Markov Decision Process

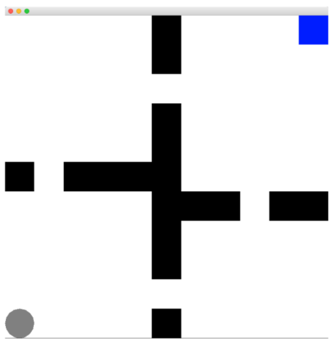
Xujian Liang, GTID: 903421366

1. Abstract

In this assignment, I focus on two different Markov Decision Processes(MDPs), Oceanic Glider Retrieve Process and Maze. I also implement Value Iteration, Policy Iteration and Q-Learning to these two MDPS.

2. MDP introduction.

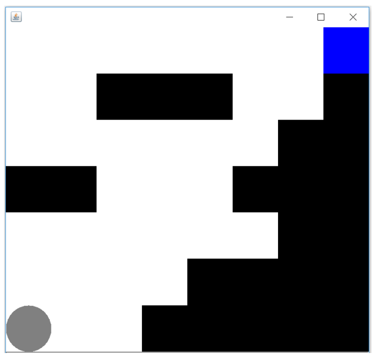
Two different MDPs will be focused for this experiment. According to the BURLAP, the orginal world is a grid world. Agent will start at the leftbottom corner and target will be positioned at righttop conrner. All the obstacles will be repseneted as black grids. The goal in this grid world is try to move the agent to our target. The movement of the ball will be limited in grid's movement. And the reward of each grid except our target may be negative like -1 and our target will have a large and positive reward such as 100. Also, we need to take the movement probability into this world. And I plan to do 1000 iterations for each experiments.



Grid World

2.1.Oceanic Glider Retrieve Process.

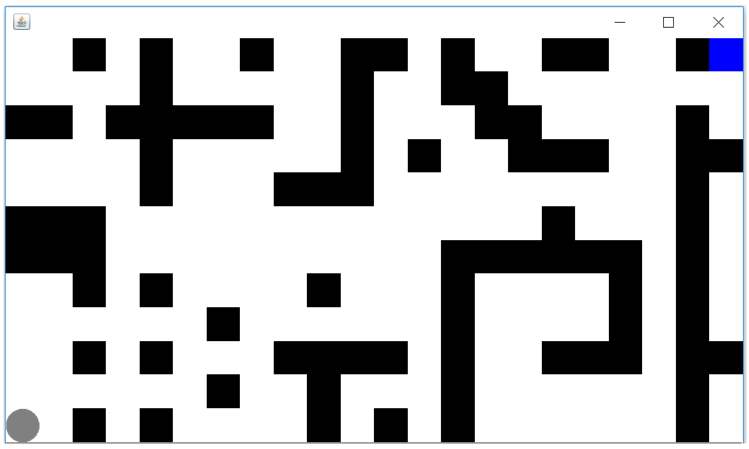
In this part, we will use the example of underwater gliders to do my first MDP. Gliders are controlled to travel follow the grid world assumptions. We will control the gliders to move vertically to void obstacles to get the bottom of the ocean. [1]



Simulating oceanic glider retrieve process.

2.2. Maze

In the maze, an agent has to find a way start at the birth place to the target without hitting the any obstacles. This problem is interesting because we have lots of common situation in the real life like maze game and rescue mission in the mountains. In my experiment, the grids have a size of 12 X 22 and the cost function is also -1 for each step. The target reward is 100.



