Markov decision process

FrozenLake-v0 is one of the problem i chose. This is a grid world problem and i am using it as a small (number of states) problem which is 16 (4x4) in my case. Let's take an example of amazon warehouse where robots are used to navigate across to pick a package from one location and drop to another location. Robot's job is to find a path from source to destination avoiding obstacles in between (wall). This is a real world usecase and very much relatable to frozen lake problem. This is why i think it is very interesting. The second problem i chose is forest management. This is a non-grid problem and i am using it as large (number of states) problem. It has two possible actions (0: Wait, 1: Cut). "Forests are increasingly threatened by windthrow" [1]. Such natural calamity can damage trees which are in good shape. This makes it very important to manage forests by cutting down trees which are prone to windthrow and retaining trees which are not. Hence this problem is interesting.

FrozenLake

Let's see FrozenLake problem first. It has starting point, frozen surface, hole and goal point, except hole every other location is safe. Agent has to find a walkable path from starting point to goal point. Ice is slippery so you don't always move in the direction you intend to. There are four possible actions (0: left, 1: right, 2: down, 3: up), diagonal moves are not allowed. I have used FrozenLake-v0 from gym [2] and generated P,R matrix using env.P. Later i have used pymdptoolbox [3] to run PolicyIteration, ValueIteration and Qlearning algorithms. Gamma (discount factor) value used is 0.8 for all three algorithms.

Policy Iteration and Value Iteration

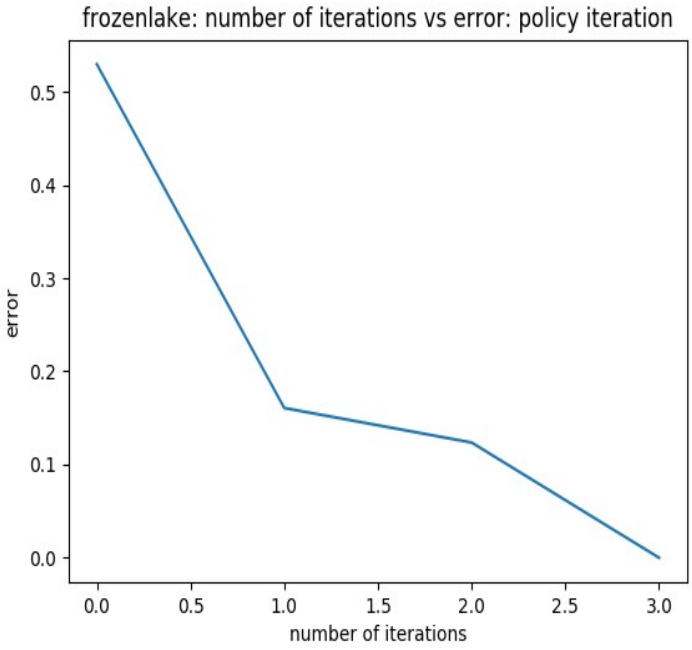

Fig 1. No of iterations to converge: PI

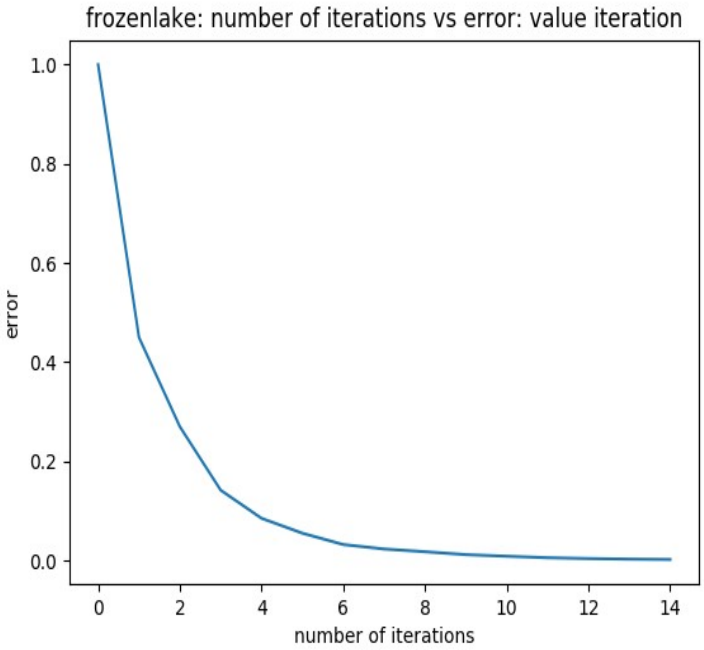

Fig 2. No of iterations to converge: VI

How many iterations does it take to converge ? Looking at above graphs (Fig1 and Fig2), we can see that policy iteration takes 3 iterations while value iteration takes 14 iteration to converge. “Since the agent only cares about the optimal policy, instead of repeatedly improving value function estimate, it will redefine the policy at each step and compute the value according to this new policy until policy converges” [4]. Hence Policy iterations takes few iterations to converge compared to value iterations.

How do you define convergence ? In case of policy iteration, we are defining convergence when next policy is same as current policy or when maximum number of iterations are reached. In case of value iteration, we are defining convergence when difference of current v value and previous v value is less then some threshold value or when maximum number of iterations are reached. In above graphs (Fig1 and Fig2) error on y-axis is convergence parameter what we just defined. Hence you can see that when error becomes zero we are saying algorithm has converged.

