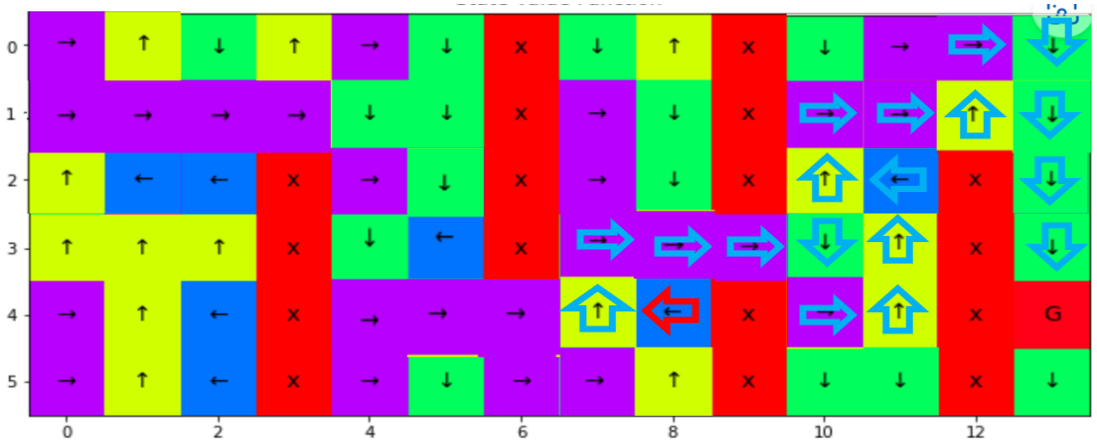


Figure 5: Policy for Q learning with arrows from (4,8) to Goal (4.13)

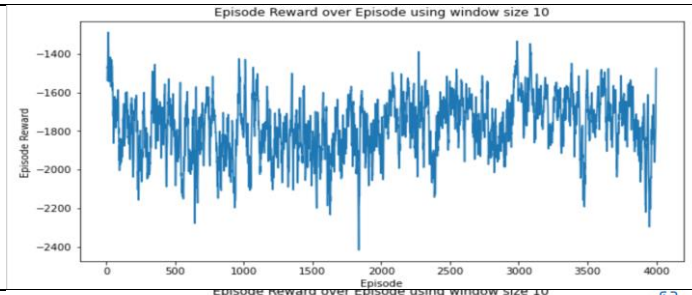
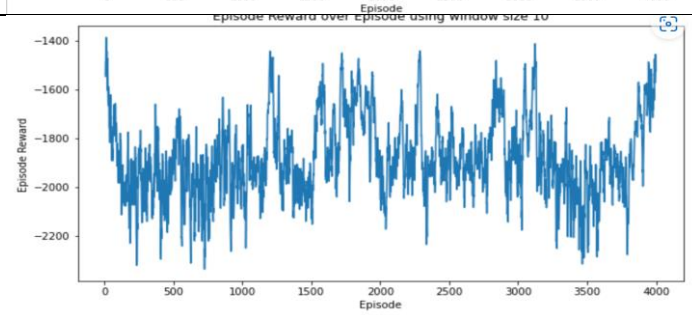
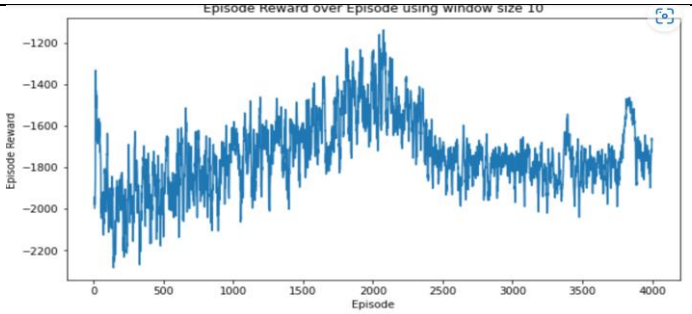
SARSA

SARSA is an on policy that updates its Q values by using the Q values of the next state and the current policy action. For on policy, the target policy is equal to the behaviour policy. It evaluates and improve the same policy from that policy that is used for the action.

I tested the same parameters in Table 1 and Table 2 for SARSA. However, those parameters could not make SARSA to push the box to the goal. A lot of times, the agent ignores the box to reach the goal by itself. After many finetuning, I figure out that epsilon 0.001 is effective for SARSA. So, I readjusted the epsilon and learning rate for Table 4 and 5. In Table 5, I used learning rate 0.32. It is a weird figure, but it works. I also realised SARSA need longer duration to coverage even though in theory SARSA should be faster than Q learning and SARSA is easily stuck in local minimum.

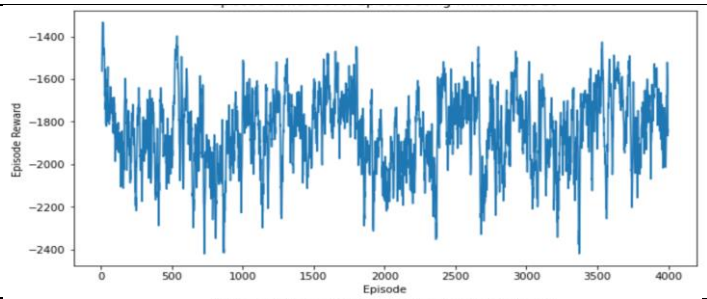
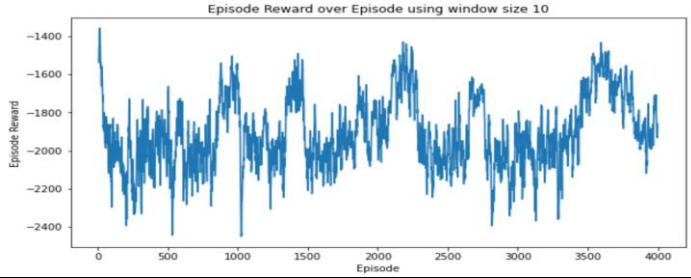
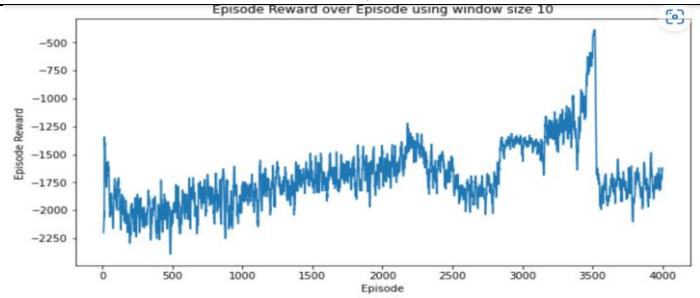
Plotting of Episode rewards vs. Episodes with learning rate of 0.5

Table 4: Plotting Episode Reward over Episodes with learning rate of 0.5 and different epsilon values

	Epsilon	Episode Reward over Episodes	Push Box to Goal	Total Reward
Learning rate = 0.5	0.1		No	NIL
	0.01		No	NIL
	0.001		Yes	~1200

Plotting of Episode rewards vs. Episodes with learning rate of 0.32

Table 5: Plotting Episode Reward over Episodes with learning rate of 0.32 and different epsilon value

	Epsilon	Episode Reward over Episodes	Push Box to Goal	Total Reward
Learning rate = 0.32	0.1		No	NIL
	0.01		No	NIL
	0.001		Yes	~500

From Table 4 and 5, the ideal parameters for SARSA are learning 0.32 and epsilon 0.001 highlighted in red. The total reward is around -500 while in Table 4 is around -1200.

Value Table

The ideal path generated using SARSA are shown below:

```
print the historical actions:[[4], [1], [1], [1], [3], [1], [4], [4], [4], [4]
, [1], [4], [2], [2], [2], [3], [2], [4], [4], [4], [2], [4], [1], [3], [1], [
4], [4], [4], [2], [4], [1], [1], [3], [1], [4], [4], [1], [4], [2], [2], [2]]
```

Figure 6: Policy for SARSA

The optimal action is highlighted in red for each state.

Table 6: Value Table for Q learning

State	Actions				
	UP	Down	Left	Right	Best Action
((5,0), (4,1))	-339.208	-340.030	-339.518	-292.133	Right
((5,1), (4,1))	-284.152	-334.614	-335.510	-333.086	Up
((4,1), (3,1))	-274.888	-324.817	-324.467	-325.731	Up
((3,1), (2,1))	-264.296	-315.349	-314.839	-316.547	Up
((2,1), (1,1))	-309.004	-305.280	-252.356	-306.159	Left
((2,0), (1,1))	-239.073	-297.128	-299.167	-296.129	Up
((1,0), (1,1))	-287.468	-290.896	-287.348	-226.508	Right
((1,1), (1,2))	-279.619	-282.431	-280.623	-214.720	Right
((1,2), (1,3))	-271.941	-270.599	-270.867	-203.758	Right
((1,3), (1,4))	-241.938	-545.664	-243.040	-194.962	Right
((1,4), (1,5))	-190.097	-236.456	-236.564	-544.051	Up
((0,4), (1,5))	-234.418	-231.728	-232.139	-180.165	Right
((0,5), (1,5))	-227.389	-164.419	-219.966	-324.48	Down
((1,5), (2,5))	-222.630	-155.668	-227.915	-324.16	Down
((2,5), (3,5))	-202.446	-147.755	-202.812	-544.051	Down
((3,5), (4,5))	-197.090	-197.373	-140.690	-543.513	Left
((3,4), (4,5))	-194.359	-132.447	-323.84	-194.430	Down
((4,4), (4, 5))	-188.161	-185.788	-543.516	-125.041	Right
((4,5), (4,6))	-182.631	-181.893	-182.511	-118.487	Right
((4,6), (4,7))	-542.438	-168.837	-168.837	-112.802	Right
((4,7), (4,8))	-165.757	-108.006	-165.618	-321.92	Down
((5,7), (4,8))	-161.357	-163.366	-162.804	-102.078	Right
((5,8), (4,8))	-97.038	-158.901	-159.352	-322.56	Up
((4,8), (3,8))	-133.214	-134.525	-80.540	-322.88	Down
((4,7), (3,8))	-73.399	-129.096	-129.164	-130.078	Up
((3,7), (3,8))	-126.486	-126.319	-328.88	-67.143	Right
((3,8), (3,9))	-144.082	-143.325	-142.936	-71.524	Right
((3,9), (3,10))	-322.24	-541.363	-67.104	-66.863	Right
((3,10), (3,11))	-63.647	-63.125	-63.805	-321.28	Down
((4,10), (3,11))	-58.685	-59.235	-322.24	-58.291	Right
((4,11), (3,11))	-54.379	-54.448	-55.443	-321.92	Up
((3,11), (2,11))	-49.367	-49.907	-49.980	-322.24	Up
((2,11), (1,11))	-46.491	-43.760	-43.231	-322.56	Left
((2,10), (1,11))	-35.950	-36.278	-322.88	-40.116	Up
((1,10), (1,11))	-30.067	-32.663	-322.56	-29.541	Right
((1,11), (1,12))	-24.645	-24.916	-24.523	-24.022	Right
((1,12), (1,13))	-19.409	-321.92	-20.907	-20.317	Up
((0,12), (1,13))	-15.983	-14.742	-15.001	-13.683	Right
((0,13), (1,13))	-10.650	-8.860	-11.727	-11.258	Down
((1,13), (2,13))	-5.651	-4.96	-5.694	-6.757	Down
((2,13), (3,13))	-2.150	-2	321.28	-3.423	Down
((3,13), (4,13))	Goal				

Same for SARSA, the step-by-step policy is placed under ANNEX B.

Policy for SARSA

Surprisingly, the ideal policy for SARSA is similar as Q learning. I divide the trajectory into two sections because it bypasses the point (4,7) twice with different action. The red arrow in coordinate (4,8) in Figure 7 mean that it will continue the remaining trajectory at coordinate (4,8) in Figure 8.

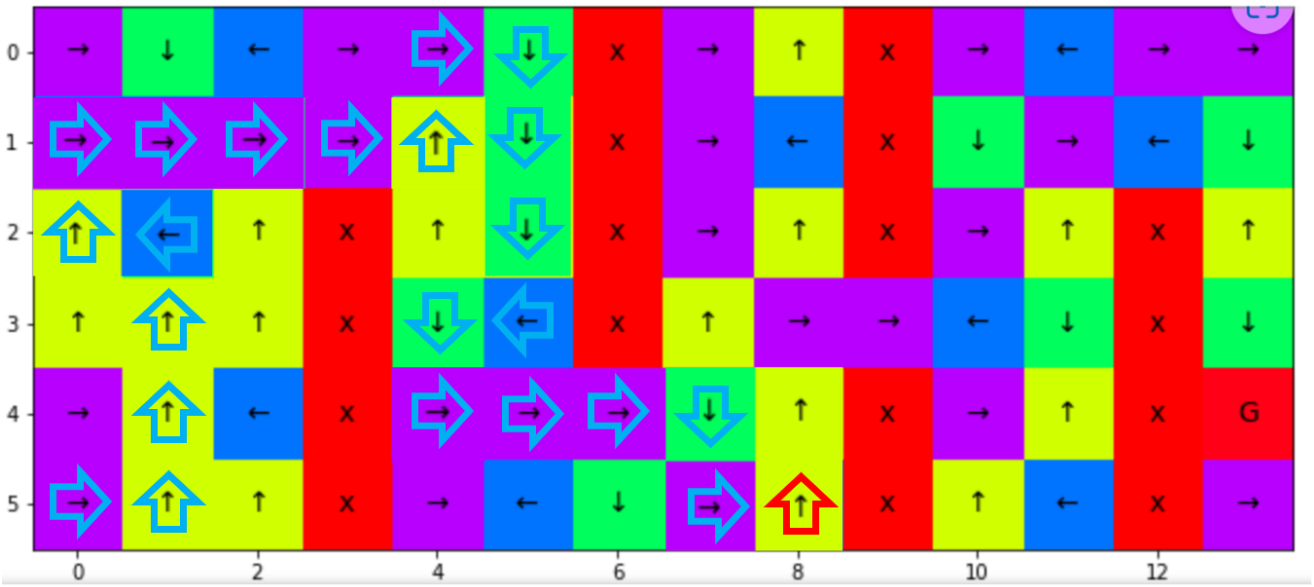


Figure 7: Policy for Q SARSA with arrows from (5,0) to (4.8)

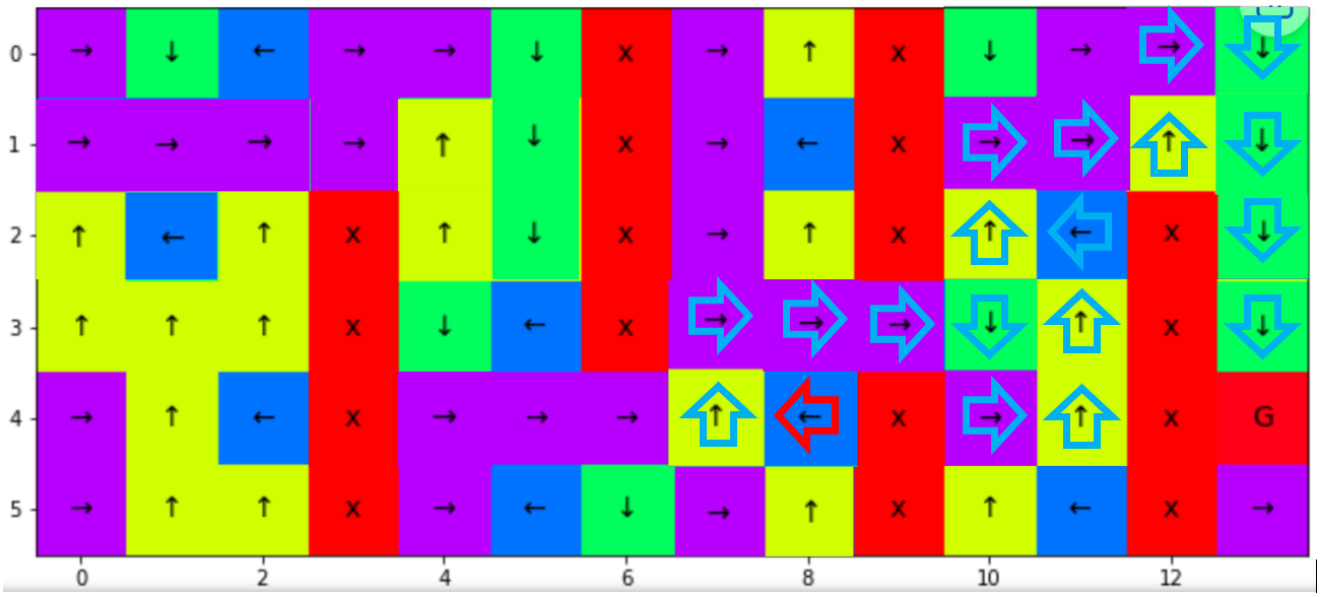


Figure 8: Policy for SARSA with arrows from (4,8) to Goal (4.13)

Experiment with Epsilon Decay

In theory, we often read about setting the epsilon to be high to let the agent to explore and slowly decay the epsilon. In this experiment, I will be applying epsilon decay to Q learning and SARSA to experiment if epsilon decay indeed improves the performance. Table 7 show the target we need to break to get better result for either Q learning or SARSA. Will by applying epsilon require less step to reach goal?