Maze simulating

3. Reinforcement Learning Algorithm

3.1. Value Iteration

The essential idea behind is that if we knew the true value of each state, it will be easy for us to make a decision and we just choose the action to maximize utility. Here, the true value of a state is the immediate reward for that state plus the expected discounted reward if the agent acted optimally form that point on.[2]

3.2. Policy Iteration

Compared with value-iteration algorithm which keeps improving the value function at each iteration until the value-function converges and that will lead the agent only cares about the finding the optimal policy and sometimes miss the optimal policy, policy-iteration at first create a random policy, then compute the utility based on this policy and update state unities and then use the updated information to select the best policy until convergence. [2]

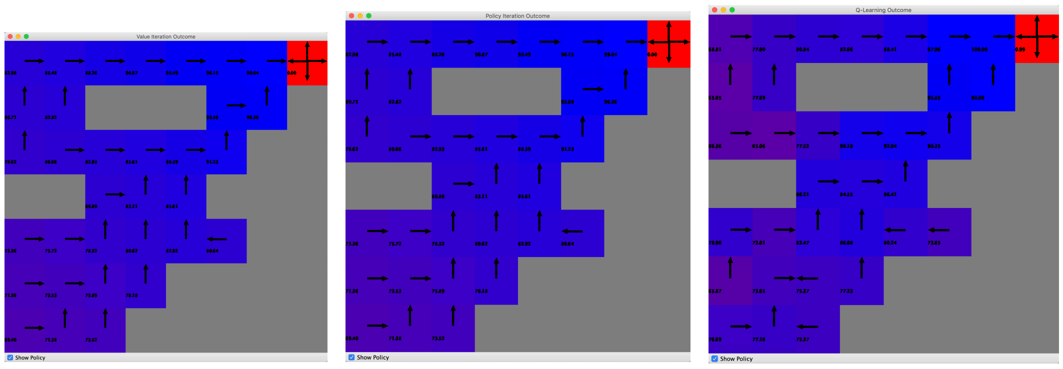
3.3.Q-Learning

Value Iteration and Policy Iteration work well for finding out the optimal policy, but both of them assume that the agent will have a great deal of domain knowledge. It can be found at
QL first assigns a Q-value to every state, for example, in my experiment I will assign very low epsilon at section 4 to show the situation when little is known about the world. Later in its life, the agent may want to almost always choose the action that looks best; there is little information to be gained, and so the value of learning is negligible. We will model this with a function that assigns a probability of being chosen for each possible action in a given state. This function should tend to choose actions with higher Q values, but should sometimes select lower Q-value actions. The probability of selecting the highest Q-value action should increase over time.[2]

4. Experiments

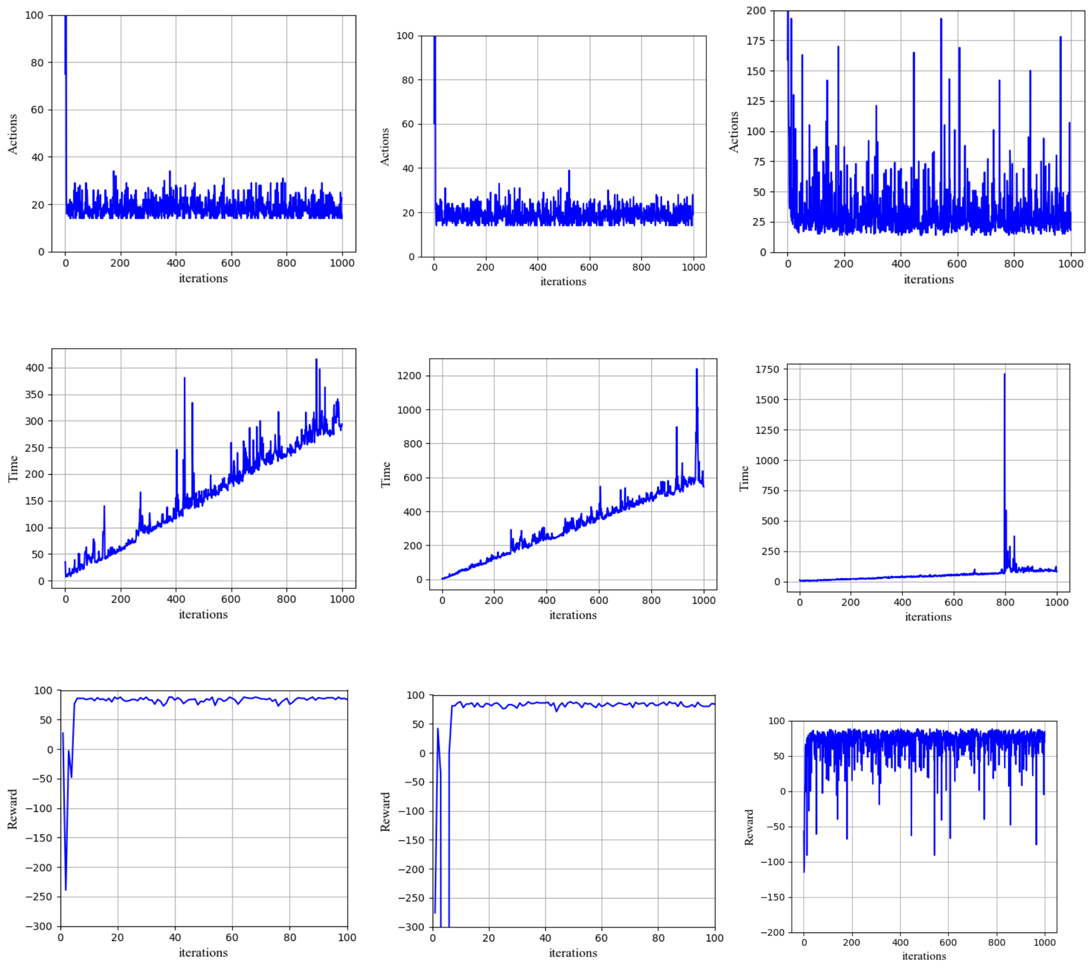
4.1.Glider retrieve process.