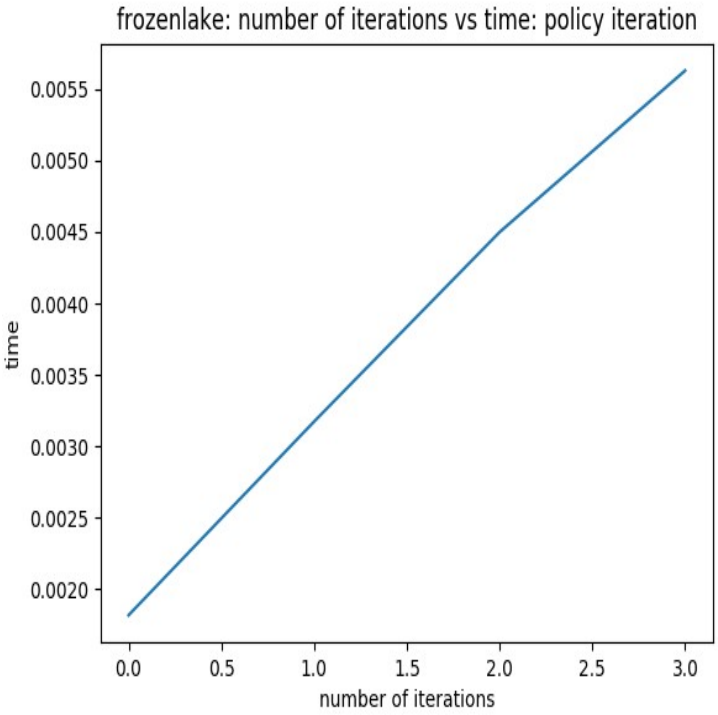

Fig 3. Convergence speed: PI

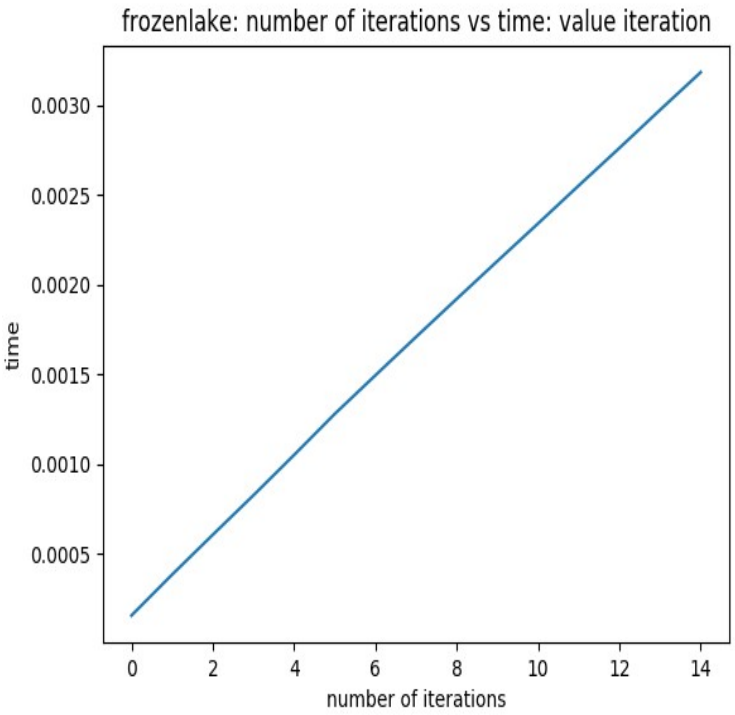

Fig 4. Convergence speed: VI

Which one converges faster ? Looking at above graphs (Fig3 and Fig4), we can see that value iteration converges faster (~2x) compared to policy iterations. Policy iteration algorithm is a combination of policy evaluation followed by policy improvement. “Each policy is guaranteed to be a strict improvement over previous one” [5]. Each policy evaluation itself is an iterative computation which evaluates value function for the previous policy. “This results in an increase in convergence speed of policy iteration presumably because the value function changes little from one policy to the next” [4]. Drawback of policy iteration is that each of it’s iterations involved policy evaluation which is an iterative

computation of value function. “Value iteration algorithm restricts it to just one sweep” [4]. It returns optimal v value using which algorithm extracts optimal policy corresponding to the optimal value. Hence value iteration is faster compared to policy iteration.