Table 7: Result before applying epsilon decay

	No. of Episode to reach goal	Step to reach goal
Q learning	~1.6k	41
SARSA	~3.5k	41

The formula of the epsilon decay I will be using is:

$$\varepsilon' = \varepsilon_{end} + (\varepsilon_{start} - \varepsilon_{end}).e^{(\frac{-1\ x\ N_i}{\varepsilon_{decay\ rate}})}$$

where ε_{start} is the initial epsilon value, ε_{end} is the final epsilon, $\varepsilon_{decay\ rate}$is the rate of the initial epsilon being decayed and N_i is the i^{th} number of episodes.

Table 8: Experiment with different initial epsilon, different final epsilon and different decay rate for Q learning and SARSA

With decay epsilon	No. of episode	ε_{start}	ε_{end}	Decay rate	No. of Episode to reach goal	Steps to reach goal
Q learning	2k	1	0.01	200	~1.9k	41
Q learning	2k	0.5	0.01	200	~1.7k	41
Q learning	2k	0.1	0.01	200	~1.71k	41
Q learning	10k	0.1	0.01	500	~1.7k	41
Q learning	2k	0.01	0.001	200	~1.5k	41
Q learning	10k	0.01	0.001	500	~1.5k	41
SARSA	4k	1	0.001	500	NIL	NIL
SARSA	4k	0.5	0.001	200	NIL	NIL
SARSA	4k	0.01	0.001	200	~3.5k	41
SARSA	10k	0.01	0.001	50	~3.6k	41
SARSA	4k	0.001	0.0005	100	~2.6k	41
SARSA	10k	0.001	0.0005	100	~2.6k	41

From the Table 8, I experiment many different decays rate with different initial epsilon, and I also discovered that Q learning for this environment the epsilon must be around 0.01 to reach the destination. And for SARSA the epsilon must be as low as 0.001. I tried exploration by setting initial epsilon as high as 1 and slowly decay. It only starts to optimal when the epsilon is around 0.001 for SARSA and 0.01 for Q learning. The reasons for this will be discuss in the Discussion section.

For Q learning, 0.01 initial epsilon, 0.001 end epsilon with 200 decay rate is the best parameters. It uses 1.5k episodes to reach the goal which is slightly better than without epsilon decay. However, no matter how I finetuned the epsilon the optimal steps to reach goal is still 41.

For SARA, 0.001 initial epsilon, 0.0005 with decay rate of 100 is the best parameters. It uses around 2.6k episodes to reach goal which is relatively improve compared to one without epsilon decay. Similarly, the number of steps is still 41.

Experiment with Learning Rate Decay

There are still other parameters we can experiment with, which is the learning rate. In deep learning, we often use learning rate scheduler to slow down the learning when converging. The purpose of applying this is to prevent the gradient being stuck at the local minima. Therefore, I will be using the concept to decay the learning rate decay using step decay.

The formula for the step decay is as follow:

$$LR_{Final} = LR_{Initial} \,.\, (decay\ rate)^{i}$$

where $LR_{Initial}$ is the initial learning rate, $decay\ rate$ is the rate of learning rate to decay and i is the number of iterations.

Table 9: Experiment with different initial learning rate and different decay rate for Q learning and SARSA

	No. of episode	Initial lr	Final lr	Decay rate	No. of Episode to reach goal	Step to reach goal
Q learning	2k	1.0	0.833	0.0001	~1.6k	41
Q learning	2K	1.0	0.505	0.0005	~1.9k	41
Q learning	10k	1.0	0.5	0.0001	~1.5k	41
Q learning	10k	1.0	0.09	0.001	~2.3k	41
Q learning	10k	0.5	0.25	0.0001	~2.3k	41
SARSA	4k	1.0	0.294	0.0006	NIL	NIL
SARSA	4k	0.5	0.118	0.0009	~3.9k	52
SARSA	4k	0.35	0.257	0.00009	~3.5k	41

Learning rate is important parameters. I experiment different range of learning rate and Q learning require a high learning rate to learn faster. Slower learning rate need longer episodes to reach the goal. For SARA, the learning rate is around 0.3. If the learning rate is higher than that, it will not reach to the goal.

Therefore, from Table 9 the ideal learning rate decay for Q learning start from 1 then apply 0.0001 decay rate. It required 1.6k episodes to reach the goal which is around the same as without learning rate decay. For SARSA, the ideal learning rate decay start from 0.35 and end of 0.257.

Experiment with Epsilon Decay & Learning rate Decay

Next, we combined both epsilon decay and learning rate decay to Q Learning and SARSA.

Table 10: Result after applying epsilon decay and learning rate decay for Q learning and SARSA

	No. of episode	No. of Episode to reach goal	Step to reach goal
Q learning	2k	~1.5k	41
SARSA	4K	~2.7k – 3.1k	41

From Table 8 and Table 9, I select the best epsilon decay and learning rate decay highlighted in red to Q learning and SARSA. Apparently, SARSA improve from 3.6k episode to 2.7k episode to reach the goal and Q learning improve from 1.6k to 1.5k.

Ultimately, our goal is to improve the number of steps to reach goal. It seemed that 41 steps are the optimal minimum required steps for both Q learning and SARSA.

Discussion

Now, we need to discuss why exploration does not work. In this environment, exploration does not work. The diagram below, Figure 9, has labelled five different sections. For example, we train 1k episodes and epsilon is set to 0.9 with decay. First the agent will explore around Section 1 in Figure 9 and let's say it managed to reach Section 2. The problem arises. The epsilon already decayed and there are still 4 more sections. The agent is not able to explore another four sections unless the epsilon is reset every time it reaches to the different section. Then it will start to explore. But if we reset the epsilon, isn't it take longer episodes to reach the goal? Thus, setting epsilon to be very low is the optimal solution for this environment as it chooses the best actions in every states.

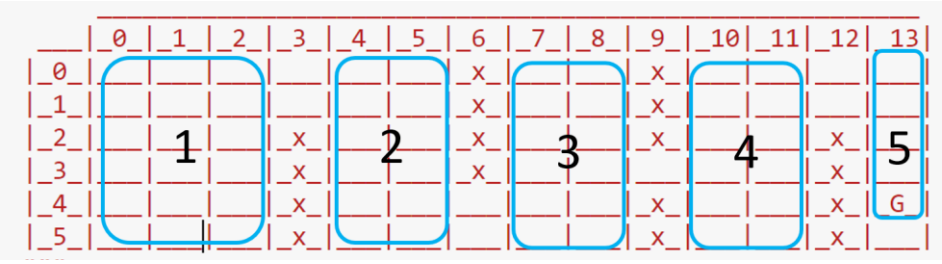


Figure 9: Environment divided into 5 sections

Unlike the Cliffwalking example, the exploration method works because the agent can explore all the states (from (3,0) to (2,11)) before exploitation.

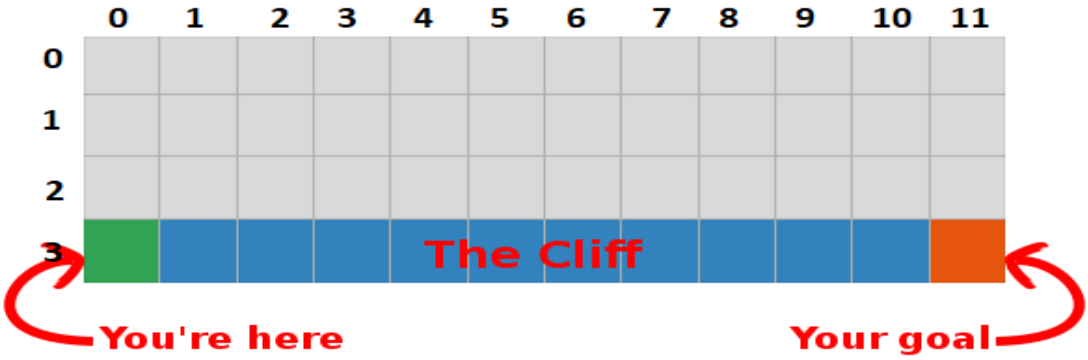


Figure 10: Example of an environment from gym - Cliffwalking

Evaluation

Table 11: The table that consists of all the optimal results experimented using Q learning and SARSA

	Learning Rate	Gamma Value	Epsilon	No. of episodes to reach G (minimum steps)	No. of steps to reach G	Total Rewards
Q learning	0.9	0.98	0.01	~1600	41	-388
SARSA	0.32	0.98	0.001	~3600	41	-388
Q learning (with learning rate decay and epsilon decay)	1.0 – 0.83	0.98	0.01-0.001	~1500	41	-388
SARSA (with learning rate decay and epsilon decay)	0.35-0.257	0.98	0.001-0.0005	~2700	41	-388

In conclusion, the minimum steps to reach goal is 41 for both Q learning and SARSA. The only different is the number of episodes to reach the destination. For Q learning, by applying learning rate decay and epsilon decay increase the performance by reaches the goal earlier and for SARSA by applying learning rate decay and epsilon decay it reduces the number of episodes from 3600 to 2700. Since all the algorithms took 41 steps to reach, we must determine which algorithm reaches the goal first which is Q learning with learning rate decay and epsilon decay.

Extra (Q Learning without Box coordinate)

Out of curiosity, I decided to convert state into grid number instead of coordinates. For example, the state initially is (5,0) will be converted to 70. I converted by using the x-axis of the coordinates of the state multiply by the row of the environment which is 14 plus y-axis.

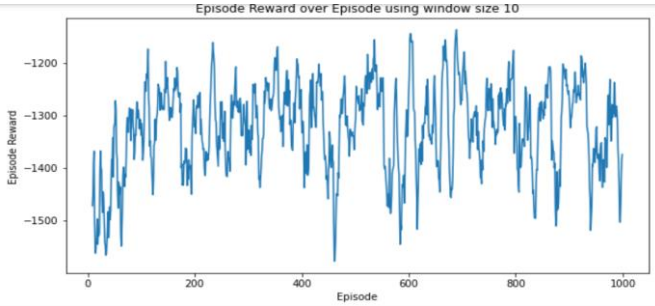
I removed the box coordinates and train using Q learning,

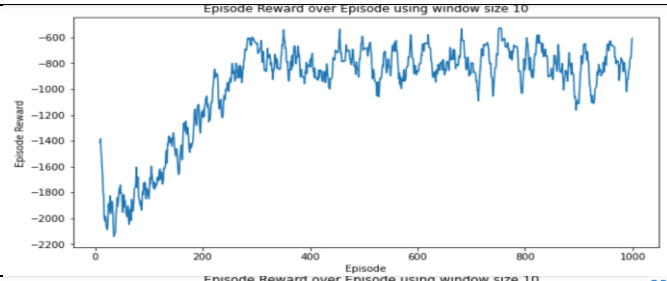
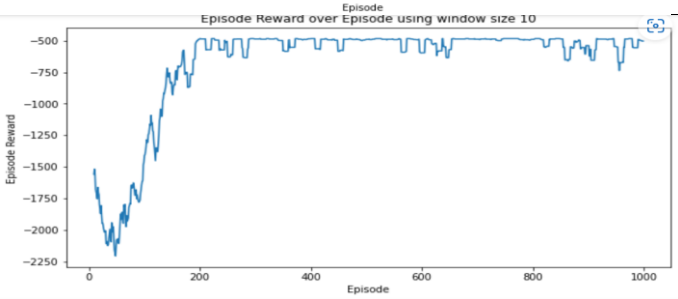
The purpose of this experiment is to analysis which trajectory does the agent move to the destination without the box location. Will it take 41 steps too?

Similarly, I will be using Q learning with learning rate 0.5 and 0.9.

Plotting of Episode rewards vs. Episodes with learning rate of 0.5

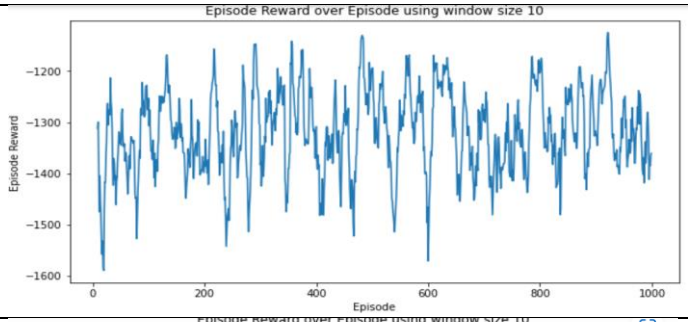
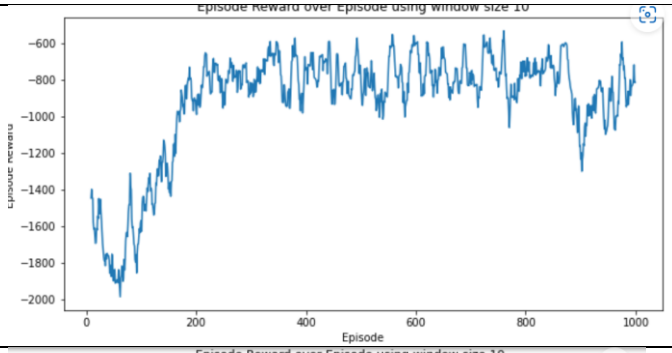
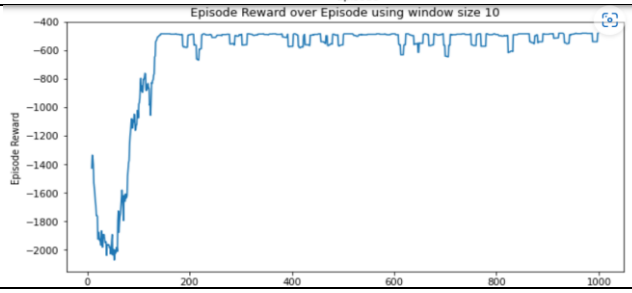
Table 12: Plotting Episode Reward over Episodes with learning rate of 0.5 and different epsilon values

	Epsilon	Episode Reward over Episodes	Pull Box to Goal	Total Rewards
Learning rate = 0.5	0.5		No	NIL

	0.1		Yes	~600
	0.01		Yes	~400

Plotting of Episode rewards vs. Episodes with learning rate of 0.9

Table 13: Plotting Episode Reward over Episodes with learning rate of 0.9 and different epsilon values

	Epsilon	Episode Reward over Episodes	Pull Box to Goal	Total Reward
Learning rate = 0.9	0.5		No	NIL
	0.1		Yes	~600
	0.01		Yes	~400

From Table 13, 0.01 epsilon and 0.9 learning rate took 200 episodes and only took 26 steps to reach the goal.