Results for three different algorithms are shown as below.



VI result(left), PI results(mid), QL result(right)

For the final action results, we could find out that VI and PI get quite similar result and QL is kind of different according to the value. It's probably due to the simplicity of the game world. For QL, it may be due to that QL want to explore the world and try to maximize the final rewards and it will have some probability distribution to have a random walk. Below the results of number of actions, iteration time and rewards according to the iteration number for three different algorithms. Obviously, PI and VI have almost similar results as the action figure.

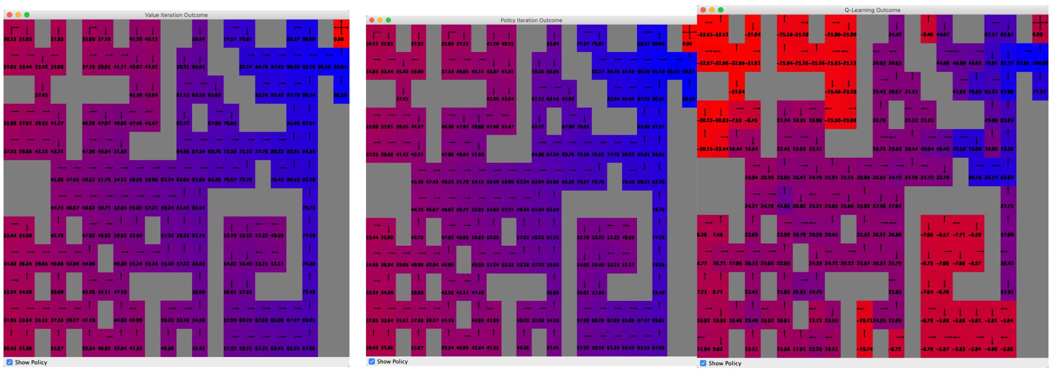


VI result(left), PI results(mid), QL result(right)

For the action part, we can see that it will be easy for PI and VI to decrease to 20 and keep stable. However for the QL, we get lots of spikes and I think it's due to the fact that QL is kind of instablizing algorithm which need to turn to the global accumulating rewards. For the time part, in the most of time QL is much faster than VI, and VI faster than PI. I think it's due to that QL will be very easy to calculate when we are in a small world, and VI just need to be greedy for each action so it will be fast also. However, PI need to rethink policy may lead to consume lots of time. For the reward, due to the simplicity, all algorithm will reduce to about 80. But QL has lots of spike toward negative. And it quite makes sense since QL has a random selection probability to lead QL to become instability.