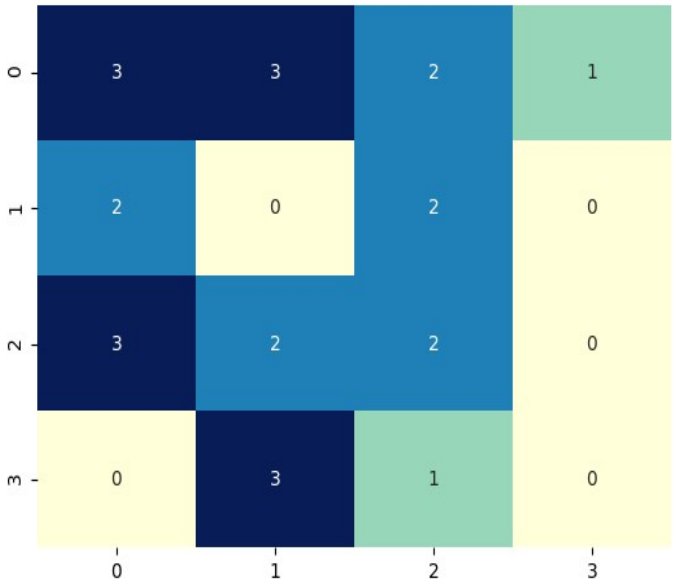

Fig 5. Optimal Policy iteration heatmap

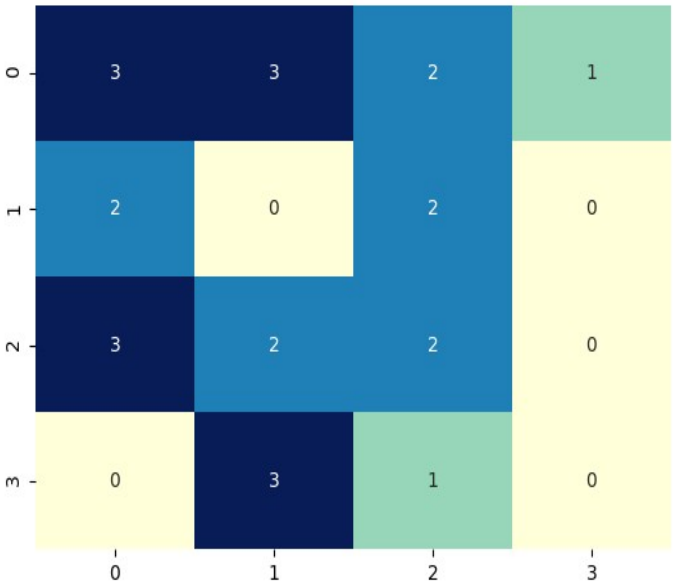

Fig 6. Optimal Value iteration heatmap

Do they converge to the same answer ? Yes, above two graphs (Fig5 and Fig6) depicts optimal policy heatmap with actions as annotations. Both the algorithm converges to the exact same policy. Since both VI and PI are guaranteed to converge to an optimal policy. Probably there exists only single optimal policy hence they converge to exact same policy.

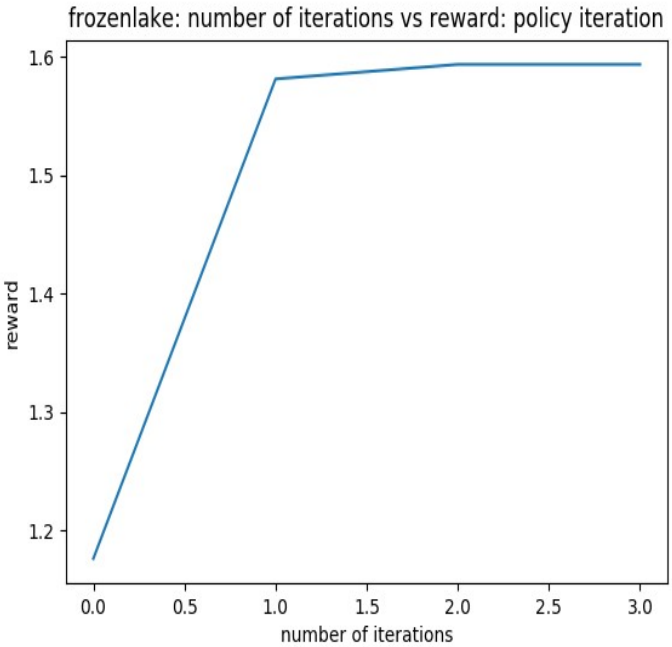

Fig 7. Policy iteration reward

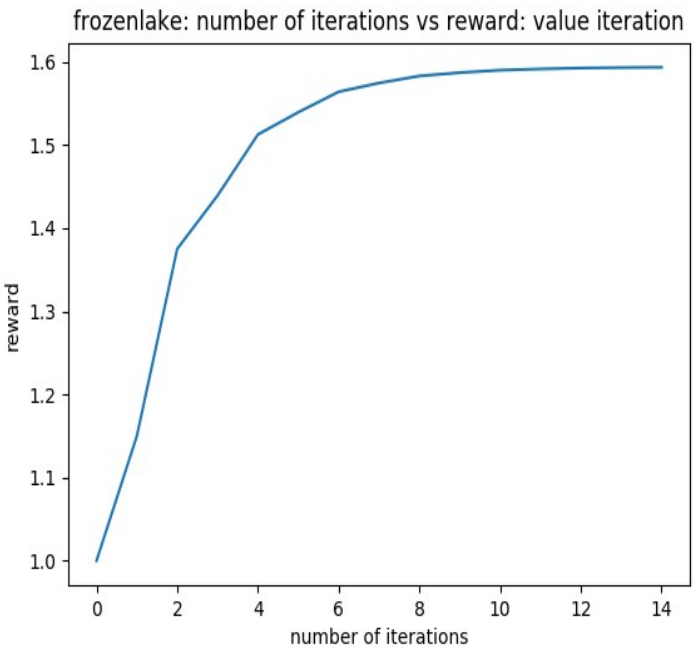

Fig 8. Value iteration reward

Both the algorithms (Fig7 and Fig8) earn same reward which is obvious as both converges to same policy.

