CS 7641: Assignment #4
Markov Decision Processes

Due on Sunday, November 27, 2016

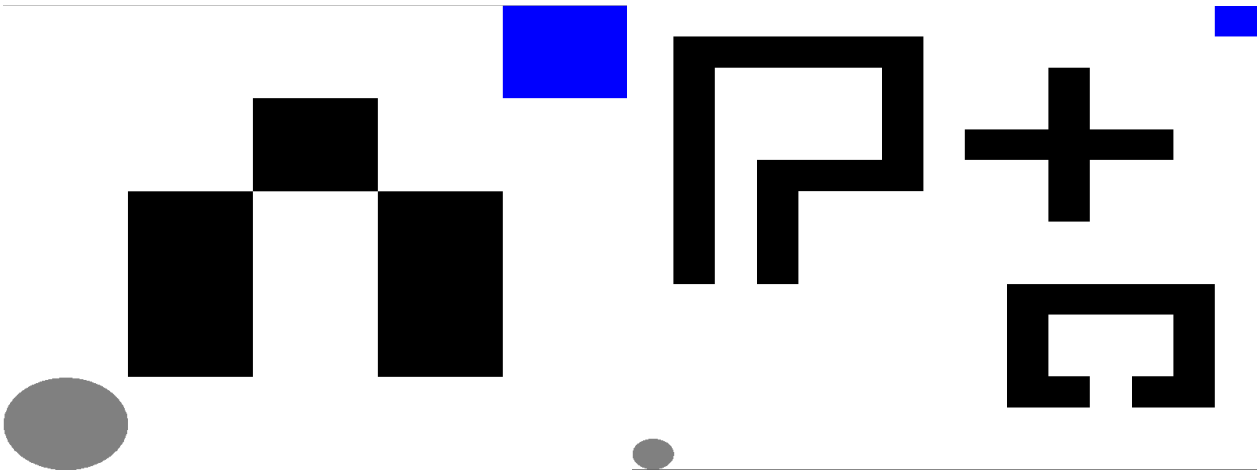
Dr. Charles Isbell

Ryan Chow

Introduction

This report explores the application of techniques from reinforcement learning to Markov Decision Processes (MDPs). Two interesting MDPs are presented below and then solved using Value Iteration, Policy Iteration, and Q-learning. These three methods are compared quantitatively using analysis criteria such as iterations, convergence, computation time, and solution optimality. This comparison yields quite interesting results that demonstrate the advantages and disadvantages of each technique when dealing with MDPs of different sizes. Additionally, various Q-learning parameters (iterations and initial q-values) are tweaked in the final section of this report to see how results can be improved.

The first MDP (Small Grid) contains 20 possible states, with a start node at the bottom left corner and a goal node at the top right corner. This grid map contains an upside-down "U" shape that the agent must avoid getting stuck in. The second MDP (Large Grid) contains 180 possible states, with a start node at the bottom left corner and a goal node at the top right corner. This grid map contains various walls that the agent must navigate around or avoid completely in order to reach the goal node.



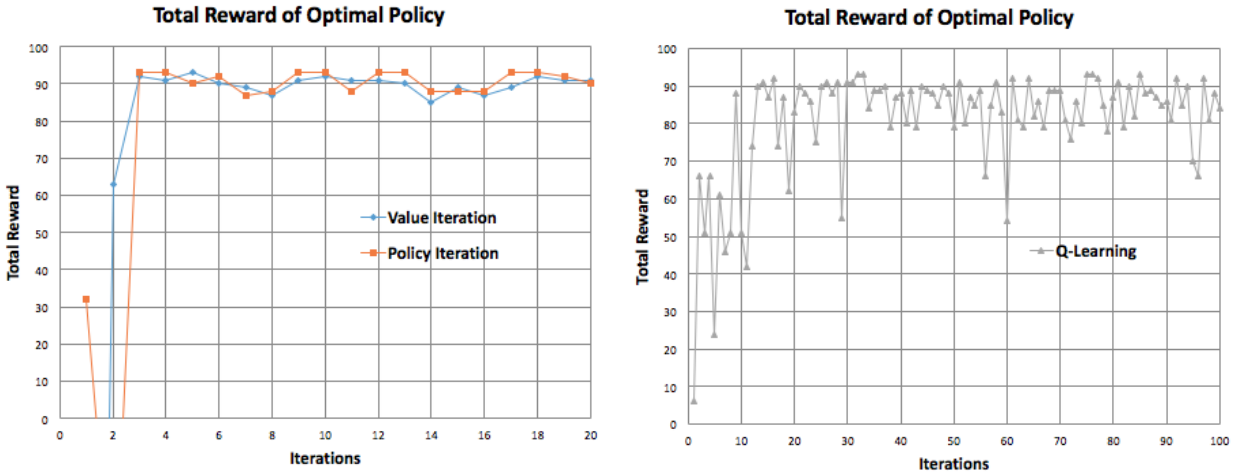
(a) Small Grid (20 states) (b) Large Grid (180 states)