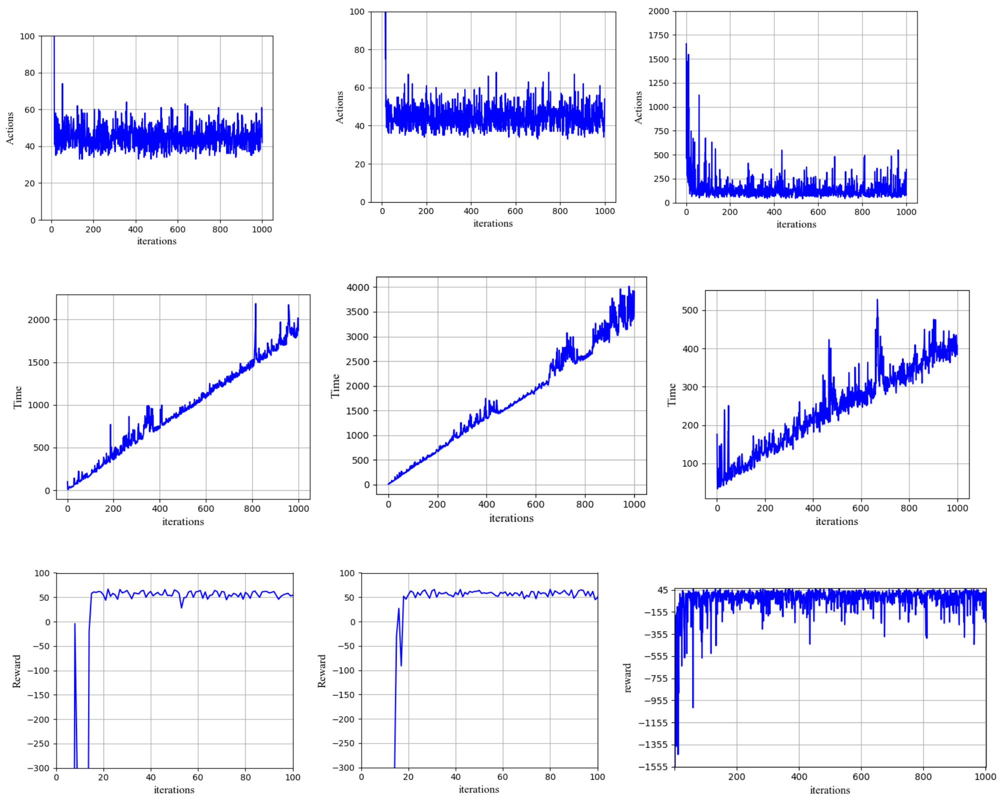
4.2.Maze

The VI, PI and QL results of the Maze case are presented below.



VI result(left), PI results(mid), QL result(right)

Again, VI and PI get similar results and QL has different final results. We can find out that on the lefttop and rightbottom coners, we get low Q-values. The reason may be that QL cannot work very well for corner area where agent's movement get limited.



VI result(left), PI results(mid), QL result(right)

For VI and PI, we get quite similar results. For action figures, we find out that VI and PI finally decrease to the level of 50 but with small spikes. Therefore, we find out that VI and PI are still reduct to a stabilizing model. However, the action number for QL decrease but still with big spikes, which shows the model is not stabilizing. And for the time part, we can see that QL take less time. I think it's partialy because QL can easier to explore a large area and computation takes less time. The VI and PI, QL converge to around 50, while GL are with lots of negative spkes. It's because that QL is the algorithms that need to focus on the global rewards and it's instabilizing.