QLearning

Here our agent doesn't know anything about the model. Agent has to figure out the best action based on q<state, action> values. Since agent doesn't have domain knowledge, q learning takes more iterations to learn and converge compared to VI/PI. I have used 50K as number of iterations. I tried varying alpha (learning rate), epsilon (exploration rate). Higher epsilon means more exploration meaning more random decisions. High learning rate means more weight to future samples compared to past. I am using gamma = 0.8 same as what i have used for VI/PI. Moreover i am not decaying alpha and epsilon for below experiments.

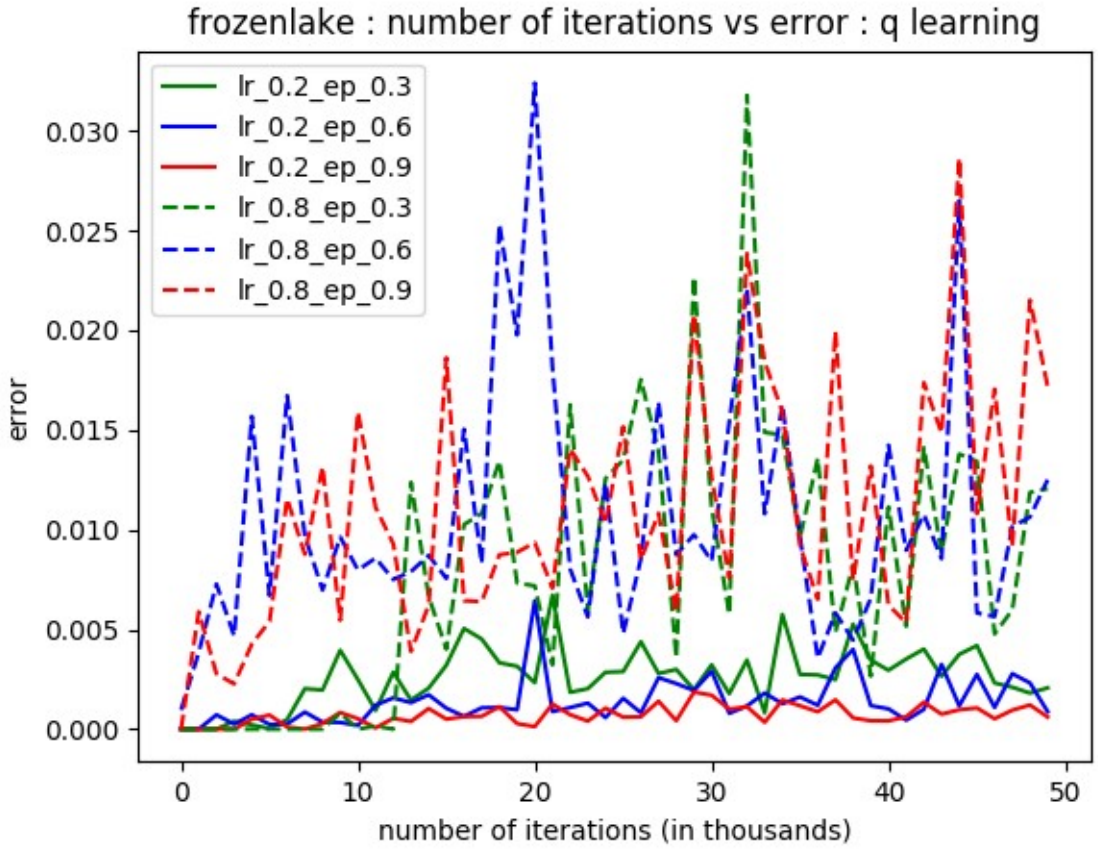

Fig 9. No of iterations to converge : QLearning

How do you define convergence ? We are defining convergence as difference of q values as given by equation **(**`dQ = self.alpha * (r + self.gamma * self.Q[s_new, :].max() - self.Q[s, a])`**)**. Error is our convergence term and we want it to be as low as possible.

Dotted line denotes high learning rate and solid line denotes low learning rates. Above graph (Fig 9) clearly shows that low learning rate (specially epsilon = 0.9) converges best among all others. Also above graph shows that qlearning takes huge number of iterations to converge compared to VI/PI because q learning doesn't know anything about model or

rewards and learn by itself doing trial and error. We choose solid red line (alpha=0.2 and epsilon=0.9) as our optimal parameters.

