CS7641 Machine Learning Assignment 4 by Georgia Tech: Markov Decision Process

T. Ruzmetov

November 20, 2017

Ownership of the following report developed as a result of assigned institutional effort, an assignment of the CS 7641 Machine Learning course shall reside with GT and the instructors of this class. If the document is released into the public domain it will violate the GT Honor Code

Introduction

In this project, we explore Markov Decision Process on two grid world problems. One with small number of states(6x6 grid, 25 states excluding obstacles), while the other one has large number of states(13x13 grid, 106 states when obstacles are excluded). Initial and finale positions for an agent is same, but structure of grid(maze) is different. We solve both problems via three different approaches: value iteration, policy iteration and Q-learning. One of the implications of grid world problem that makes it interesting is rout planning. In our daily life, for instance, we commute from home (initial state) to work(finale destination) and it is important for us to know the best rout especially when we live in a crowded city. Also, by assigning negative reward to states associated with high crime rate blocks we can design not just best rout but also safest rout. Traffic information can also be entered into a model to improve the rout. Above mentioned example definitely makes grid world problem very interesting to study.

Markov Decision Process

MDP is stochastic decision making process, where the policy learnt by an agent for selecting next action is based on current observed state and action. Besides, this new policy depends only on nearest previous state and action and not on earlier states which is called Markovian property. The estimations made based on the Markovian property are as good as the estimations made with the knowledge of the full history up to that point.

Value Iteration

Value iteration uses Bellman equation, which essentially states that the true value (or utility) of a state is the immediate reward for that state, plus the expected discounted reward if the agent acted optimally from that point on:

$$U(s) = R(s) + \gamma argmax_a \sum_{s'} T(s, a, s')U(s')$$

1. Assign each state a random value
2. For each state, calculate its new utility based on its neighbor's utilities.
3. Update each state's utility based on the calculation above: $U_{i+1}(s) = R(s)+\gamma argmax_a \sum_{s'} T(s, a, s')U_i(s')$
4. Stop upon convergence.

This algorithm is guaranteed to converge to the optimal solutions.

Policy Iteration

In policy iteration method we directly and iteratively update policy until we reach optimal policy. We can do so by modifying value iteration to iterate over policies. We start with a random policy, compute each state's utility given that policy, and then select a new optimal policy. Here is the algorithm:

1. Randomly initialize policies for all states
2. Evaluate $U_t = U^{\pi_t}$ given π_t
3. Improve $\pi_{t+1} = argmax_a \sum T(s, a, s')U_t(s')$

and Bellman's equation simplifies into n linear equations with n unknowns that is easy to solve: $U_{i+1}(s) = R(s) + \gamma \sum_{s'} T(s, \pi_t(s), s')U_i(s')$

Q-Learning

Value Iteration and Policy Iterations rely on domain knowledge, where agent accurately knows the transition function and the reward for all states in the environment. In contrast, Q-learning agent learns a policy or value function directly from experience (model-free). Q-learning estimates optimal Q-values of an MDP, where its behavior is dictated by taking actions greedily with respect to the learned Q-values[4].

1. Arbitrarily initialize Q-values for all state-action pairs
2. Then, until learning converges...
    - Choose an action (a) in the current world state (s) based on current Q-value estimates.
    - Take the action (a) and observe the the outcome state (s') and reward (r).
    - Update $Q(s, a) = Q(s, a) + \alpha[r + \gamma max_{a'} Q(s', a')Q(s, a)]$