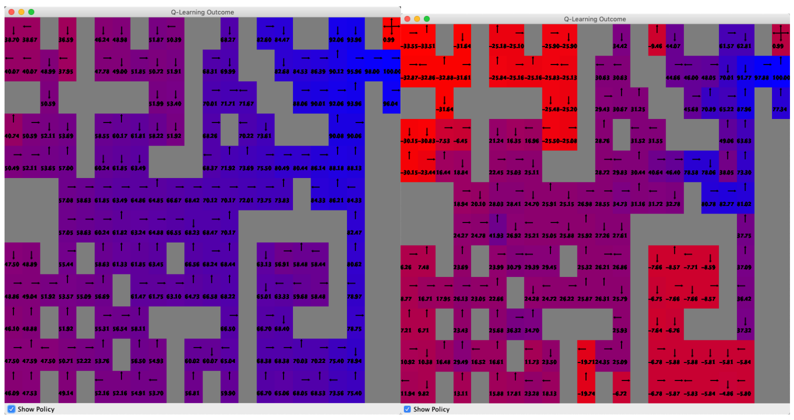
5. Change the strategy.

In the above section, for the QL part, we use epsilon-greedy strategy. According to BURLAP, the eplison means the probability for the policy to return a random action and 1- epsilon probability to return greedy action. What's the difference? This idea gives our chance to think about two different policy, one random, another greedy. Maybe random one can do better in the future, but now, greedy one can do the best work.

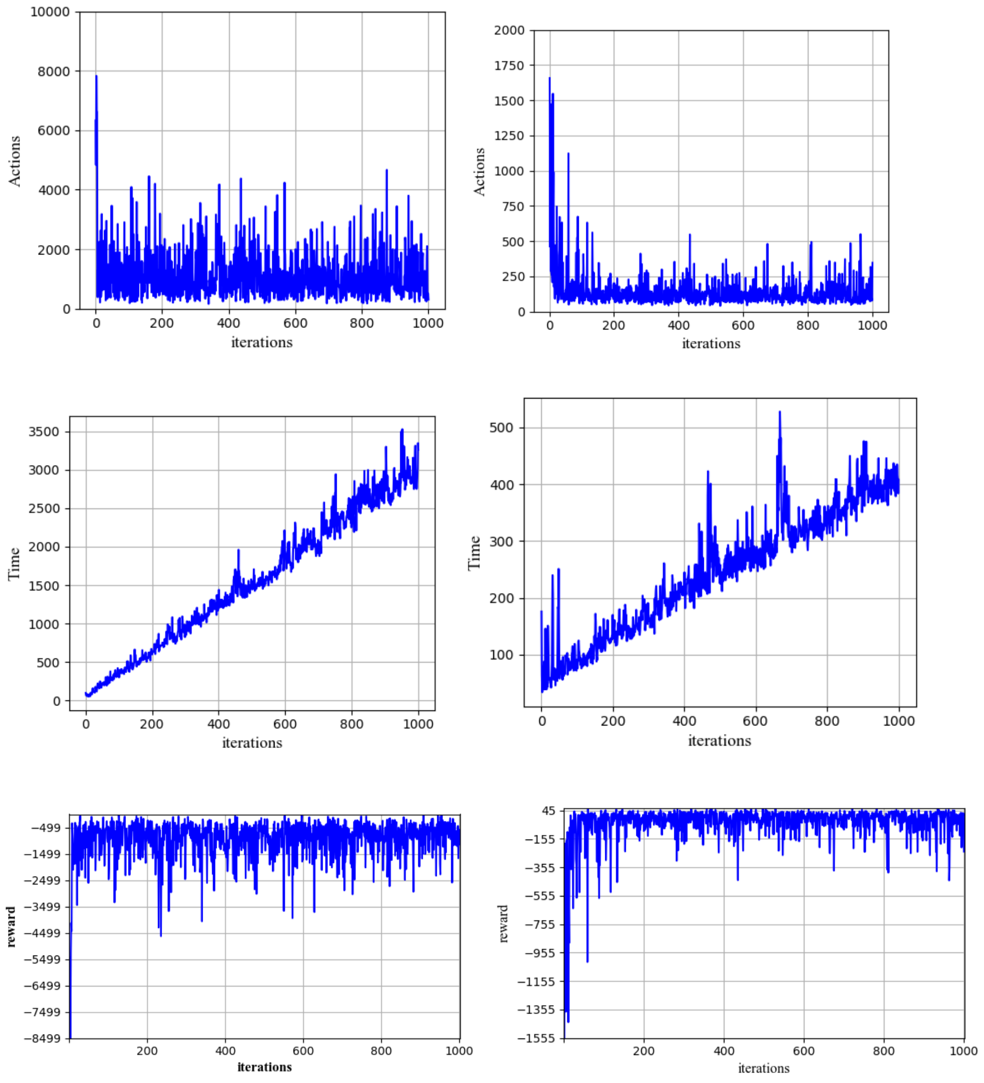
The Boltzmann strategy will weigh the Q-value for each action and chooses the action with this weighted probability in the following moves. And this strategy has a parameter named temperature which controls how greedy the Boltzmann distribution is. By control the temperature, we can easily get find out the different results accompanied by different temperature.