Figure 1: Two interesting MDP grid worlds

These MDPs are interesting because they pose interesting challenges for an agent looking to maximize reward over its lifetime. The Small Grid is fairly simple at first glance - optimal paths could be following the grid perimeter North four steps and then East four steps. However, the stochastic transition nature (80% chance of expected action, 20% chance other action) may influence the optimal policy to avoid the bottom route completely to mitigate the chance of accidentally entering into the upside-down "U" wall. The Large Grid is similarly interesting because it has two areas that the agent can get stuck in and should avoid completely, in addition to a plus-shaped obstacle near the goal node. It will be interesting to see if the agent encourages a longer diagonal path towards the goal instead of following either perimeter to avoid low-utility states. These grid world problems are grounded in and inspired by many 2D-arcade games, robotic applications, and simplified planning problems.

Small MDP

Figure 2 below shows the total reward of the optimal policy as the number of iteration number increases. All experiments were run using 100 iterations. Value and policy iterations were able to converge on the optimal solution after only three or four iterations, this was a very low number compared to Q-learning which required upwards of twenty iterations to converge on a solution. For such a small MDP space, this is a pretty significant difference which I imagine is only exacerbated in larger MDP spaces. The reason for this is that Q-learning requires the agent to learn from the grid world, through trial and error, without any domain knowledge. In contrast, value and policy iteration assume full knowledge of the transition function and reward for all states in the environment. This additional information allows the value and policy iteration algorithms to converge much faster on an optimal policy.



(a) Value and Policy Iteration

(b) Q-learning

Figure 2: Total Reward of Optimal Policy

Figure 3 below shows the number of actions required for the agent to reach the goal node from the start node. It is interesting to see that once value and policy iteration converge, they hover roughly around the same number of steps. Q-learning is far more erratic, as it may take longer for the Q-learning agent to uncover the small nuances of the map because it must learn from trial and error experience.

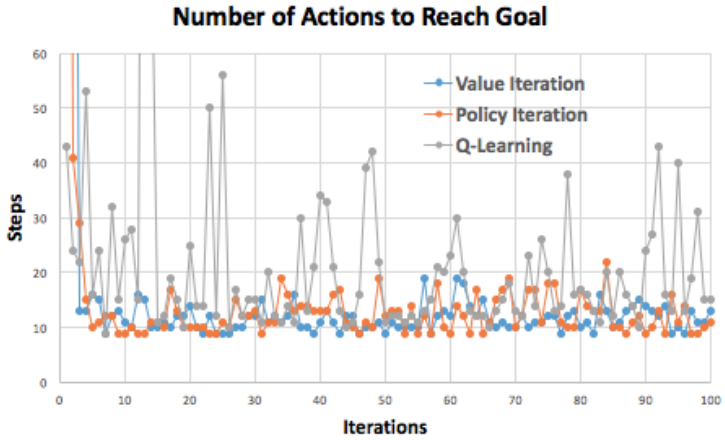


Figure 3: Steps to reach goal for small MDP

Figure 4 below demonstrates the computation time for each of MDP techniques over 100 iterations. It is obvious that Q-learning is the least computationally-expensive method while policy iteration is just barely quicker than value iteration for this small MDP grid. Intuitively, this makes sense because value iteration is storing all the values of each state while policy iteration finds the policy directly. In addition, value iteration make take a long time to converge even if the underlying policy for it isn't changing. Q-learning on the other hand, does not need to compute the values for every single state at every iteration. The agent in this scenario is an online learner and computes based on trial-and-error, far less computationally expensive than value or policy iteration.