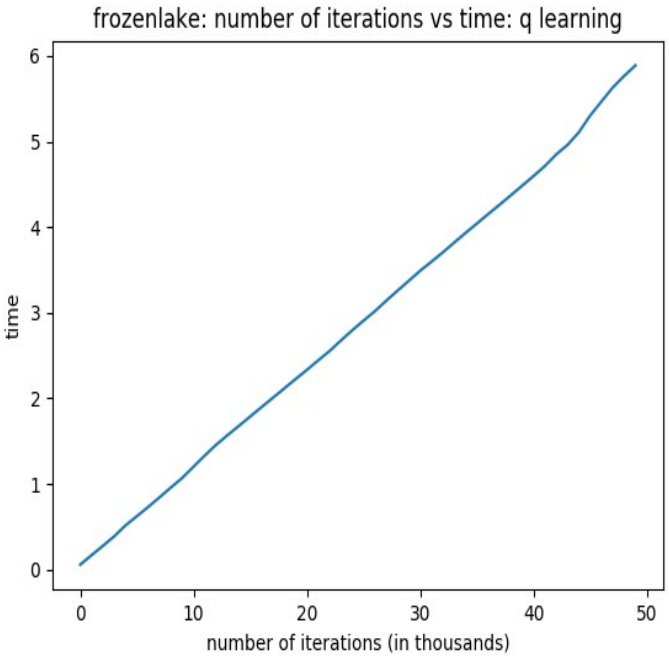

Fig 10. Convergence speed : Qlearning

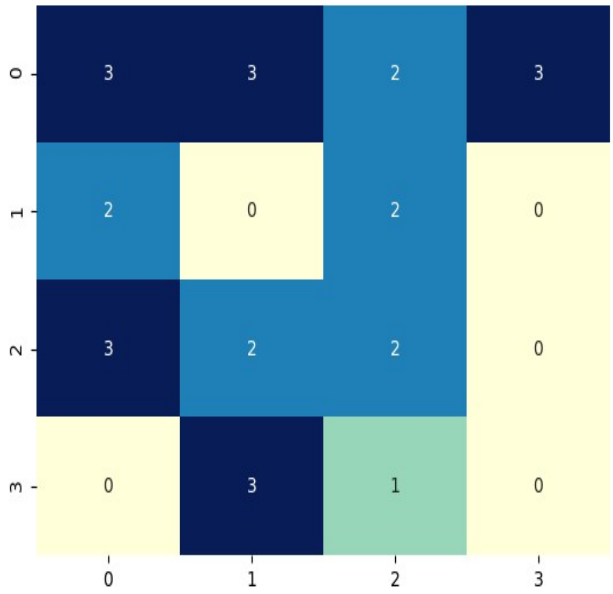

Fig11. Qlearning heatmap

How fast is Qlearning ? Qlearning is very very slow compared to VI/PI. As qlearning is model free algorithm which means it doesn't know anything about the state transitions and rewards, it has to disover what is good and bad actions by itself. Besides Fig 9. shows that algorithm prefer to do a lot of exploration by taking random decision (epsilon=0.9 converges best). It takes almost 50k iterations to converge. Hence qlearning takes a lot of time to converge.

Do they converge to same answer ? No, Qlearning does not converge to same policy as that of VI/PI.

Forest Management

Let's move to forest management problem. It takes number of states as an input parameter. I have used 2000 as this is my large problem. It has two actions (0: wait, 1: cut). Remaining parameters like reward for wait, reward for cut and fire probability i have used default values. Gamma (discount factor) value used is 0.9 for all three algorithms. I have used pymdptoolbox [3] to run three algorithms. To make this problem hard (interesting) i have reduced epsilon further (threshold value) for value iteration.

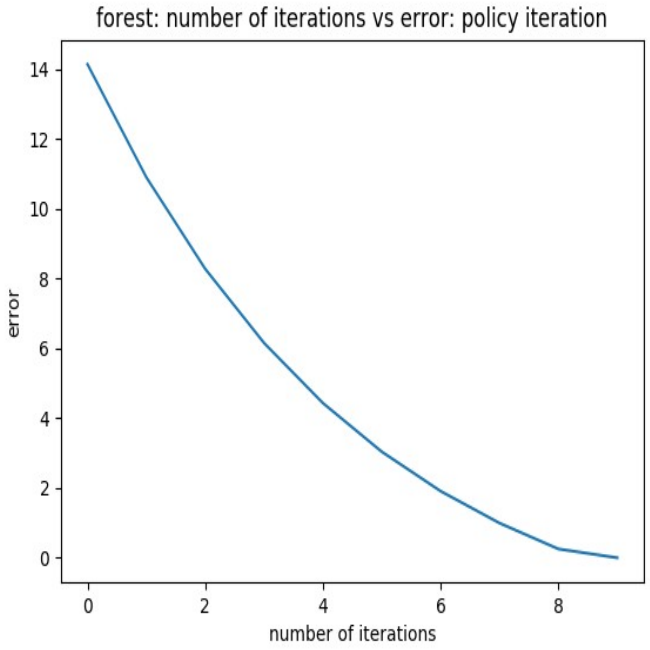

Fig 12. No of iterations to converge: PI

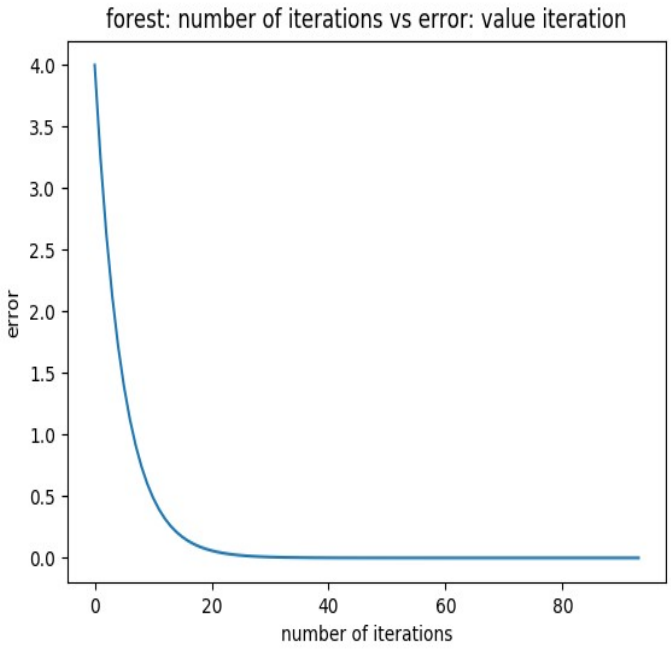

Fig 13. No of iterations to converge: VI

How many iterations does it takes to converge ? By looking at above graphs (Fig 12 & Fig 13), policy iteration took 10 (x3 frozen lake) iterations while value iteration took 94 (x7 frozen lake) iterations to converge. Value iterations takes more iterations here since we have made problem difficult by reducing threshold value.