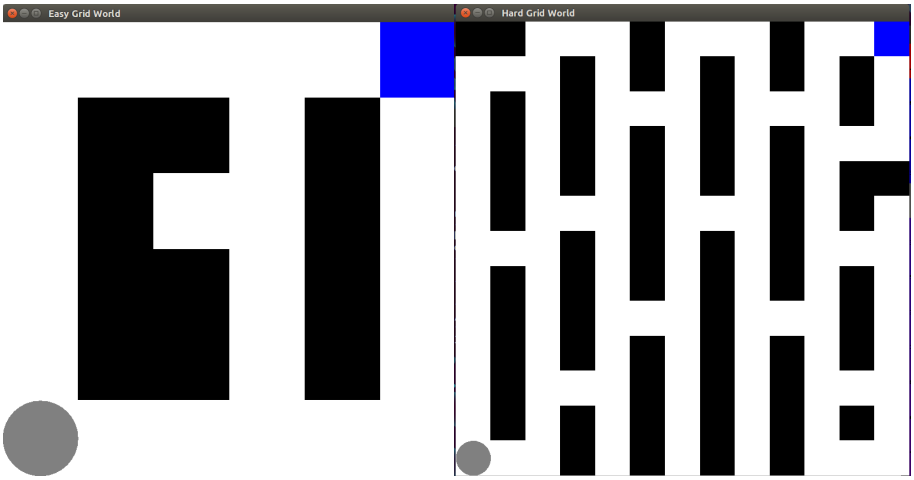

Figure 1: Here we show 6x6(left plot) easy and 13x13(right plot) hard grid world problem topologies. As depicted, grey circle is the agent and blue square at top right corner is final destination.

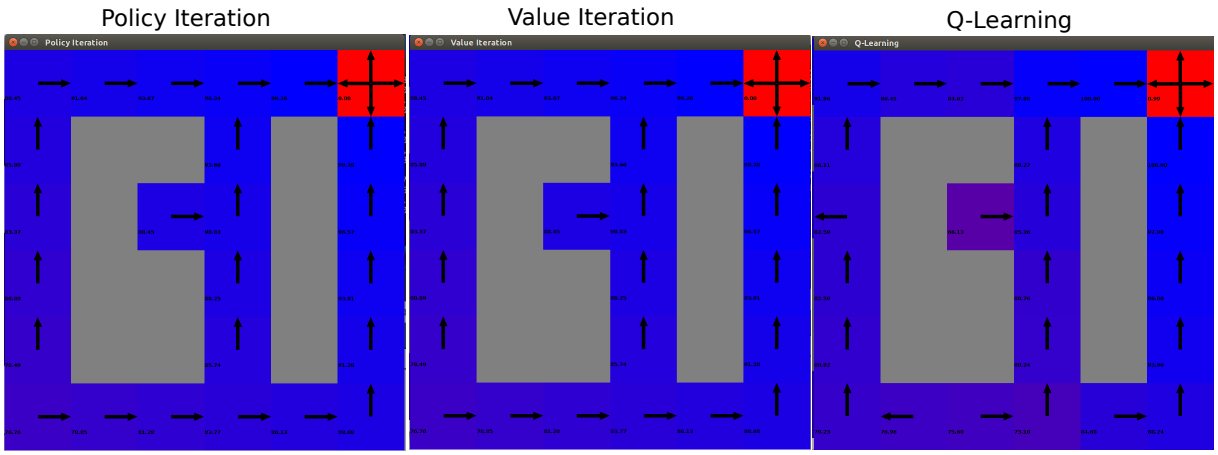

Figure 2: Finale, optimal policy is depicted for all three algorithms with the heat map representing V-values for each cell.

Easy Grid World Problem

Easy Grid World problem composed of 6x6 grid, where agent starts its journey at bottom left corner and by finding optimal policy should end its journey at top right corner for a given structure of maze Fig1. All three algorithms are applied to this problem. Optimal policies are illustrated in Fig2, where V-values that is the expected reward of each state are shown via heatmap: blue for high V-values and red for low V-values. Interestingly, states that are close to terminal(goal) state, V-values increase.

Value iteration and policy iteration algorithms show similar optimal policy at 300 iterations, which suggests that this optimal policy must be the correct policy. Even V-values shown in heatmaps