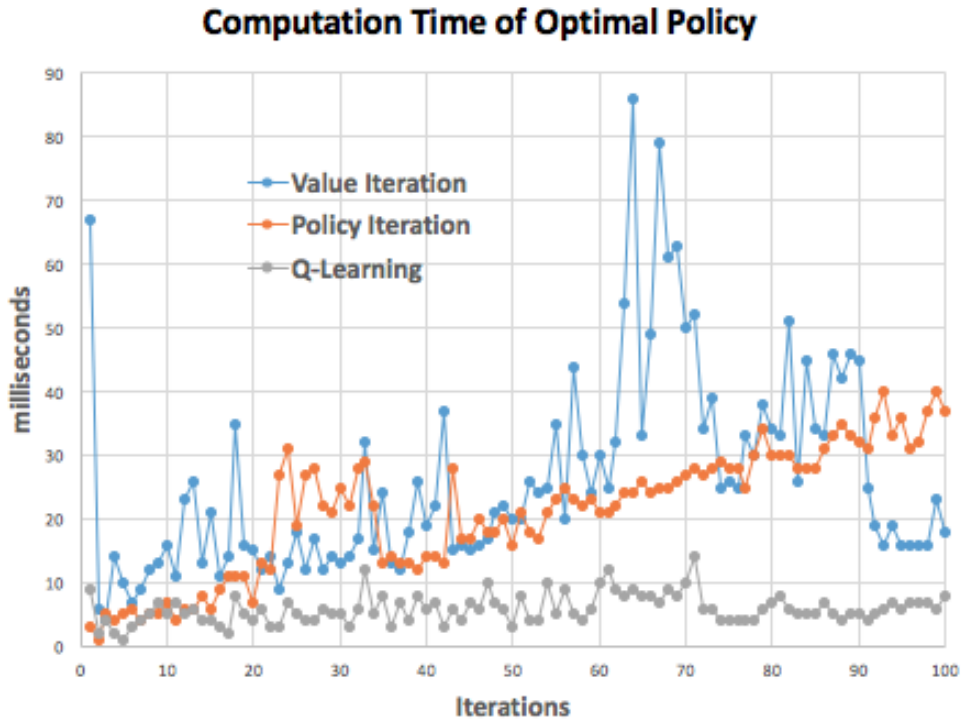
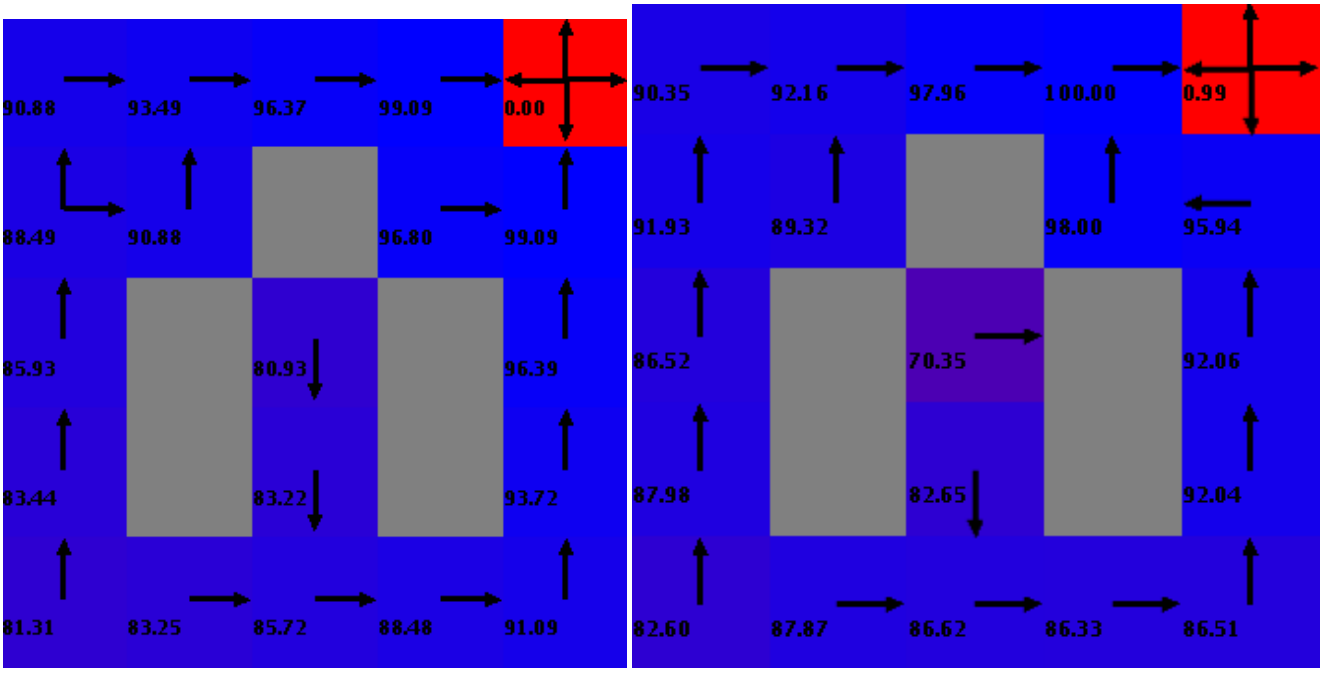