What is the convergence criteria ? It is same as what we discussed above for grid frozen lake problem.

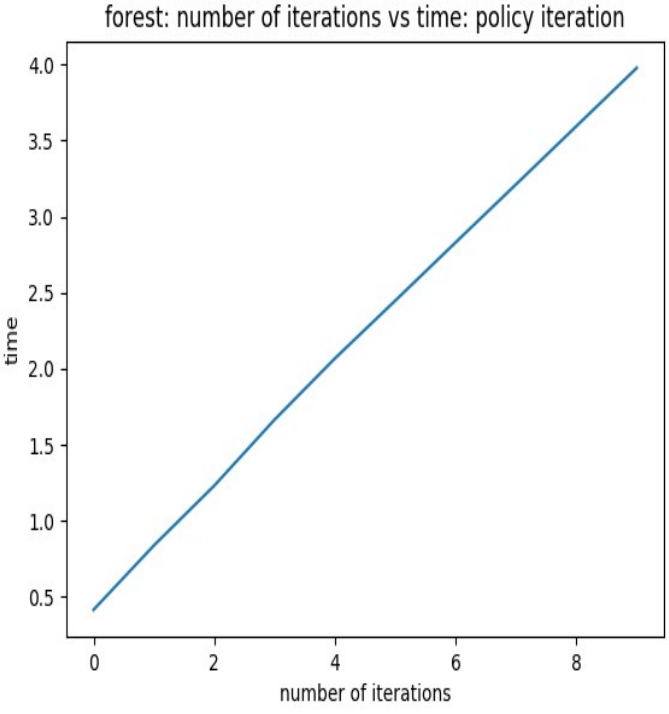

Fig 14. Convergence speed: PI

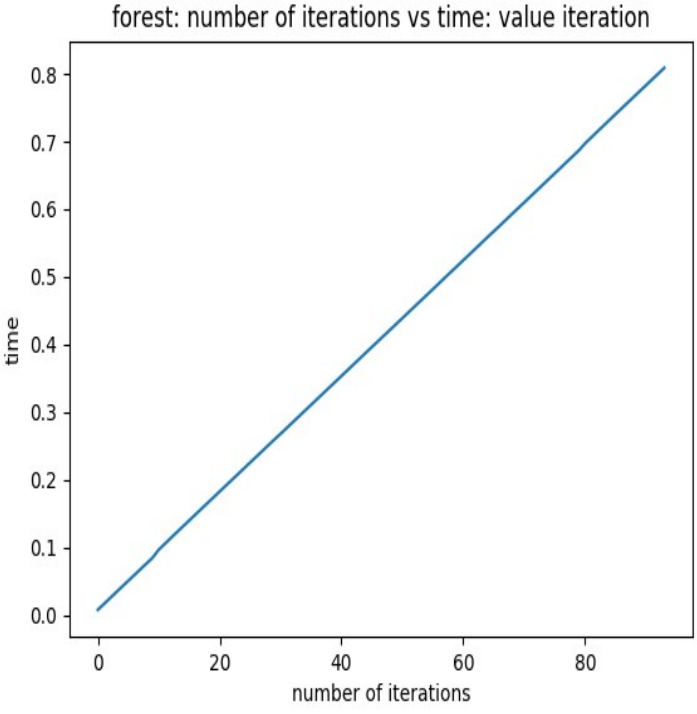

Fig 15: Convergence speed: VI

Which one converges faster ? Above graph (Fig 14 and Fig 15) clearly shows that value iterations converges (x5) faster compared to policy iteration for the same reason explained above for frozen lake example.