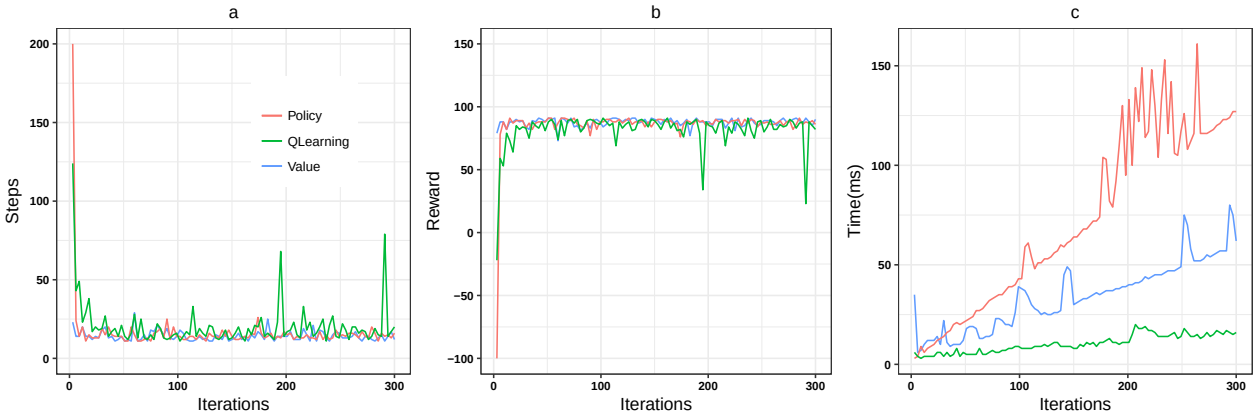

Figure 3: Here we show convergence results for easy problem, where number of iterations are plotted against average number of steps taken from any state to terminal state (a), average reward (b) and finally performance time for all three methods(c).

perfectly agree with each other. In addition, convergence plots shown in Fig.3 demonstrate that for both value iteration and policy iteration methods reward and steps converge very fast within 4-5 iterations, in fact both above methods convergence happened at the same time Fig4(a,b). In contrast, Q-learning algorithm(lr=0.6, epsilon=0.1) takes a lot longer time to converge to optimal policy because even at very finale steps of iterations we see high fluctuations in reward value Fig.3. In a heatmap shown for Q-learning in Fig.2, cells (4,1) and (1,3) show the action which is opposite to actions of optimal policy suggesting that method hasn't completely converged to an optimal policy.

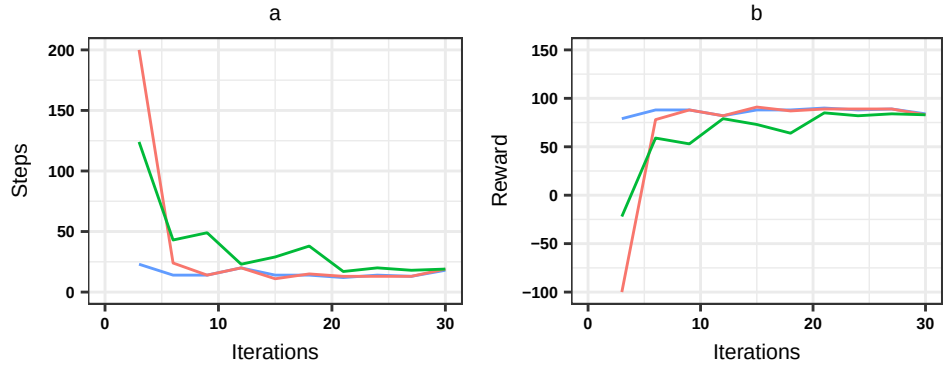

Figure 4: Closer view of convergence for easy problem

Furthermore, both steps and rewards plotted vs iterations Fig.3 show that Q-learning initially show very high fluctuations and slowly converge. After 300 iterations fluctuations reduce to some extent but full convergence is not yet achieved suggesting need for more iterations or hyperparameter optimization. Iterations vs computation time plot shows that policy iteration is most time consuming and Q-learning is the least time consuming. Curves for all three methods show linear increase in computation time with respect to iterations taken.