V Table

```
print the historical actions: [[1], [4], [1], [1], [4], [1], [4], [4], [2], [2], [2], [4], [4], [4], [1], [4], [4], [4], [1], [1], [4], [4], [4], [2], [2], [2]]
```

Table 14: V Table for Q learning without Box

	Actions				
	UP	Down	Left	Right	Best Action
70 (5,0)	-372.707	-401.168	-398.185	-401.606	Up
56 (4,0)	-395.518	-405.249	-396.591	-363.824	Right
57 (4,1)	-355.964	-400.171	-393.818	-910.8	Up
43 (3,1)	-347.576	-386.819	-391.968	-990.415	Up
29 (2,1)	-376.251	-385.431	-386.431	-338.643	Right
15 (1,1)	-372.437	-372.544	-381.557	-366.634	Right
16 (1,2)	-366.244	-370.359	-361.638	-322.305	Right
17 (1,3)	-355.142	-917.1	-352.014	-312.525	Right
18 (1,4)	-356.293	-301.549	-343.019	-1211.667	Down
32 (2,4)	-342.408	-291.375	-1005.03	-334.212	Down
46 (3,4)	-329.761	-282.015	-918	-320.748	Down
60 (4,4)	-318.522	-311.521	-1010.07	-273.484	Right
61 (4,5)	-310.547	-310.965	-307.778	-263.759	Right
62 (4,6)	-917.1	-289.225	-290.518	-252.816	Right
63 (4,7)	-240.628	-283.720	-285.410	-267.769	Up
49 (3,7)	-254.425	-261.537	-913.5	-227.172	Right
50 (3,8)	-259.835	-236.533	-239.662	-212.421	Right
51 (3,9)	-918.9	-918.9	-223.388	-196.347	Right
52(3,10)	-178.926	-200.541	-197.892	-198.042	Up
38 (2,10)	-160.128	-185.670	-920.7	-176.306	Up
24 (1,10)	-160.117	-163.168	-1011.96	-139.927	Right
25(1,11)	-142.189	-171.418	-174.618	-118.293	Right
26(1,12)	-117.947	-1015.56	-114.821	-95.197	Right
27(1,13)	-96.008	-70.609	-100.867	-102.131	Down
41 (2,13)	-94.385	-46.54	-921.6	-69.859	Down
55(3,13)	-49.779	-23	-1013.21	-46.3788	Down
69(4,13)	GOAL				

From the V table, we can see that all the coordinates that the agent move are the box location. It moves pass (4,1) then (3,1) and eventually reach the goal. If the agent follow the same box location does it mean this method work? But I did not include this as my solution. This is just an experiment.

Policy and Value Table for Q learning without Box coordinate

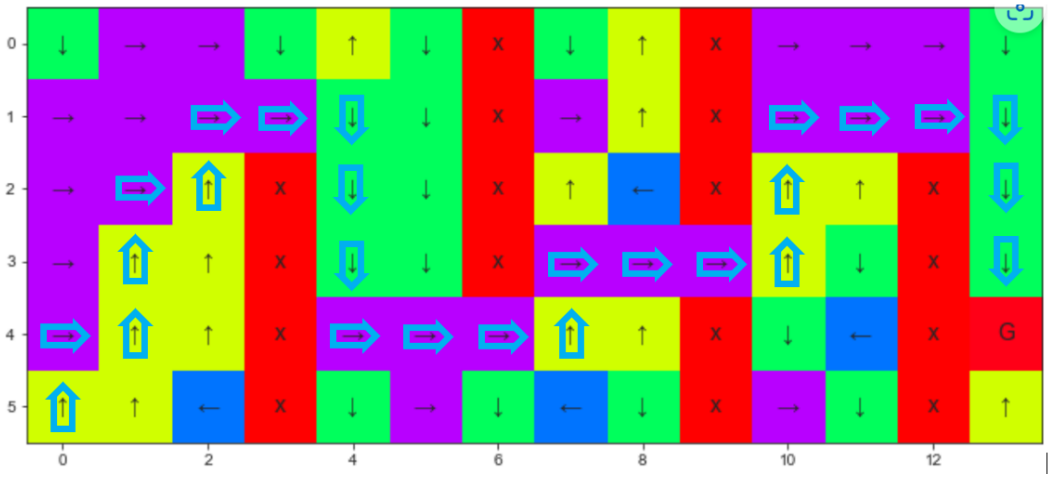


Figure 11: Policy for Q learning (with no box) with arrows from (5,0) to Goal (4.13)

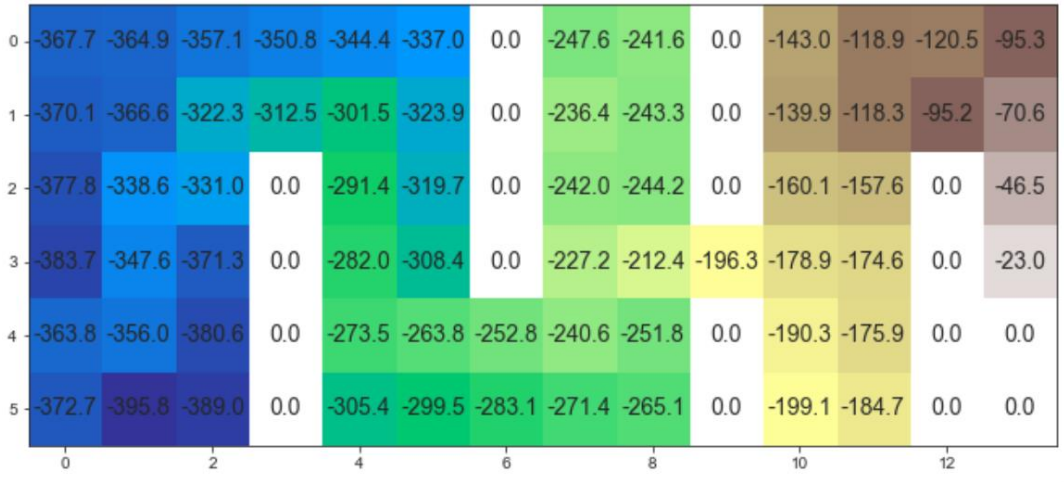
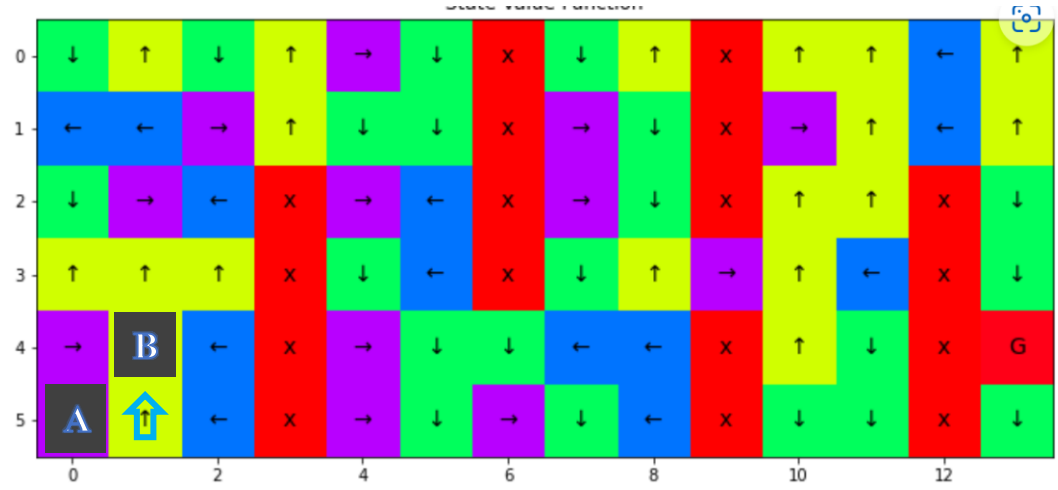


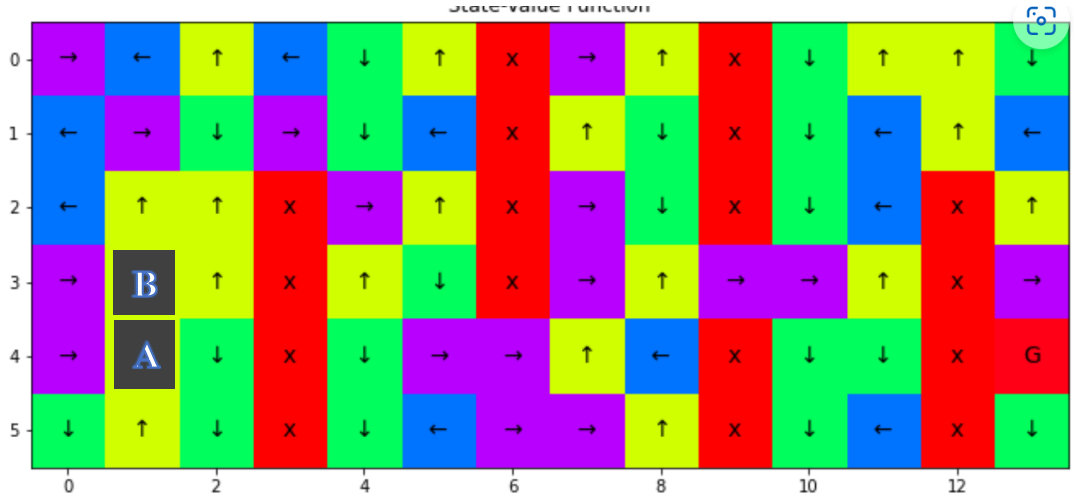
Figure 12: Policy for Q learning (with no box) with values for 6 x 14

ANNEX A (Q learning policy step by step)

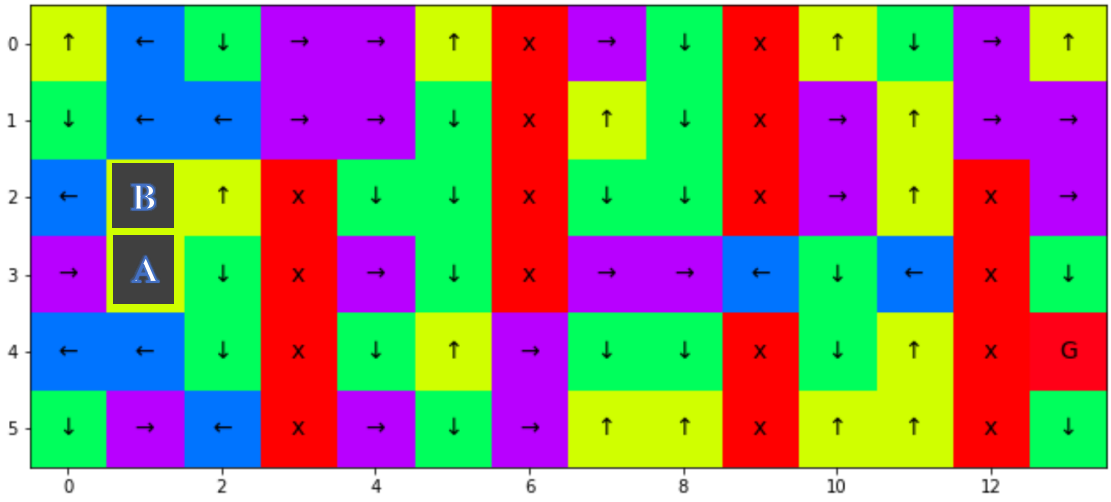
Agent (5,0) move → to (5,1) then move ↑ to Box (4,1)



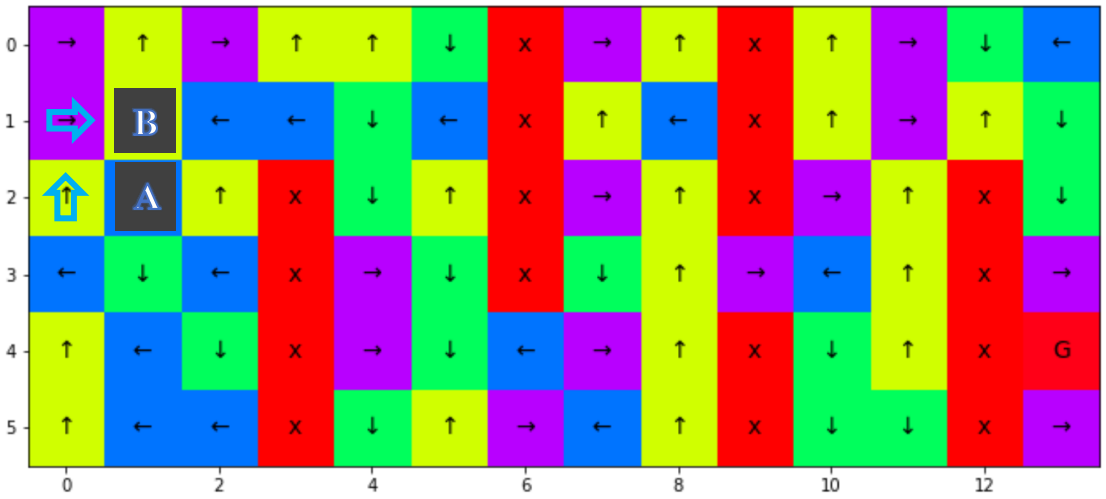
Agent (4,1) move ↑ to Box (3,1)



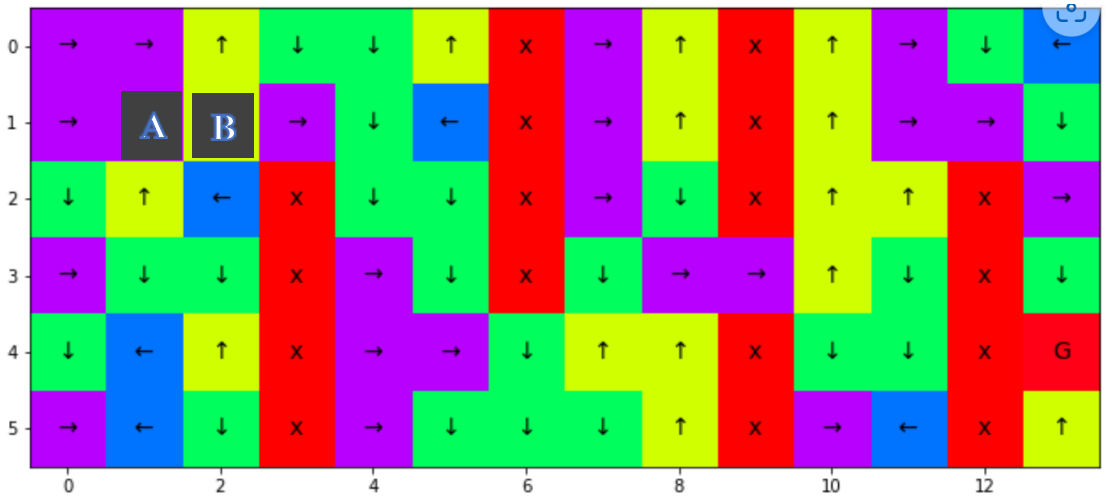
Agent (3,1) move ↑ to Box (2,1)



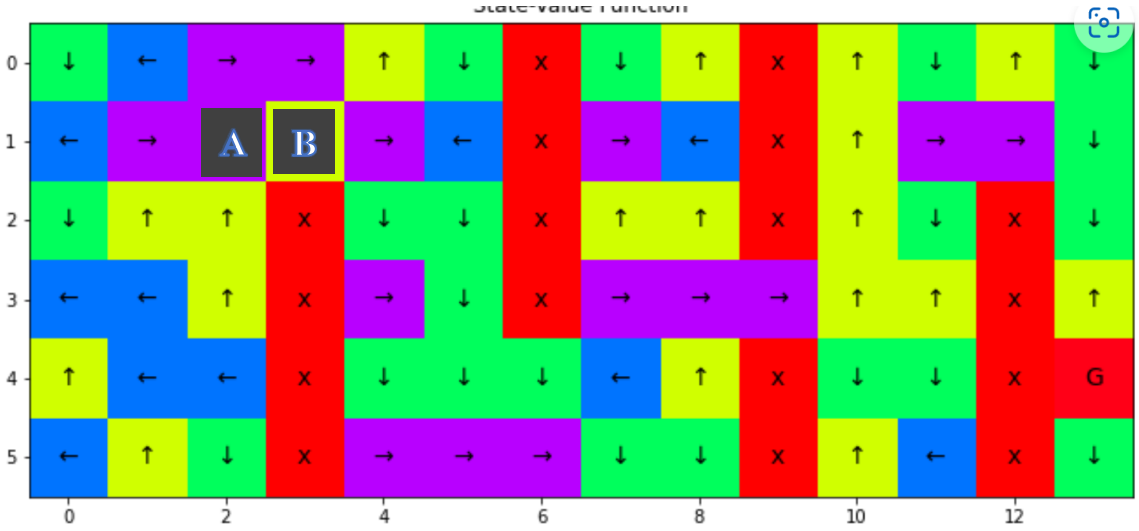
Agent (2,1) move ← to (2,0) then ↑ to (1,0) then → to Box (1,1)