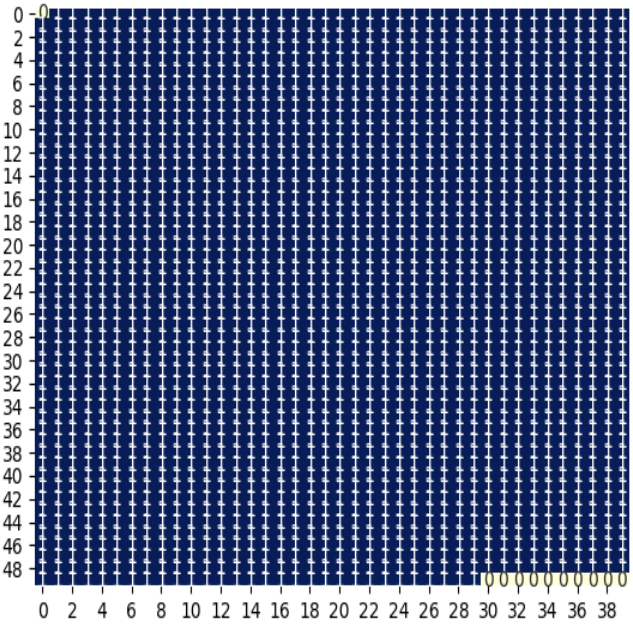

Fig 16. Policy iteration heatmap

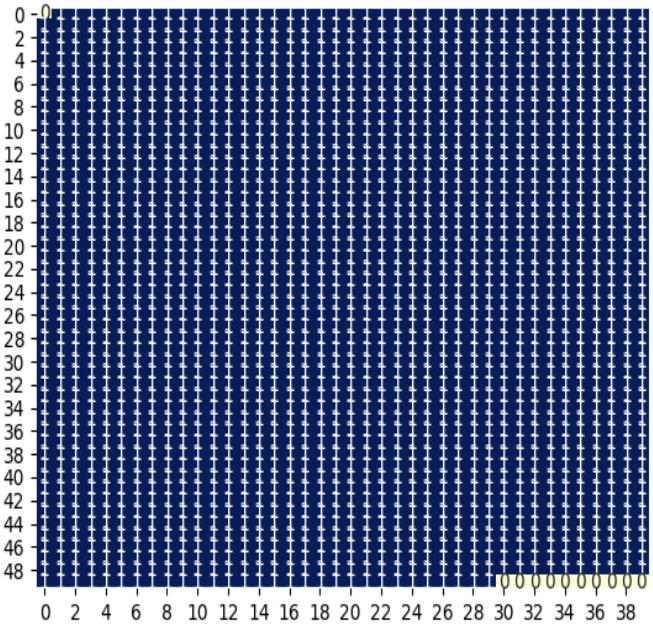

Fig 17. Value itetation heatmap

Do they converge to the same policy ? Yes, Above graphs (Fig 16 and Fig 17) clearly depicts that both the algorithm converges to the same policy. Since both VI and PI are guaranteed to converge to an optimal policy. Probably there exists only single optimal policy hence they converge to exact same policy.

How number of states affects ? It affected in two ways. More number of iterations to converge, more time to converge. In case of frozenlake (smaller) PI took 3 while VI took 14 whereas in case of forest (larger) PI took 10 and VI took 94 iterations to converge. Similary with respect to time, in case of frozenlake PI took around 0.0055 while VI took around 0.0030 whereas in case of forest PI took around 4 while VI took around 0.8.

Qlearning

Here our agent doesn’t know anything about the model. Agent has to figure out the best action based on q<state, action> values. Since agent doesn’t have knowledge, q learning takes more iterations to learn and converge compared to VI/PI. I have used 10K as number of iterations. I tried varying alpha (learning rate), epsilon (exploration rate). Higher epsilon means more exploration meaning more random decision. High learning rate means more weight to future samples compared to past. I am using gamma = 0.9 same as what i have

used for VI/PI. I am decaying alpha and epsilon for below experiments contrary to frozenlake.