Figure 4: Computation time for small MDP

Small MDP continued on next page...

Figure 5 below shows the optimal policies found by each of the methods. Note that value and policy iteration converged on the same exact optimal policy, so only one image is shown below on the left. Q-learning found a solution that is clearly less optimal than the solutions found by value/policy iterations. The state in the middle center of the grid and the state directly below the goal are clearly sub-optimal. However, the remaining 18 states are nearly identical to the optimal solutions found by value and policy iteration. At this point, there seems to be a tradeoff between solution optimality and computation time. Q-learning was able to find a slightly less optimal policy compared to value/policy iteration, but Q-learning was able to do it nearly a magnitude of order quicker. It will be interesting to see if this conclusion scales to a large MDP in the next section.



(a) Value/Policy Iteration (b) Q-learning

Figure 5: Optimal Policy after 100 iterations

Small MDP continued on next page...

Large MDP

For the Large Grid, the difference in convergence between Q-learning and value/policy iteration are similar to the results found for Small Grid. Value and policy iteration converged on an optimal solution by around twenty iterations, while Q-learning does not appear to converge until around forty or more iterations. It is important to note the scale of the y-axis because the magnitude of fluctuation at around the reward convergence line is much larger for Q-learning than value/policy iteration. This could mean that the Q-learning has not yet actually converged during these 100 iterations and could require much more iterations to settle.

(a) Value and Policy Iteration

(b) Q-learning

Figure 6: Total Reward of Optimal Policy

The different optimal solutions for value/policy iteration and Q-learning are further demonstrated by the optimal steps required to reach goal node from start node as shown in Figure 7 below. It is clear that value and policy iteration consistently require around thirty steps to reach the goal node while Q-learning has very erratic behavior. This may be because Q-learning is much more likely to make sub-optimal decisions as it is still learning about the environment. The performance difference between Q-learning and value/policy iteration is greater in this Large Grid compared to the Small Grid, meaning that Q-learning struggles to scale a comparable optimal solution as the state space grows larger.

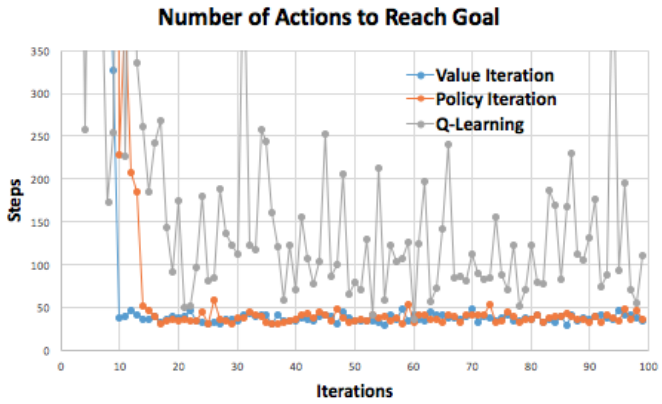


Figure 7: Steps to reach goal for large MDP

Large MDP continued on next page...