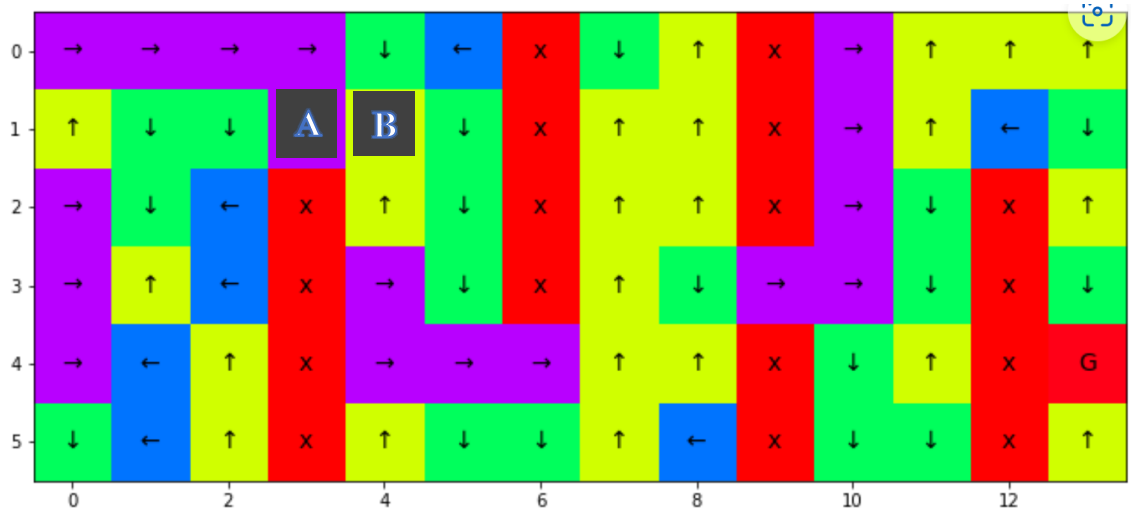
Agent (1,1) move → to Box (1,2)



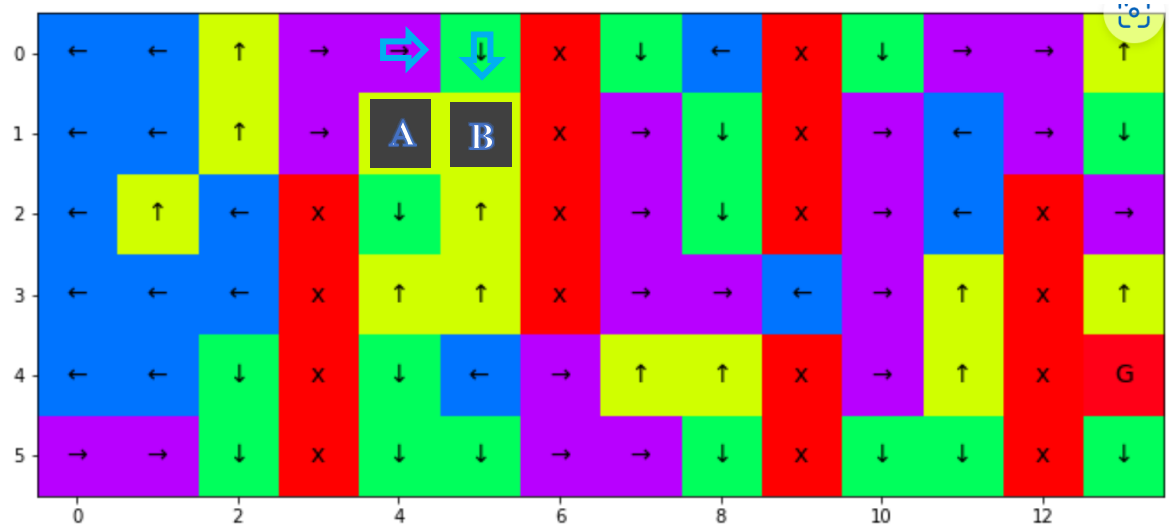
Agent (1,2) move → to Box (1,3)



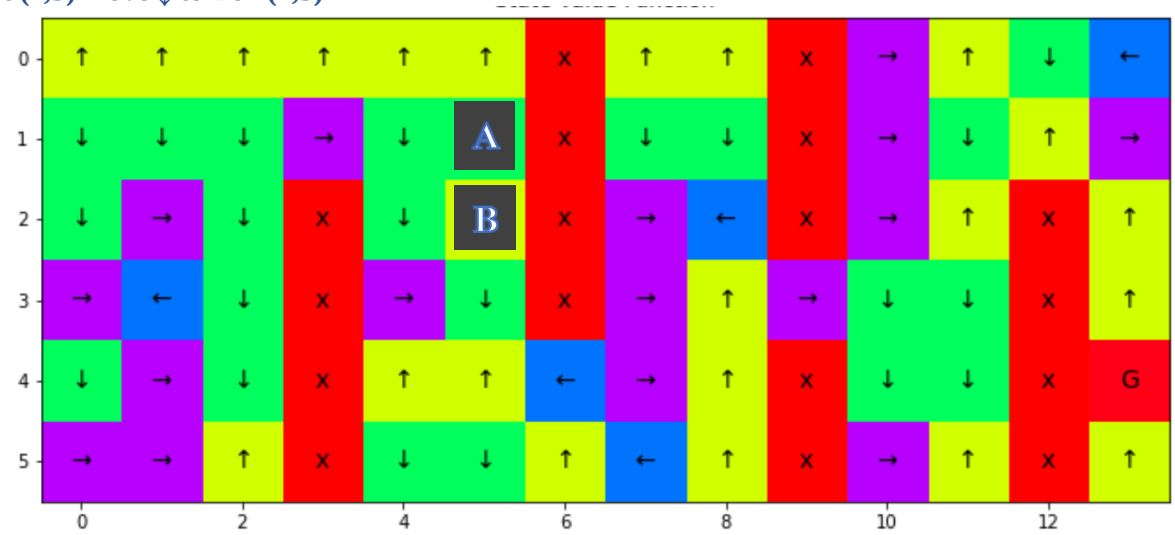
Agent (1,3) move → to Box (1,4)



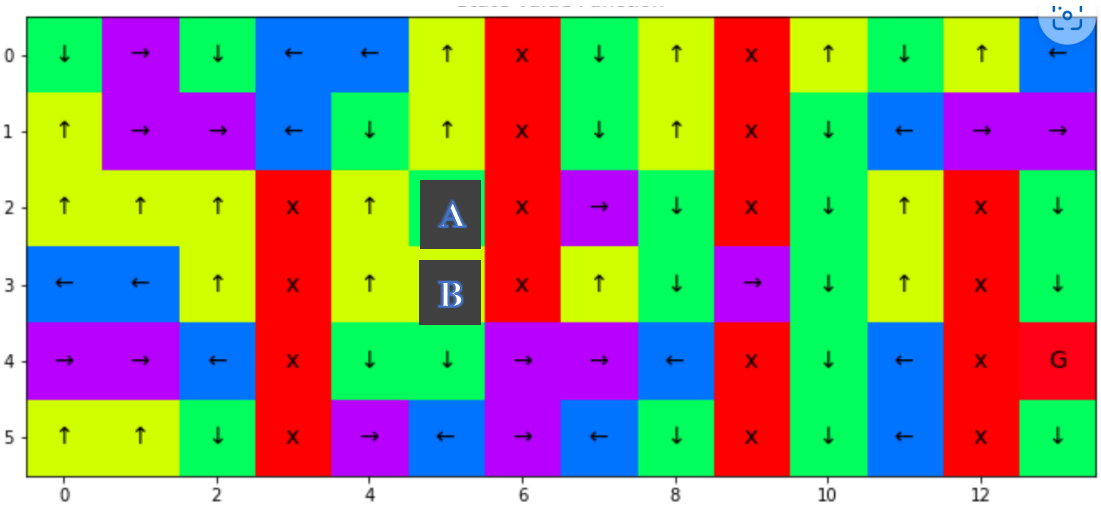
Agent (1,4) move ↑ to (0,4) then → to (0,5) then move ↓ to Box (1,5)



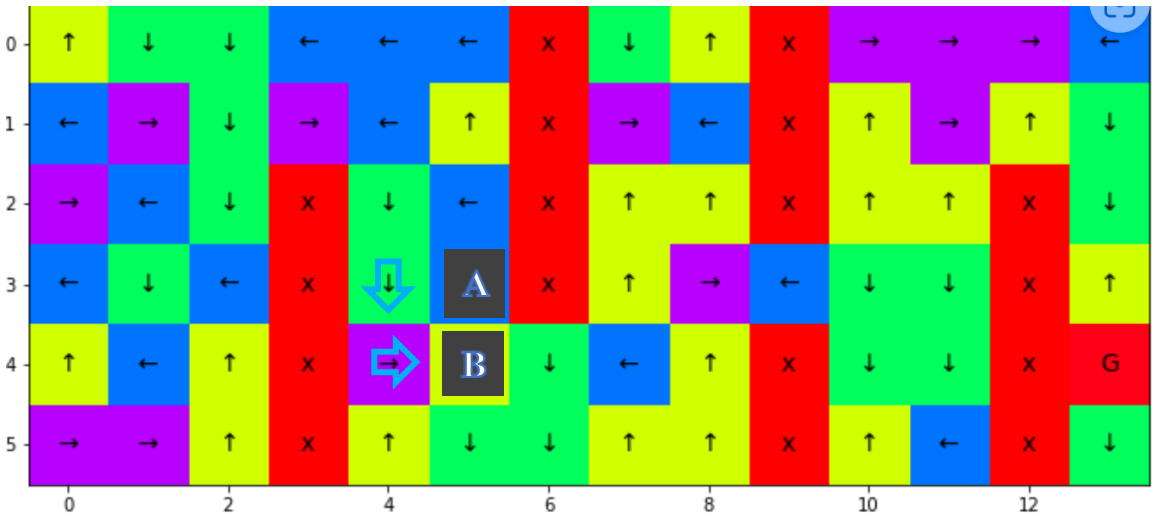
Agent (1,5) move ↓ to Box (2,5)



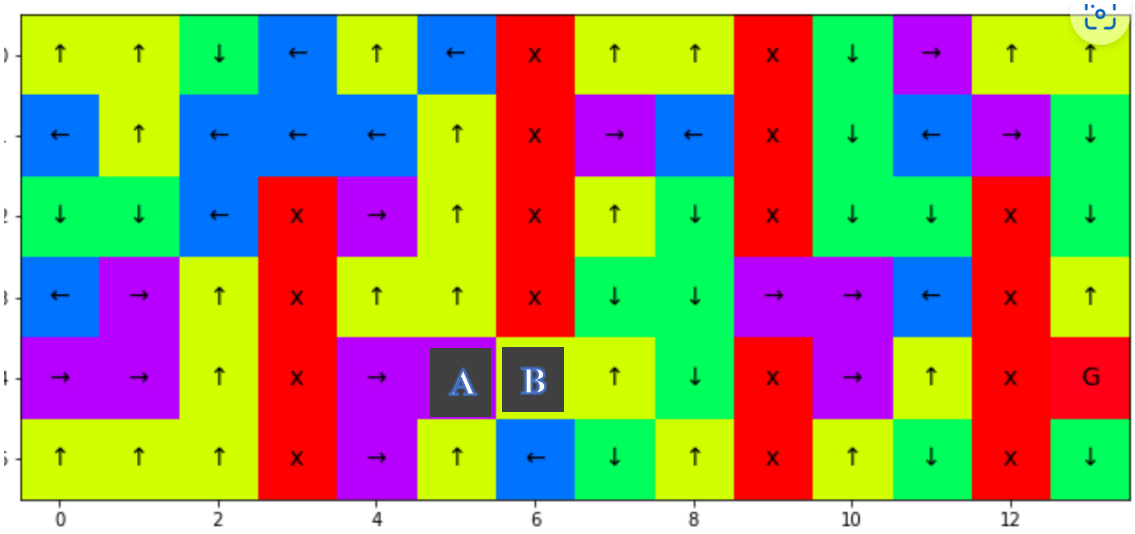
Agent (2,5) move ↓ to Box (3,5)



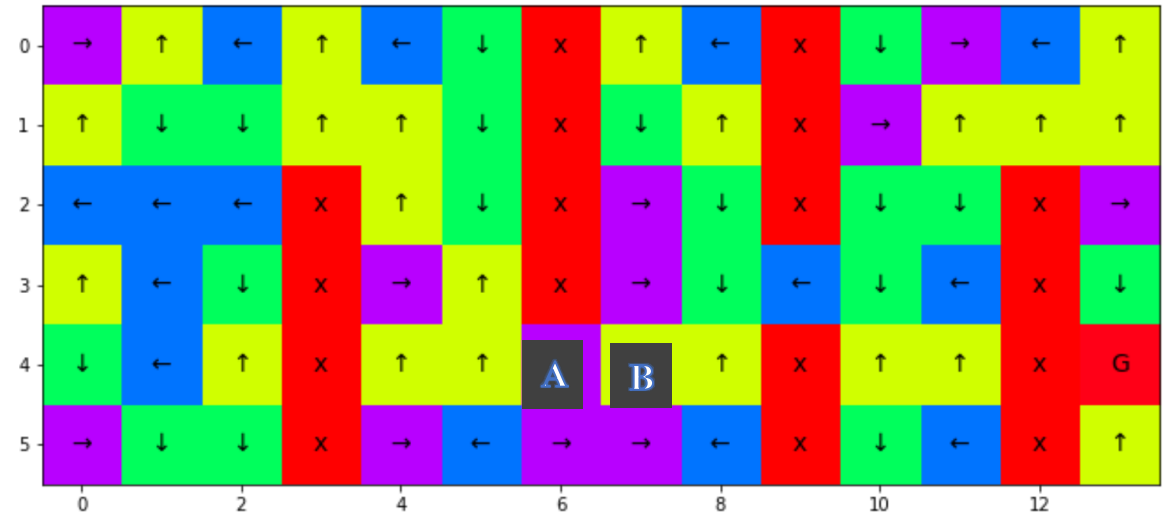
Agent (3,5) move ← to (3.4) the move ↓ to (4,4) then move → to Box (4,5)



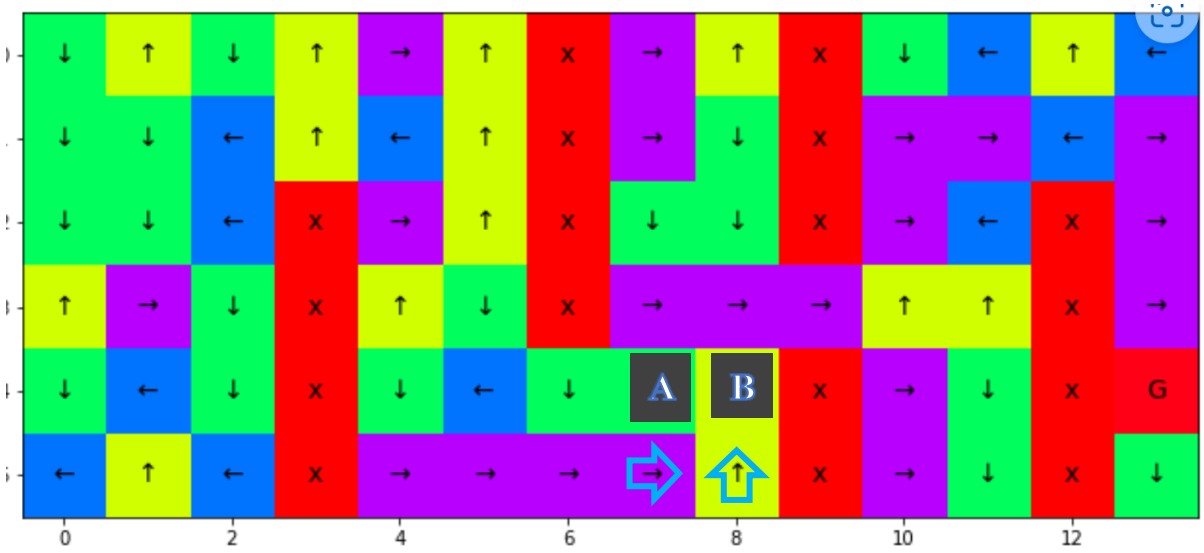
Agent (4,5) move → to Box (4,6)