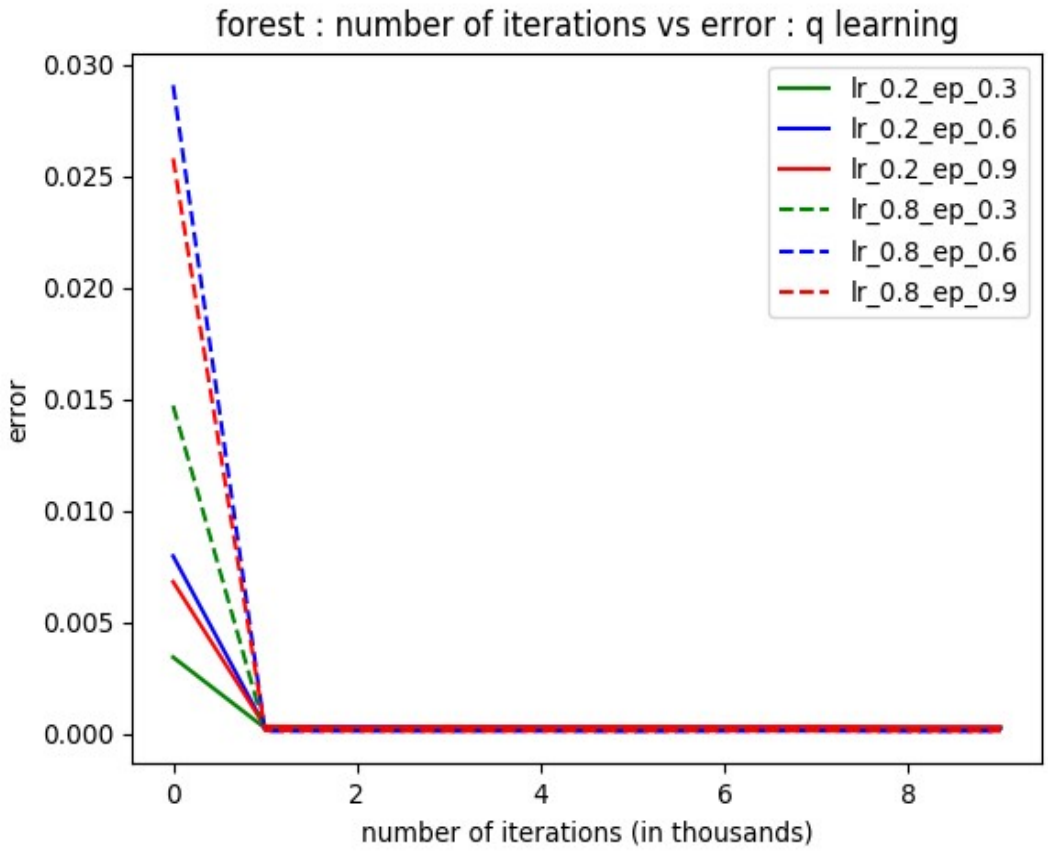

Fig 18. Iterations to convergence : Qlearning

Dotted line denotes high learning rate and solid line denotes low learning rates. Above graph (Fig 18) clearly shows that everything converges after first 1000 iterations. Also above graph shows that qlearning takes more number of iterations to converge compared to VI/PI but pretty less number of iterations compared to frozenlake qlearning probably because here we have introducted decay for learning rate and epsilon parameters which is balancing between exploration and exploitation. We chose solid red line (alpha=0.2 and epsilon=0.9) as our optimal parameters.

How do you define convergence ? Convergence is defined same as what we defined above for frozen lake.

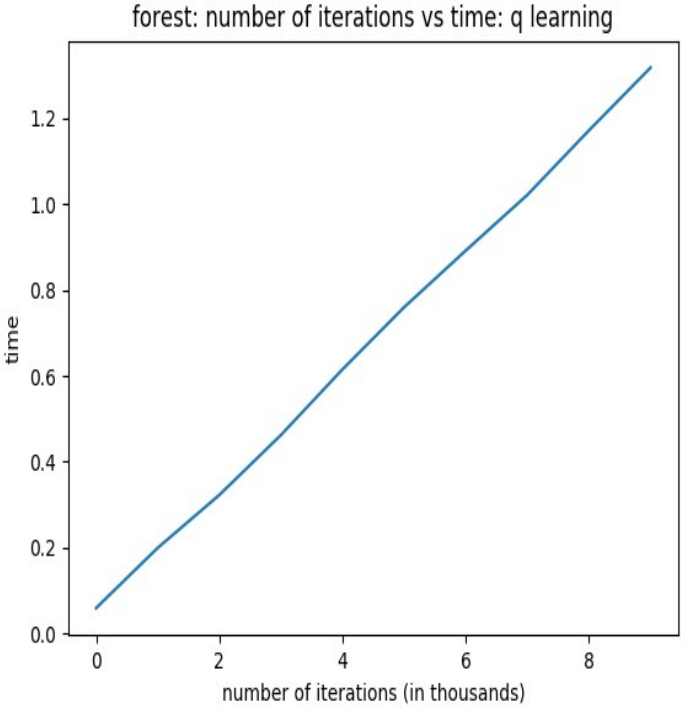

Fig 19. Convergence speed

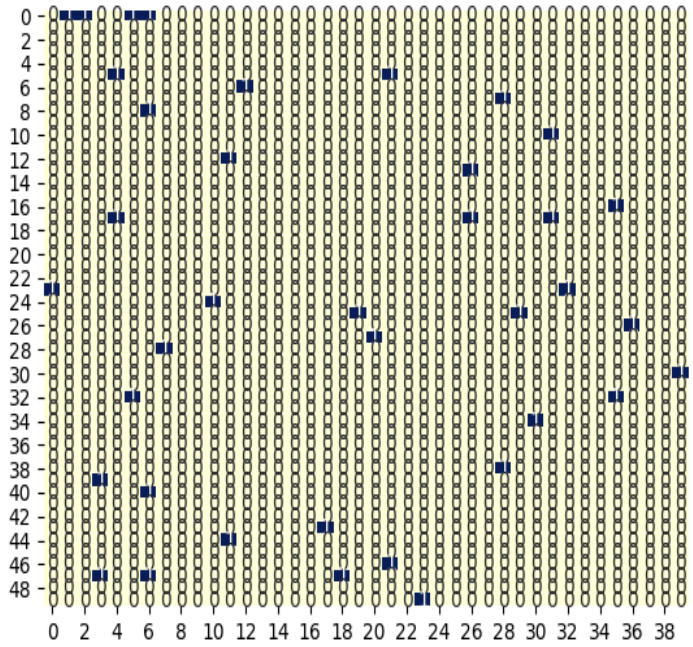

Fig 20. Qlearning heatmap

How fast is Qlearning ? Above graph (Fig 19) shows that qlearning converges faster (x20) compared to policy iteration and (x4) compared to value iteration. This behavior is quite surprising as Qlearning is generally slower compared to VI/PI probably because Qlearning is convering in 1000 steps and algorithm just update q value in q table which is not a lot of computation compared to what VI/PI do.

Do they converge to the same answer ? No, Fig 20. shows that policy returned by q learning is not same as that of VI/PI.

References

[1] Retrieved from https://hal.archives-ouvertes.fr/hal-01413550/document
[2] Retrieved from https://gym.openai.com/envs/FrozenLake-v0/
[3] Retrieved from https://github.com/hiive/hiivemdptoolbox
[4] Retrieved from https://medium.com/@m.alzantot/deep-reinforcement-learning-demysitifed-episode-2-policy-iteration-value-iteration-and-q-978f9e89ddaa Moustafa Alzantot, July 9, 2017.
[5] Retrieved from http://incompleteideas.net/book/RLbook2018.pdf