In order to get better understanding about Q-learning method I performed 2000 iterations with

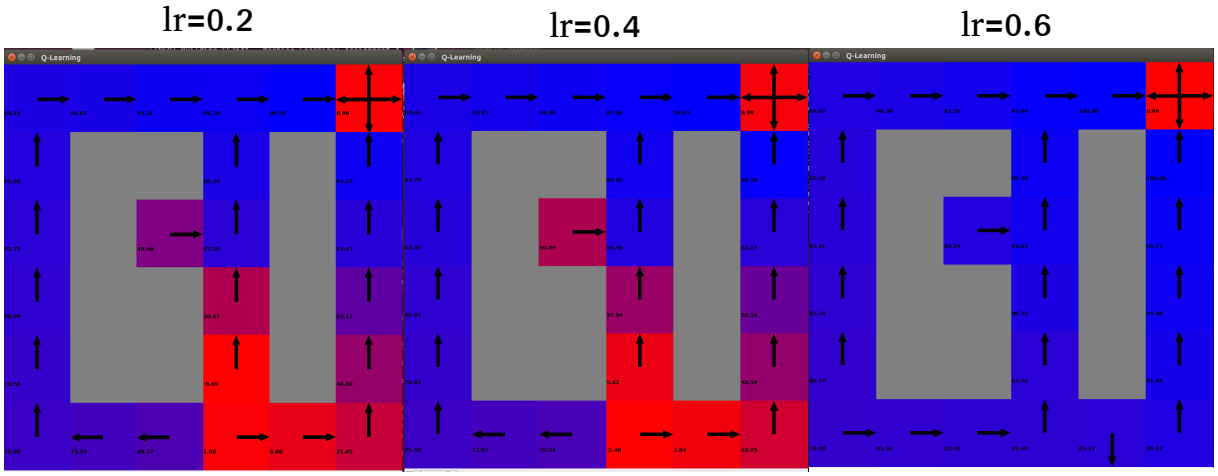

Figure 5: Easy Grid World optimal policies for Q-learning algorithm is shown for different values of learning rate and 2000 iterations.

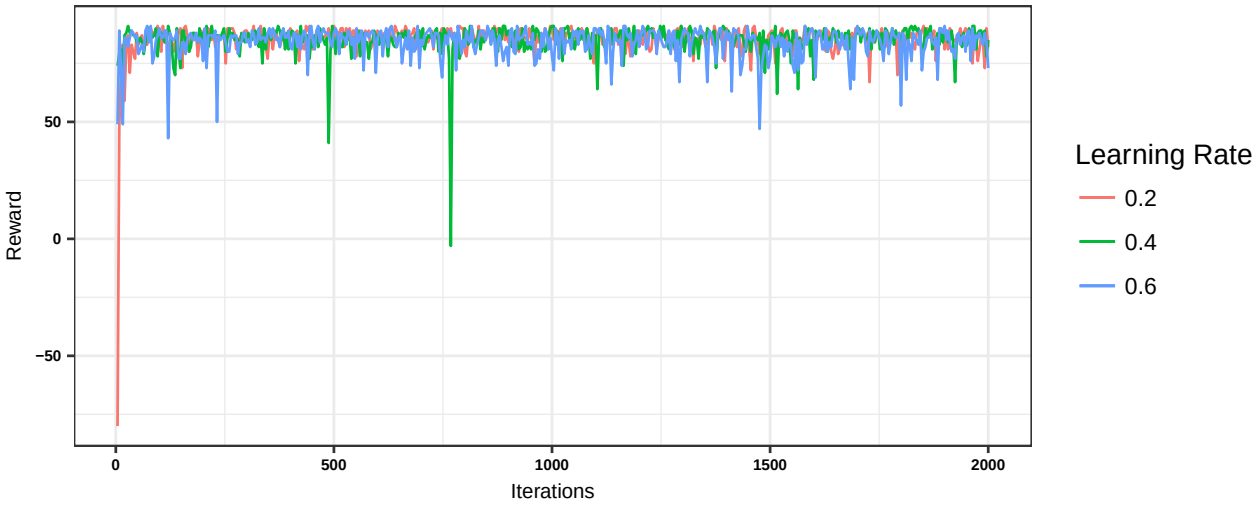

Figure 6: Q-learning analysis for easy grid world problem at different values of learning rate

three different values of learning rate just to see if the policy can give similar optimal policy as other methods. This investigation was done on easy grid world problem with small number of states, where epsilon value kept constant $\epsilon = 0.1$. As we can see in Fig.4 by increasing learning rate optimal policy improves approaching the ideal policy obtained by other two methods. This suggests that larger learning rate assists converge faster, but as we can see in Fig.5 it also results in high fluctuation of reward value which makes convergence hard. In order to overcome this problem, one can apply a method where learning rate decreases with iterations. Then initially at large values of learning rate model converges fast and later as learning rate is small, it helps to suppress high fluctuations consequently giving best optimal policy.