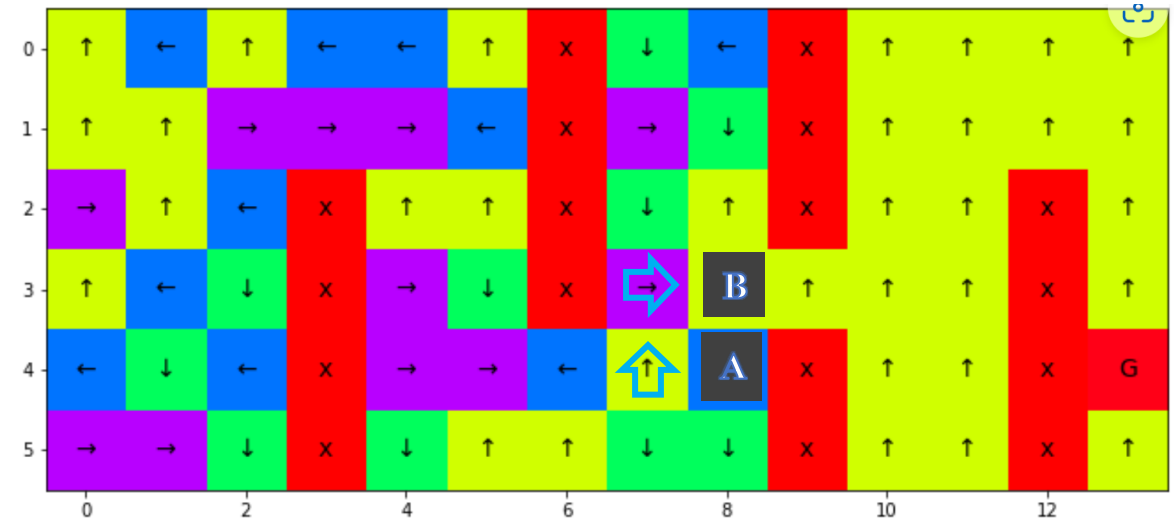
Agent (4,6) move → to Box (4,7)



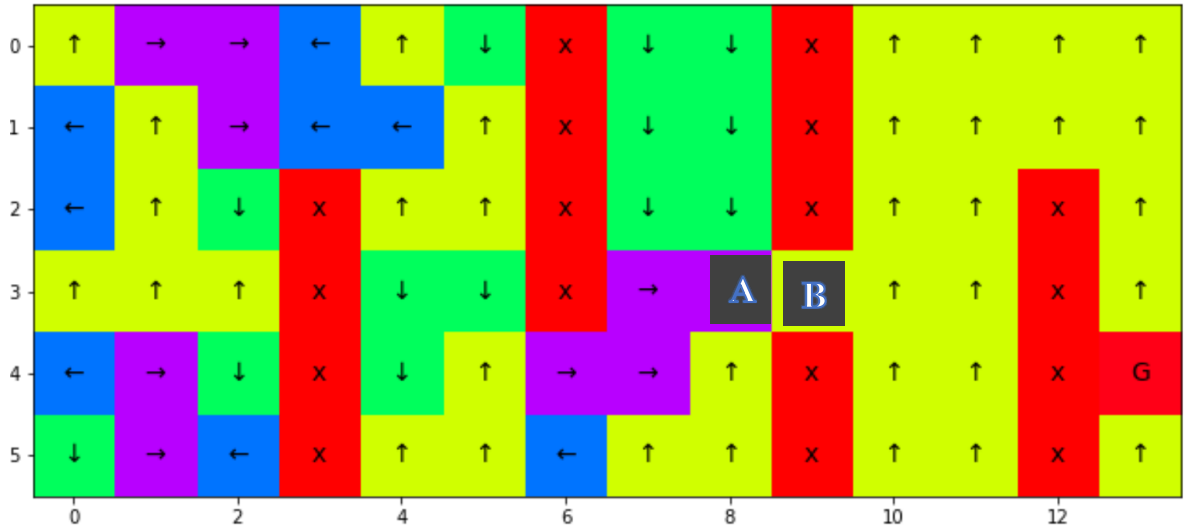
Agent (4,7) move ↓ to (5,7) then move → to (5,8) then move ↑ to Box (4,8)



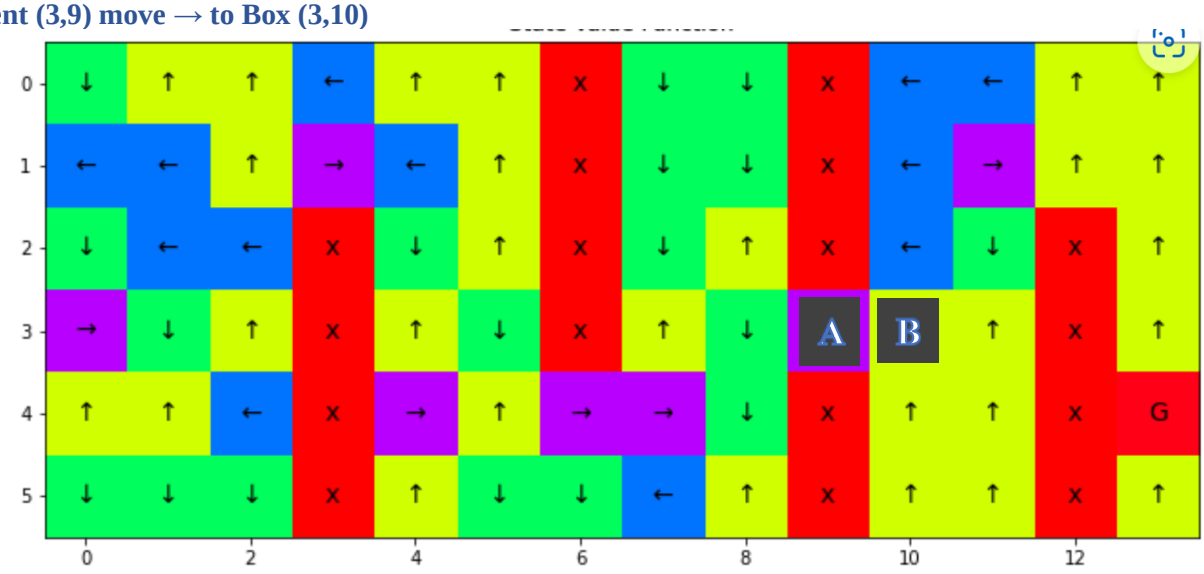
Agent (4,8) move ← to (4,7) then ↑ to (3,7) then → to Box (3, 8)



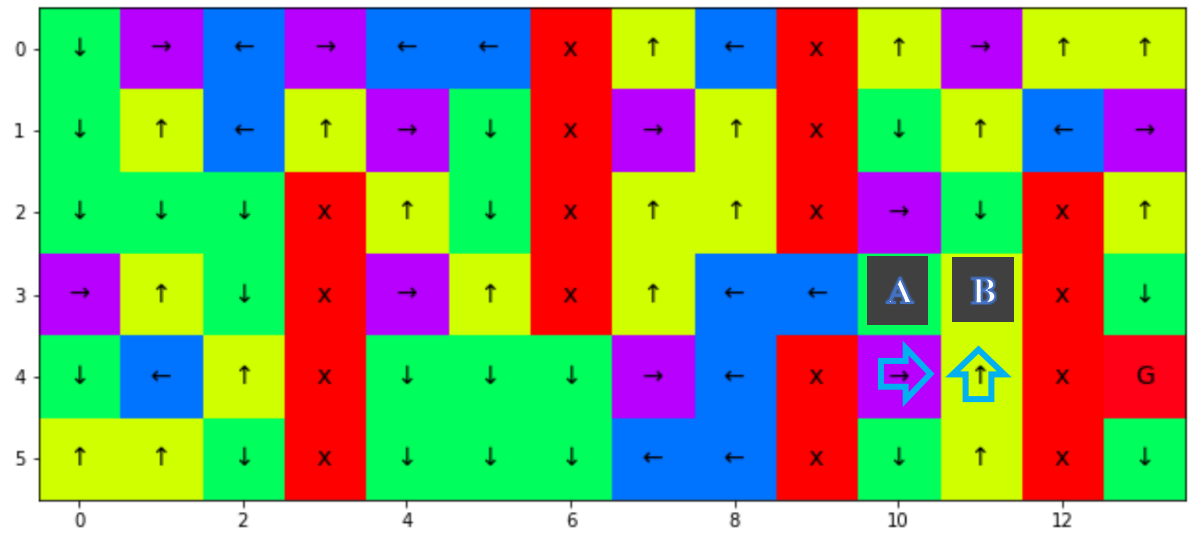
Agent (3,8) move → to Box (3,9)



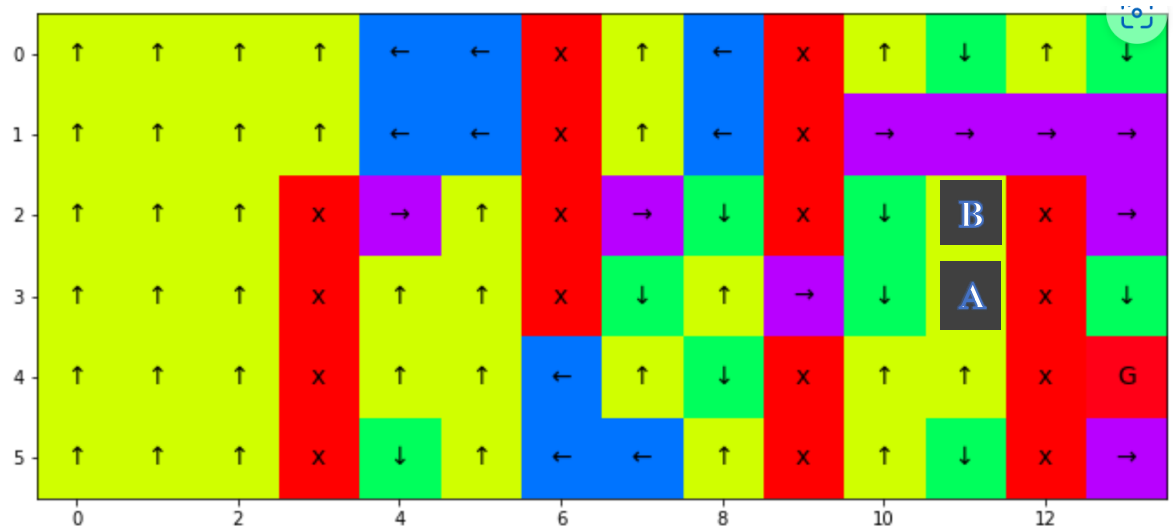
Agent (3,9) move → to Box (3,10)



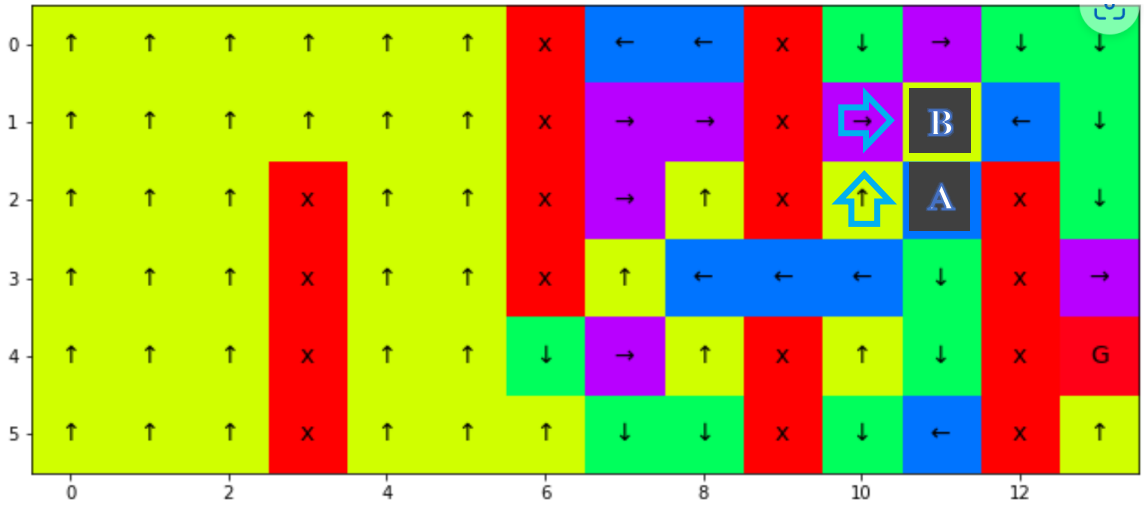
Agent (3,10) move ↓ to (4,10) then move → to (4,11) then move ↑ to Box (3,11)



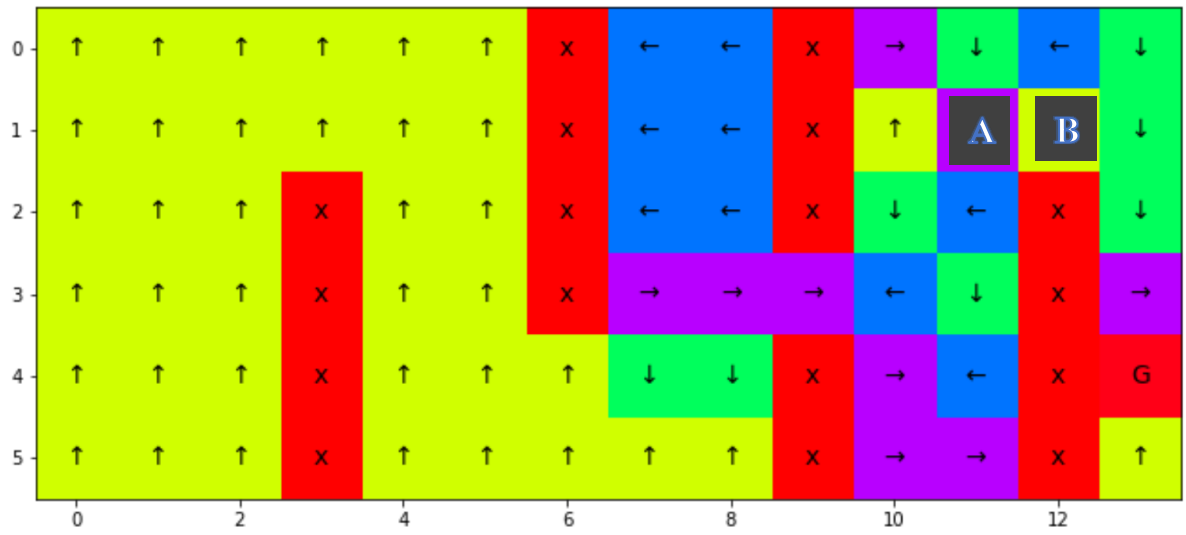
Agent (3,11) move ↑ to Box (2,11)