5.1.EpsilonGreedy Strategy

Now, time to focus on the Epsilon-Greedy strategy with epsilon value 0.9 and 0.1. The results are presented below:



QL EpsilonGreedy with epsilon = 0.9 (left) and 0.1 (right).

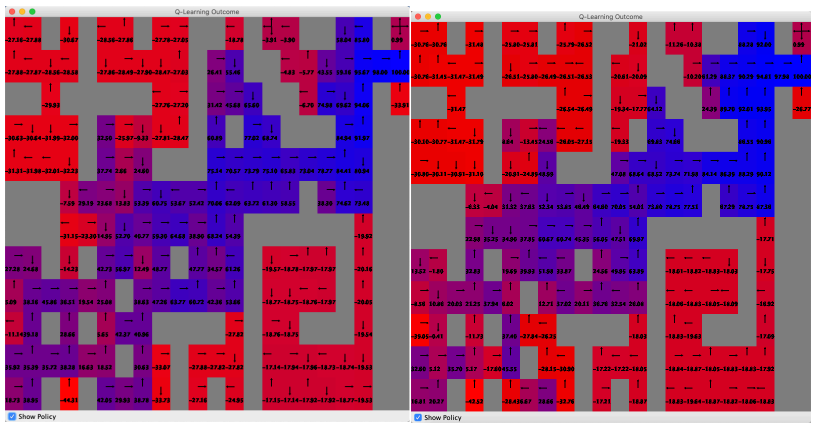


QL EpsilonGreedy with epsilon = 0.9 (left) and 0.1 (right).

The results are quite different according to different epsilon, especially in the Q-value distribution. The one with epsilon = 0.1, we get low Q-value in the two corner areas, while the one with epsilon = 0.9, we get higher Q-values in these two areas, which is kind of misleading and cannot find the correct way out. I think that a high epsilon value may lead algorithm to do more random work and explore unfamiliar areas and sometimes lead to the wrong way. Due to the same reason, the algorithm will need more action, time during the iteration and rewards become also low due to the extra explore actions. Therefore, in this case, it's better for us to choose lower epsilon value to handle with maze problem.