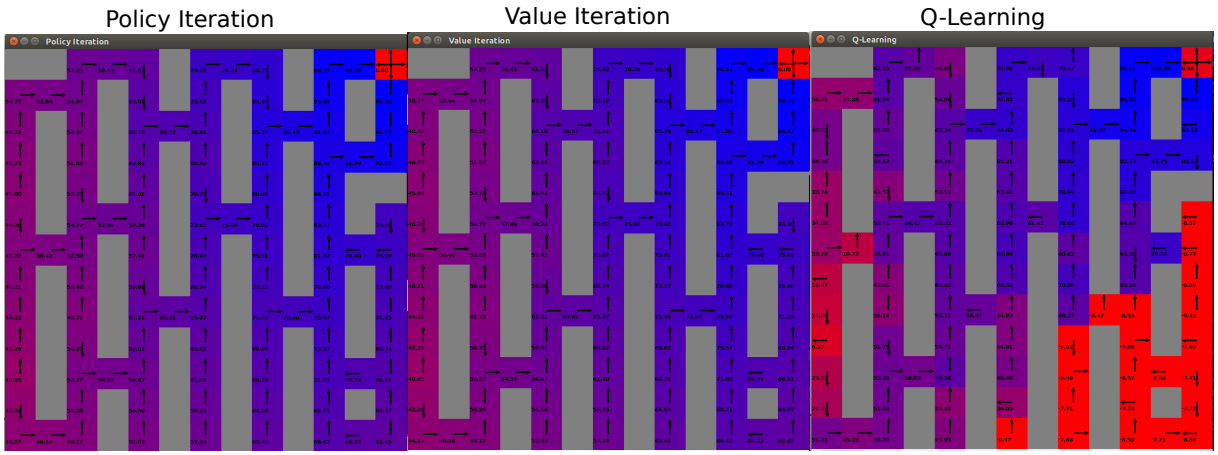

Figure 7: Hard Grid World optimal policy is shown for all three algorithms with heat map representing V-values for each cell.

Hard Grid World Problem

Hard Grid World problem composed of 13x13 grid, where agent takes a long journey in a topologically complex maze Fig.1(right plot). Optimal policies are illustrated in Fig.7, where V-values that is the expected reward of each state are shown via heatmap. Again, surprisingly, value iteration and policy iteration algorithms show similar optimal policy evaluated after 1000 iterations. Value iteration algorithm solves Bellman's equation by iteratively updating utility function, wheres policy iteration approach directly updates policies(iteratively in similar passion) in order to find optimal policy. So, then it is not surprising to see both algorithms to result in same finale policy given adequate number of iterations since methods have a lot in common.

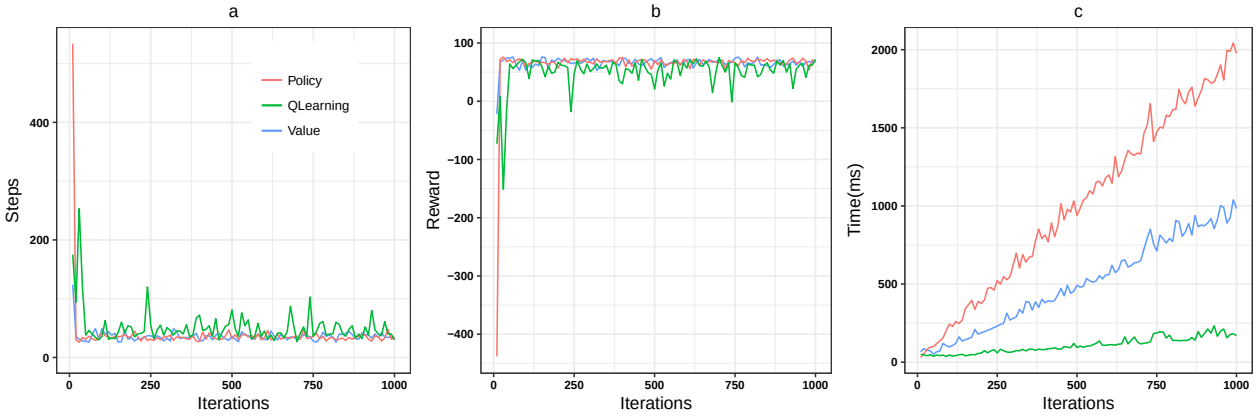

Figure 8: Here we demonstrate convergence results for hard problem, where number of iterations are plotted against number of steps (a), total reward (b) and finally performance time for all three methods(c)