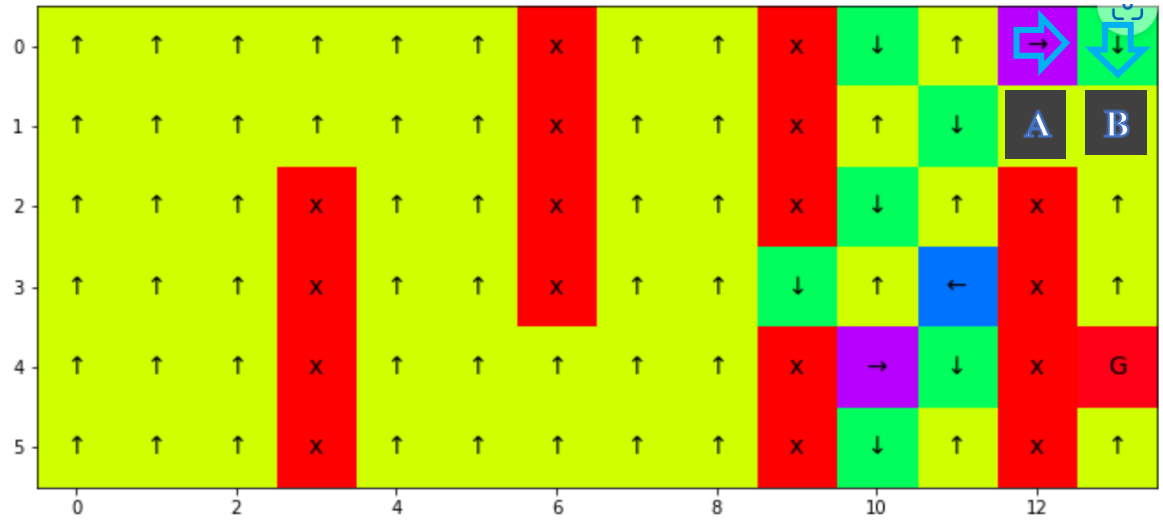
Agent (2 ,11) move ← to (2,10) then ↑ to (1,10) then → to Box (1 ,11)



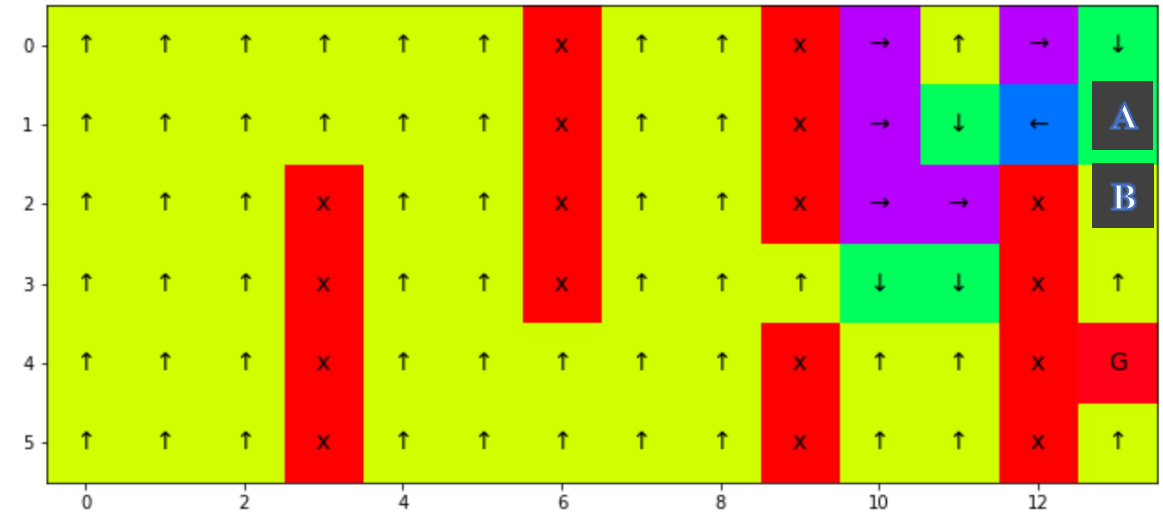
Agent (1 ,11) move → to Box (1,12)



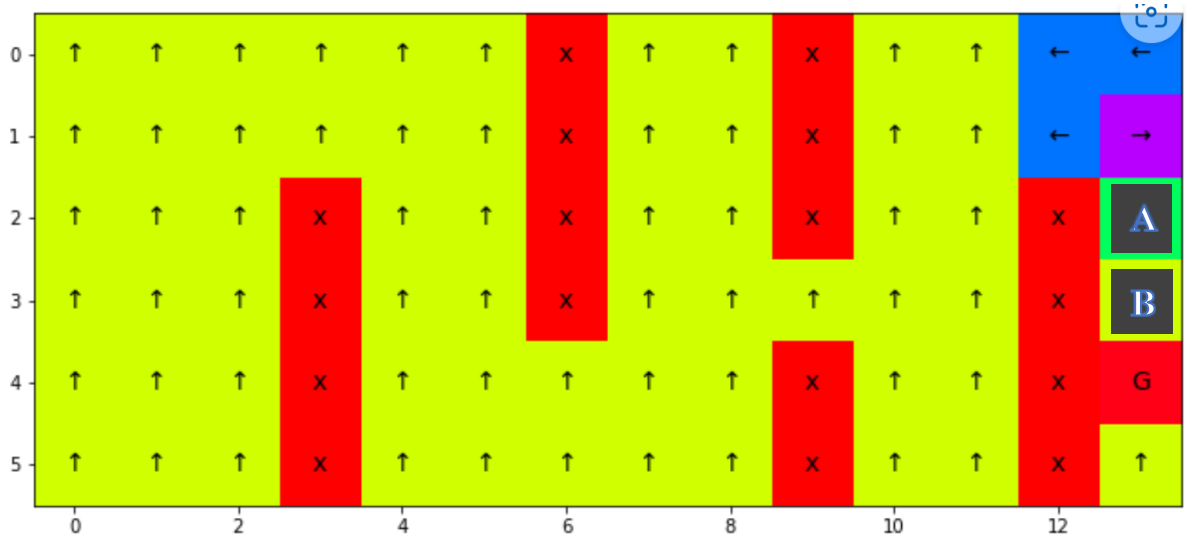
Agent (1 ,12) move ↑ to (0,12) then → to (0,13) then ↓ to Box (1,13)



Agent (1 ,13) move ↓ to Box (2,13)

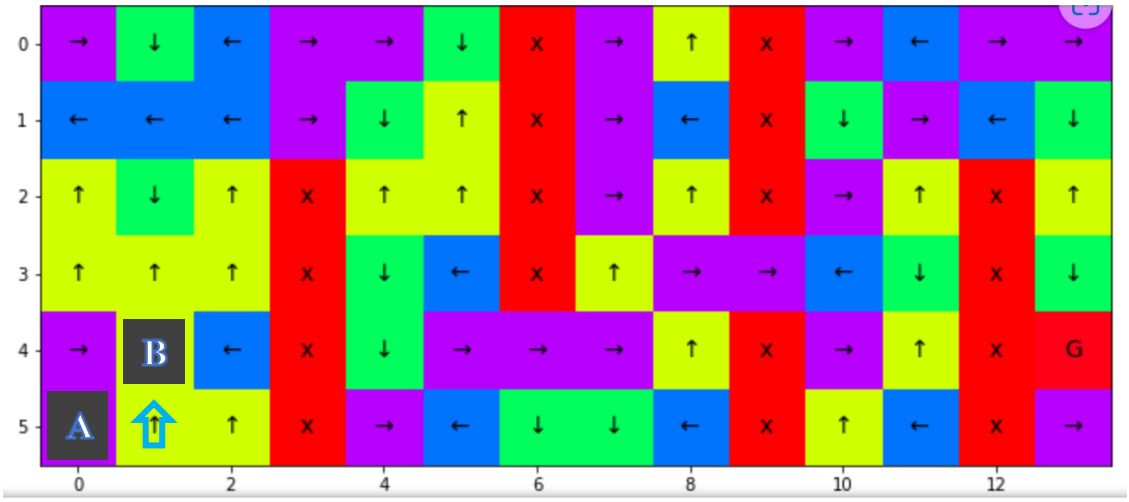


Agent (2 ,13) move ↓ to Box (3,13)

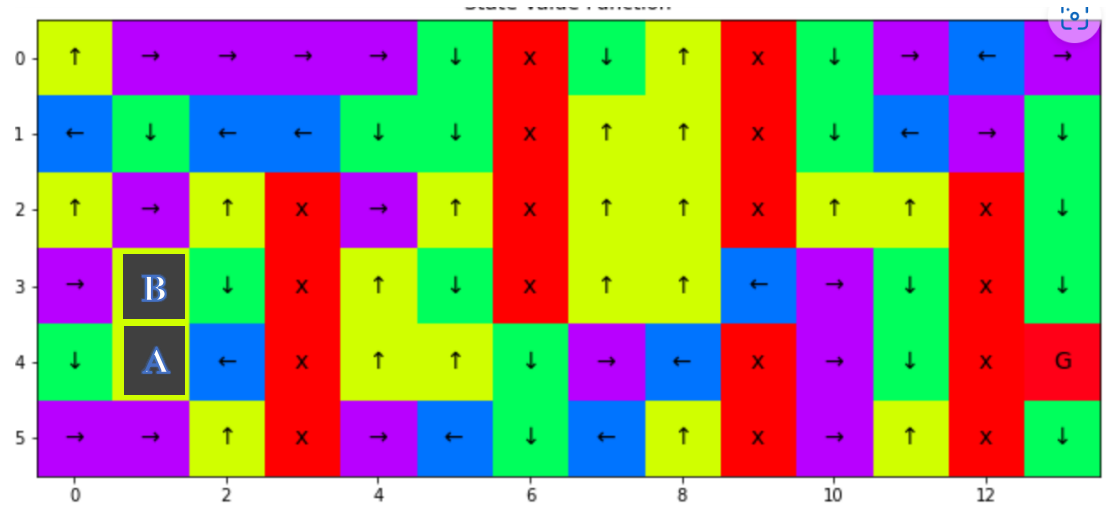


ANNEX B SARSA (Step-by-step)

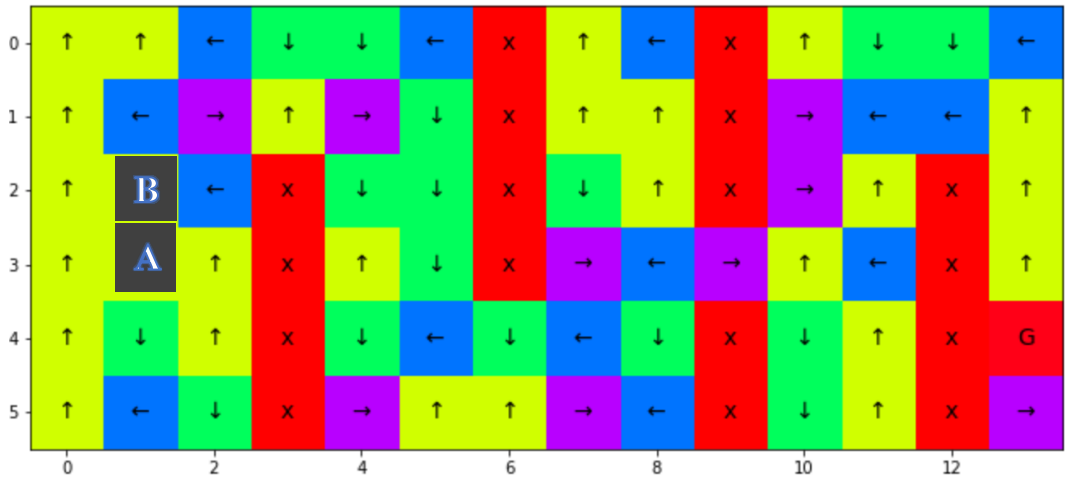
Agent (5,0) move → to (5,1) then move ↑ to Box (4,1)



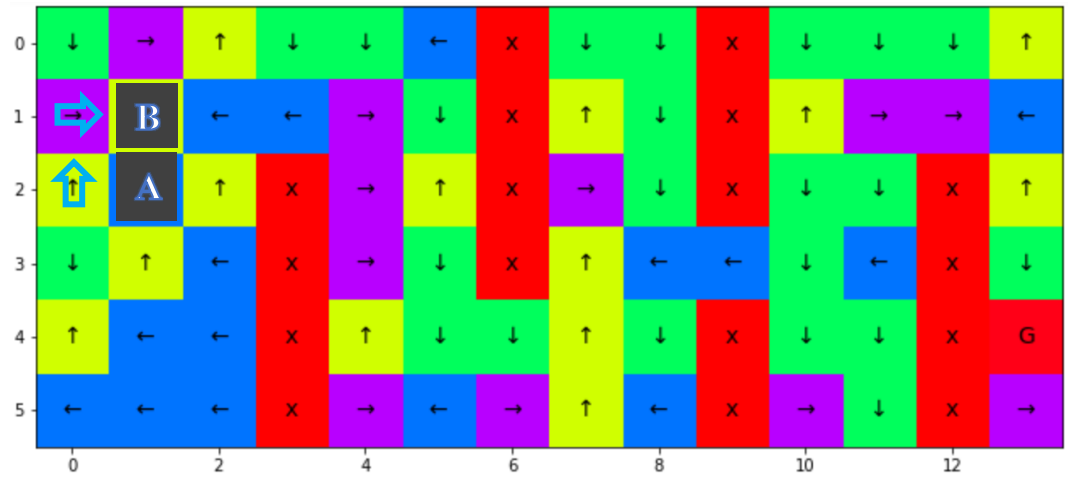
Agent (4,1) move ↑ to Box (3,1)



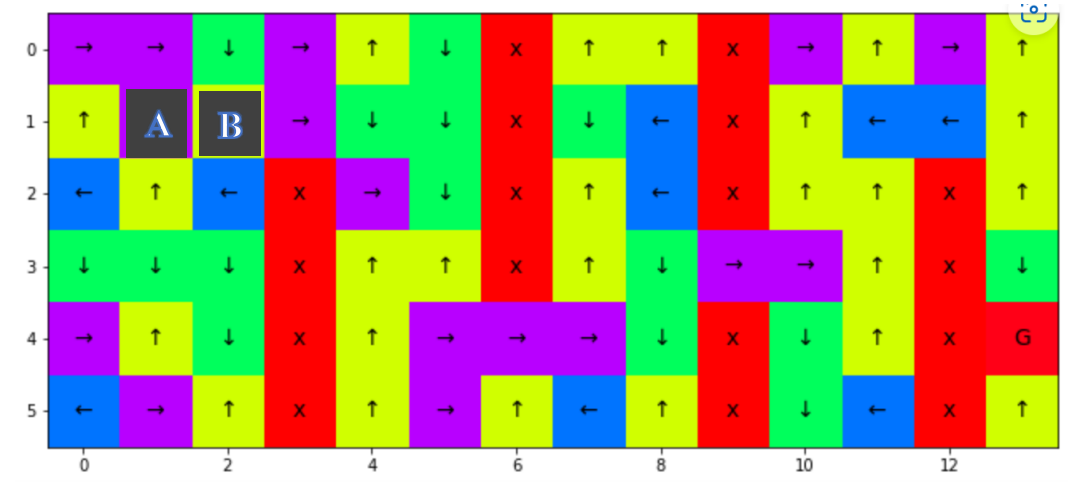
Agent (3,1) move ↑ to Box (2,1)