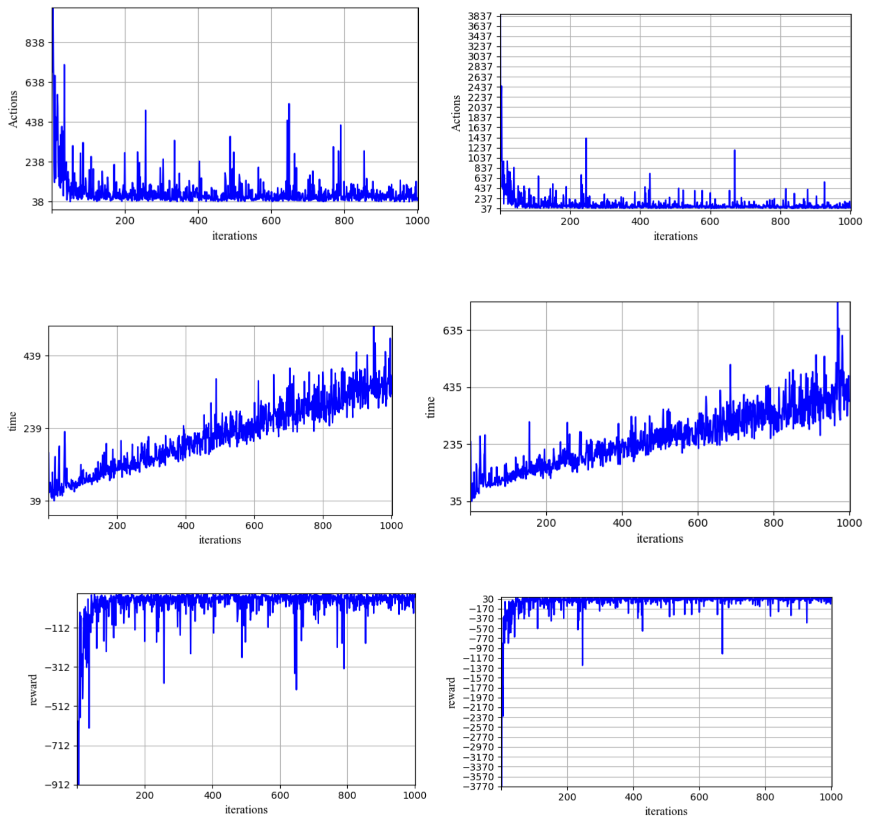
5.2. Boltzmann Strategy

Pay attention to the Boltzmann Strategy with "temperature" value of 0.1 and 0.9. The results are presented below.



Boltzmann results with temperature = 0.1 (left) and temperature = 0.9 (right).

The maps show that final results are quite similar with both temperature = 0.1 and 0.9. We also find out that the corner have low values means low activity area, indicating the algorithm doesn't learn much in those area. Also, both of them could find the best path to the target. No big difference may due to the reason that this maze has very low energy differences.



Temperature = 0.1(left), 0.9(right)

With temperature = 0.9, the algorithm needs more action and time compared with 0.1. The reason behind it is that the temperature value will decreasein a annealing way. With a high temperature, the algorithm will need more time to decrease. The rewards also decrease with high temperature value. In a summary, lower temperatures perform better for maze problems.

5.3. Comparison for two different Strategies for the results.

	Boltzmann	Epsilon
Commons	Lower values means lower action and time Both show the instability of QL which related to QL Both shows linearity for the time.	
Differents(idea)	weigh the Q-value for each action and chooses the action with this weighted probability according to the Boltzmann distribution in the following moves	when we choose the actions, we will take them into consideration that might have a higher reward in a later time by use eplsion probability to take random action into consideration.

6. Conclusions

In a summary, both VI, PI, QL can do a good job in the maze problem and glider problem. And we can find the difference betweem three algorithm. First, VI and PI are stablizing but QL cannot lead to stabilizing model. It’s because QL are trying to consider the overall reward and VI and PI are kind of more greedy.

QL has different strategies such as Epsilon and Boltzmann. There still exists other strategies which are not contained in this experiment. Also, for the maze problem, the temperature for Boltzmann cannot do much change. But the change of epsilon can make a big difference. I think it’s part of this maze has very low energy differences and lead to no change about distribution after we change the temperature.

7. Reference

[1] “Under water” Wikipedia https://en.wikipedia.org/wiki/Underwater_glider.

[2] “Value Iteration, Policy Iteration and Q-Learning” University of Hartford

http://uhaweb.hartford.edu/compsci/ccli/projects/QLearning.pdf Accessed 6 April 2009.