Heatmap for Q-learning(Fig.7,lr=0.6, epsilon=0.1) show that 1000 iterations was not sufficient for convergence. Some of the cells, especially once at lower left corner show actions that point against

the wall and some point opposite to a optimal policy achieved by other two methods. One can resolve this issue by reducing the learning rate, but that might require even longer number of iterations.

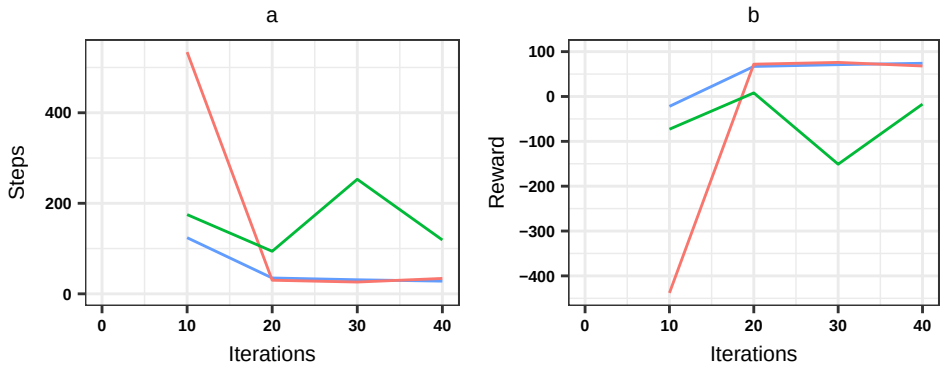

Figure 9: Closer view of convergence for hard problem

Value iteration and policy iteration methods again show convergence at the same time Fig.9, but for this problem it took around 20 iterations that is longer compared to 4-5 iterations obtained for easy grid world problem. In Fig.8, we show convergence plots for hard grid world problem done for 1000 iterations for all three methods. Once optimal policy is found value iteration and policy iteration methods remain at that state with minor fluctuations. On the other hand, Q-learning method show high fluctuations in both steps and rewards throughout whole iteration process. Time consumption of policy iteration is the highest and Q-learning methods computation clock time is almost constant with iterations. Hard grid world problem has 106 states that is large compared to 25 states of easy problem. Due to this difference, computation time for hard problem is longer. For instance, it took ~700 ms to cover 300 iterations for hard problem, where to cover same amount of iterations easy grid world problem consumed ~125 ms.

Conclusion

We have investigated practical aspects of Markov decision processes and reinforcement learning algorithms via value iteration, policy iteration and Q learning. Our studies show that value iteration and policy iteration algorithms converge to an optimal policy very fast since both methods use some form of domain knowledge. On the other hand, Q-learning algorithm directly learns from experience(model-free), which makes it hard to converge. Thus, in order to achieve optimal policy one has to iterate Q-learning method a lot longer. This doesn't mean longer time, because since method uses greedy approach time performance is almost constant with iterations. On the other hand proper tuning of learning rate found to be important. For a high learning rate the agent improves its policy faster but struggles to settle down to an optimal policy. This problem can be solved by setting dynamics learning rate that decays with number of iterations.

Thanks

References

1. T. M. Mitchell, Machine Learning. McGraw Hill, 1997.
2. L. P. Kaelbling, M. Littman, and A. Moore, 'Reinforcement learning: A survey,' Journal of Artificial Intelligence Research, vol. 4, pp. 237-285, 1996.7
3. R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction. London, England: The MIT Press, 1998.
4. http://burlap.cs.brown.edu/index.html
5. https://github.com/truzmeto/RL_MDP
6. https://github.com/juanjose49/omscs-cs7641-machine-learning-assignment-4