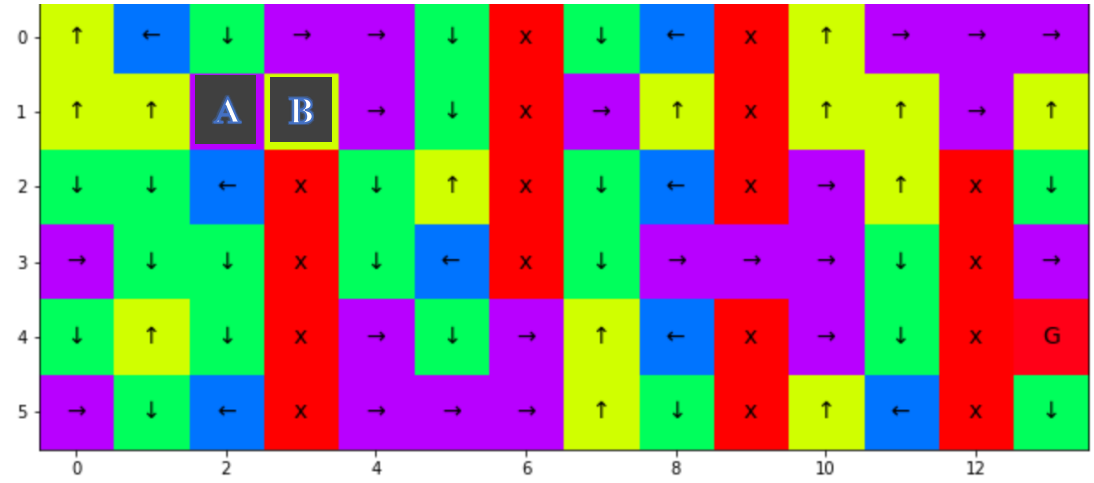
Agent (2,1) move ← to (2,0) then ↑ to (1,0) then move → to Box (1,1)



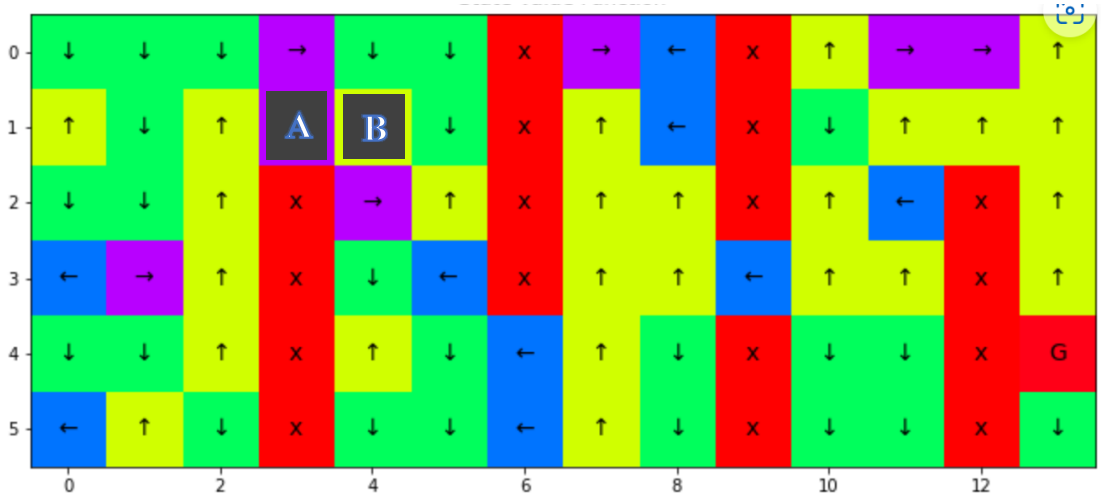
Agent (1,1) move → to Box (1,2)



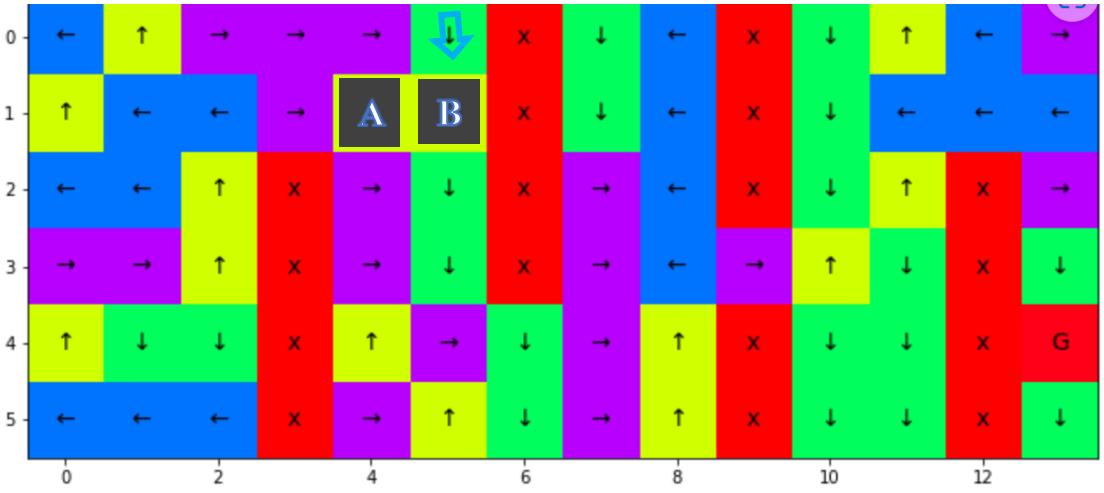
Agent (1,2) move → to Box (1,3)



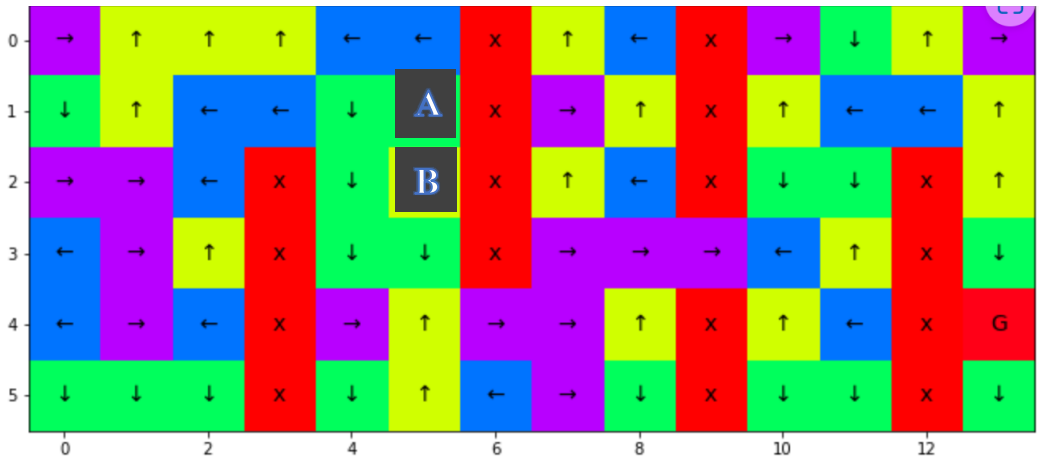
Agent (1,3) move → to Box (1,4)



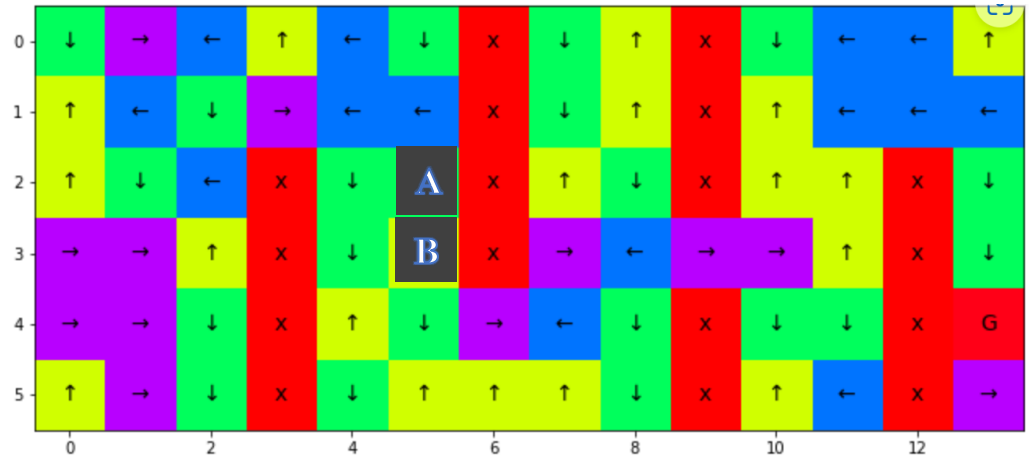
Agent (1,4) move ↑ to (0,4) then move → to (0,5) then move ↓ to Box (1,5)



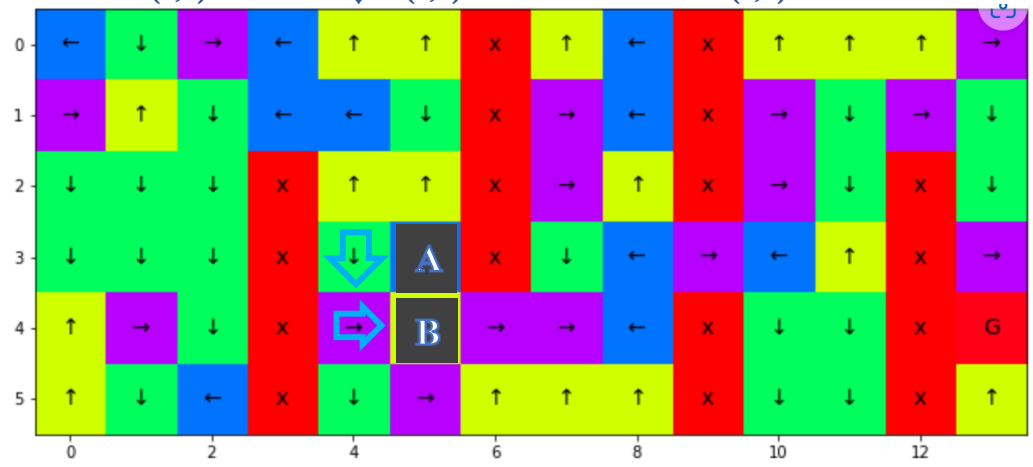
Agent (1,5) move ↓ to Box (2,5)



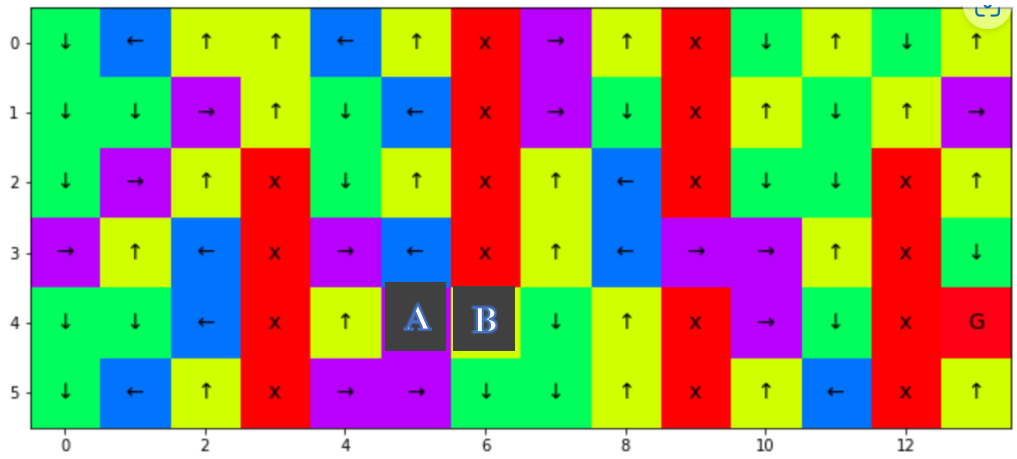
Agent (2,5) move ↓ to Box (3,5)



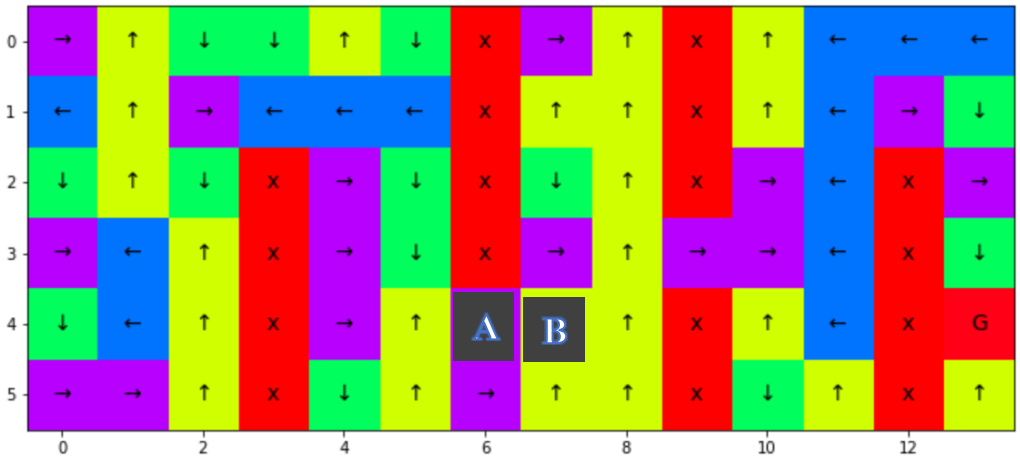
Agent (3,5) move ← to (3,4) then move ↓ to (4,4) then move → to Box (4,5)



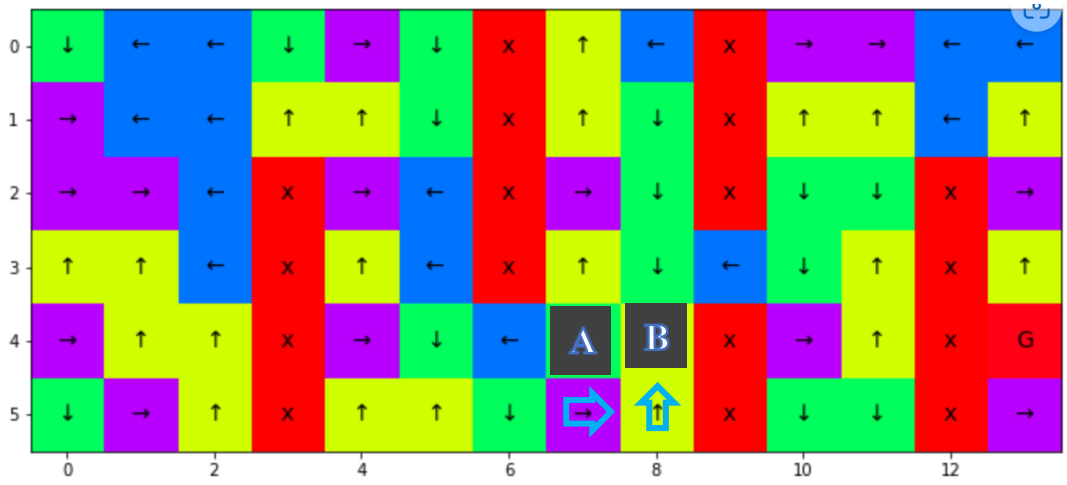
Agent (4,5) move → to Box (4,6)



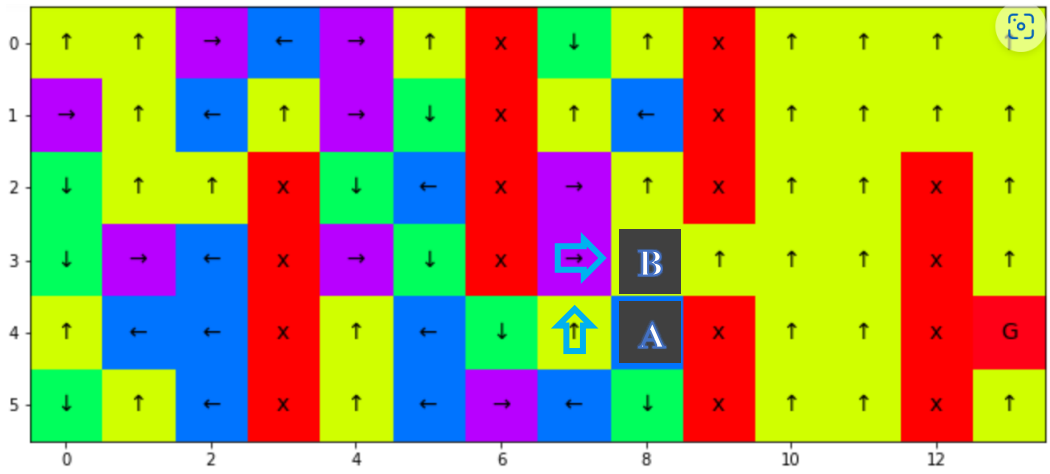
Agent (4,6) move → to Box (4,7)



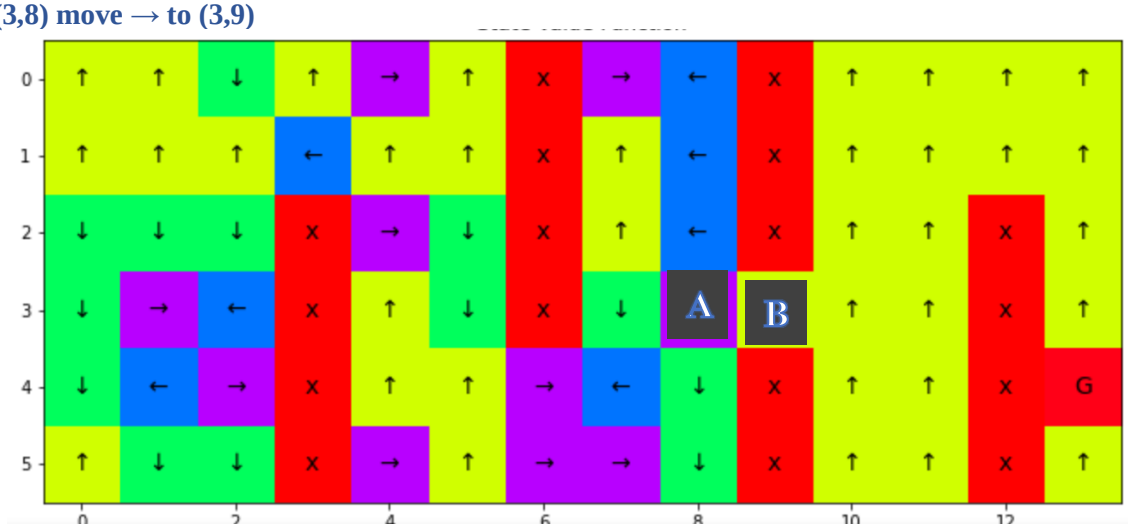
Agent (4,7) move ↓ to (5,7) then move → to (5,8) the move ↑ to Box (4,8)



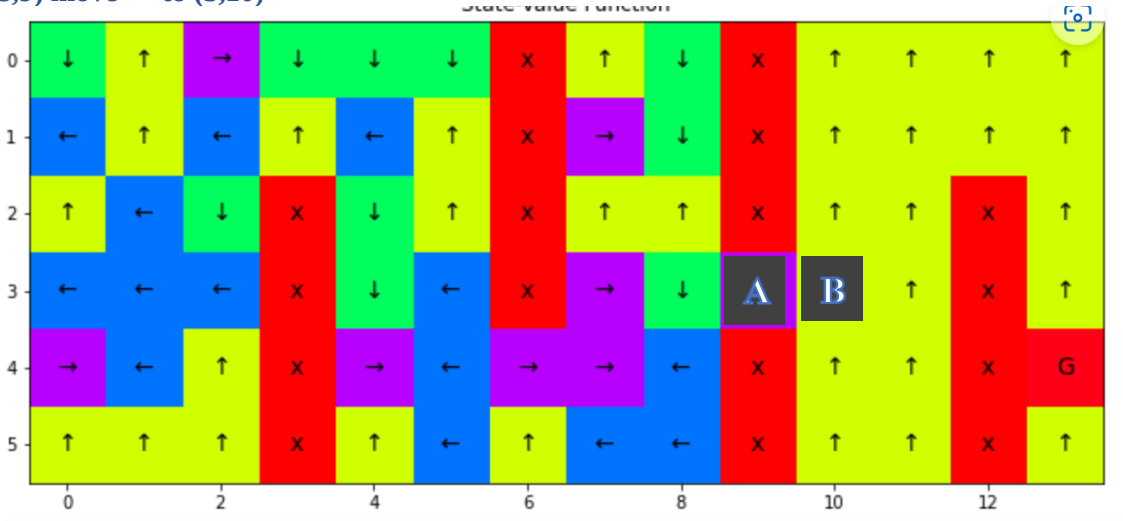
Agent (4,8) move ← to (4,7) then move ↑ to (3,7) then move → to Box (3,8)



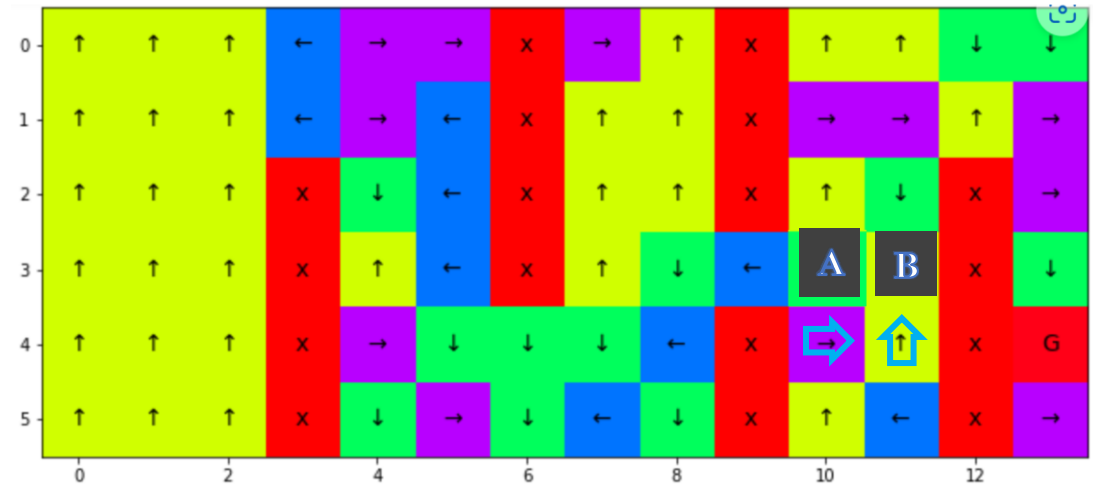
Agent (3,8) move → to (3,9)



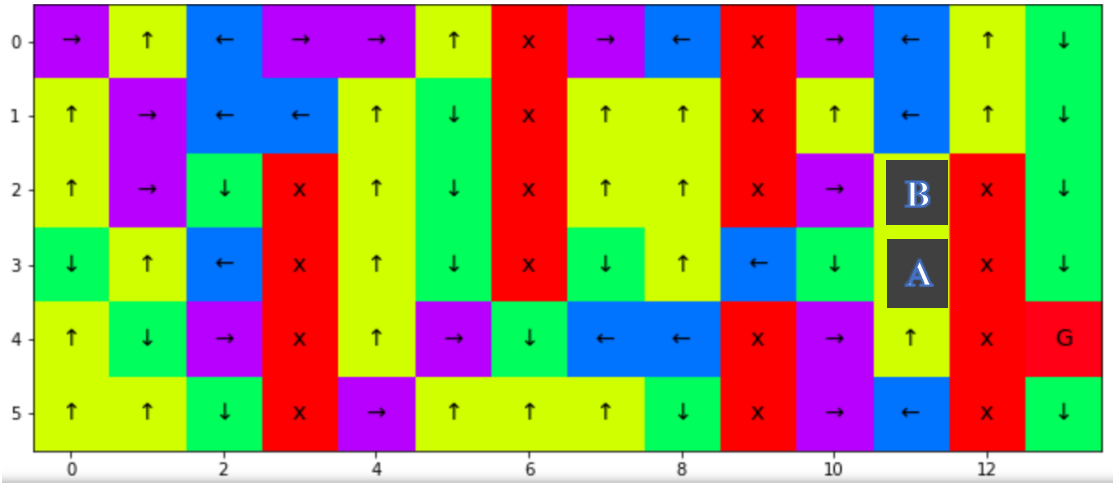
Agent (3,9) move → to (3,10)



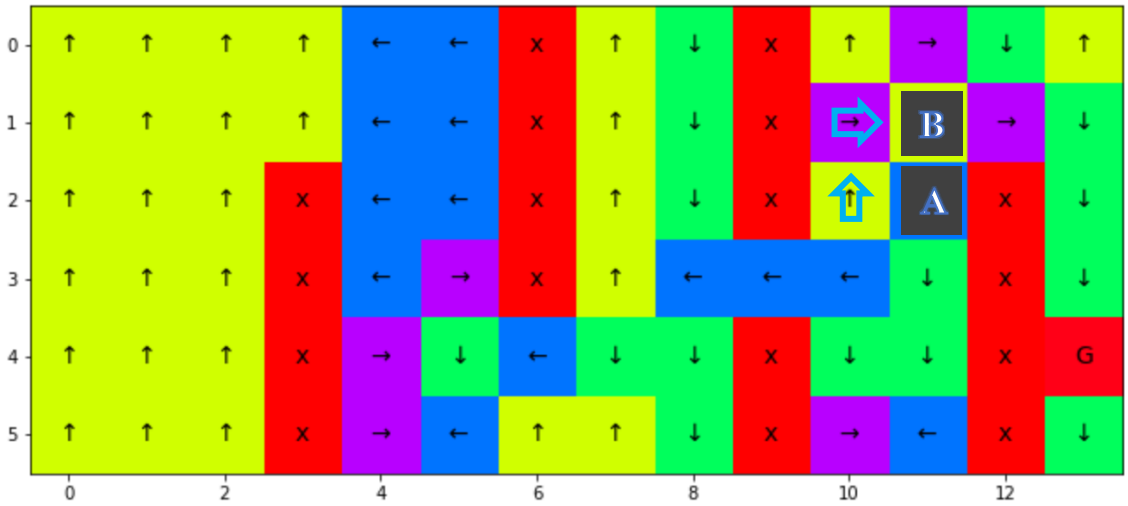
Agent (3,10) move ↓ to (4,7) then move → to (4,11) then move ↑ to Box (3,11)



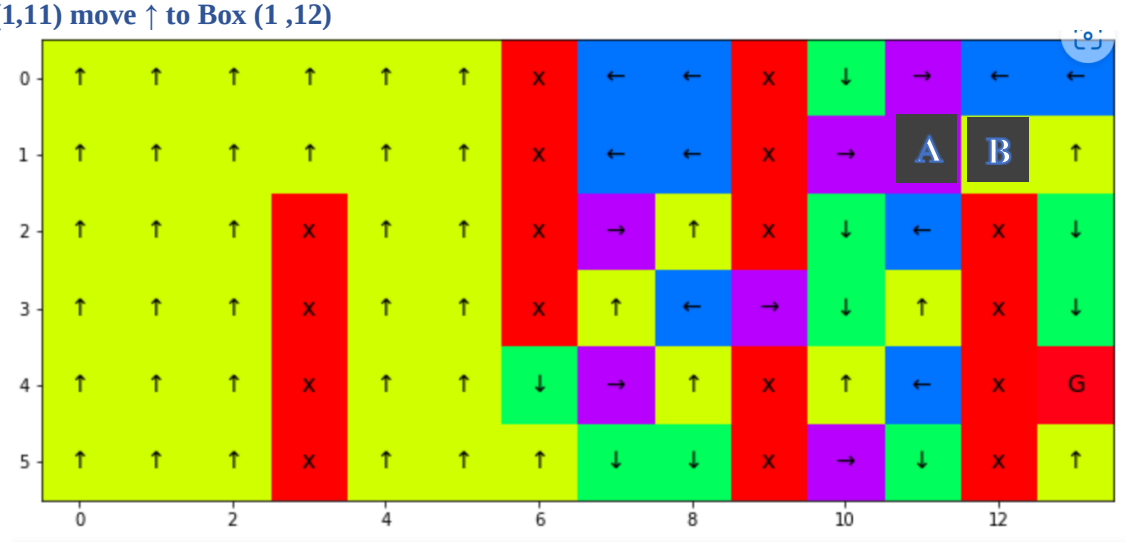
Agent (3,11) move ↑ to Box (2,11)



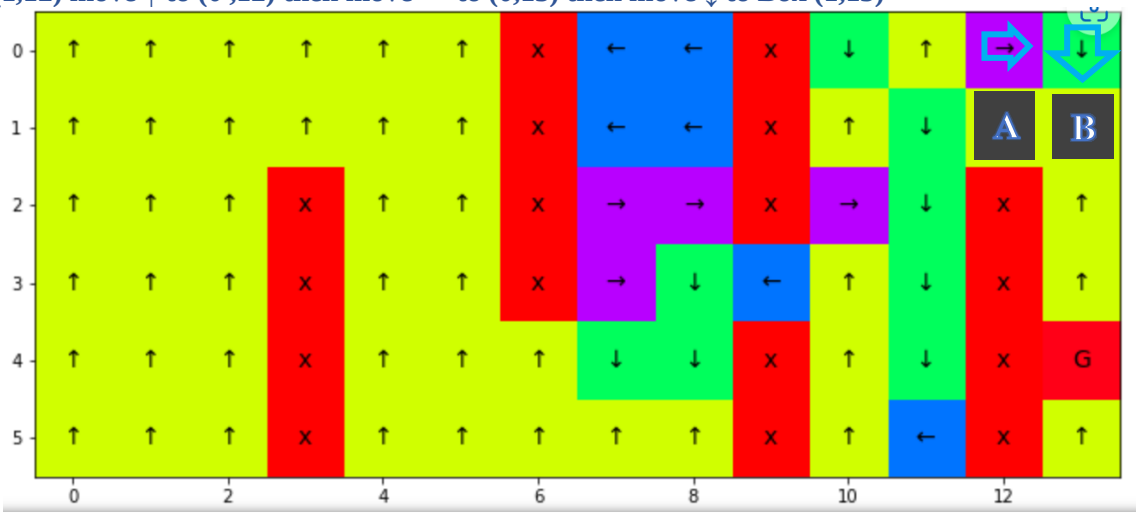
Agent (2,11) move ← to (2,10) then move ↑ to (1,10) then move → to Box (1,11)



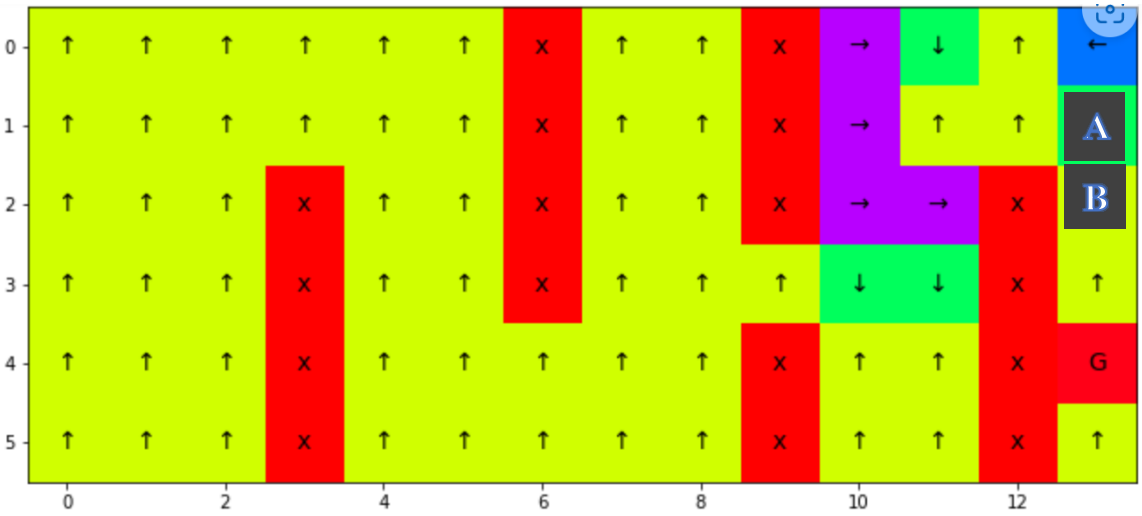
Agent (1,11) move ↑ to Box (1 ,12)



Agent (1,12) move ↑ to (0 ,12) then move → to (0,13) then move ↓ to Box (1,13)



Agent (1,13) move ↓ to Box (2,13)



Agent (2,13) move ↓ to Box (3,13)