The computation time for Q-learning, value iteration, and policy iteration were very surprising to me. Figure 8 below shows that each method grows linearly in time with the number of iterations (as expected from the previous section Small Grid). However, this time the policy iteration quickly outgrows the value iteration in terms of computation time. I'm not sure what causes this, but since the policy iteration is divided into two steps (policy evaluation and policy improvement), I think that the large number of states is causing the policy evaluation step to consume a lot of computation time due to the large state space.

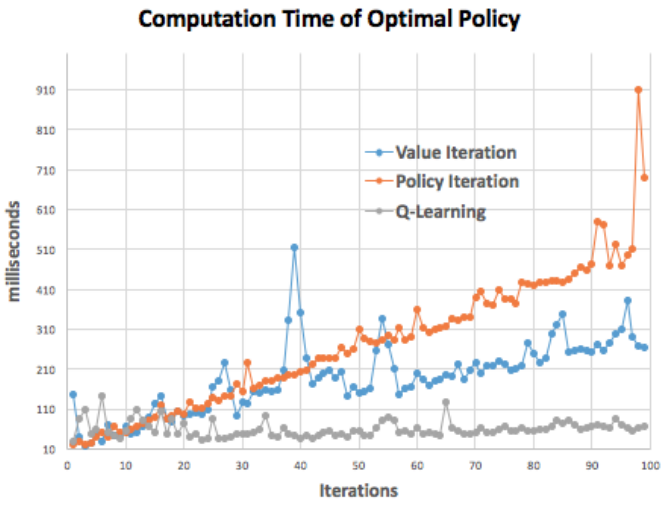
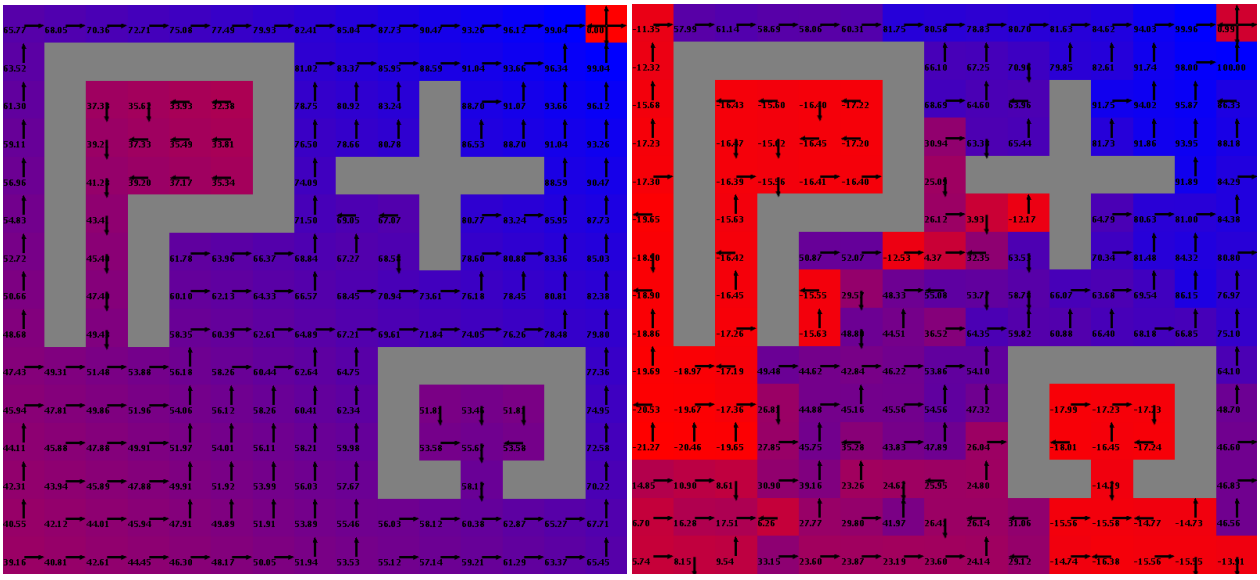


Figure 8: Computation time for large MDP

Again, value and policy iteration converge on the same optimal policy after 100 iterations as shown on the left below in Figure 9. 100 iterations for Q-learning is able to find a good solution for the states near the goal node, but fail to find good policies for the states near the bottom left start node. Additional Q-learning experiments with larger number of iterations are shown in the next section to see if comparable policies can be found.



(a) Value/Policy Iteration (b) Q-learning

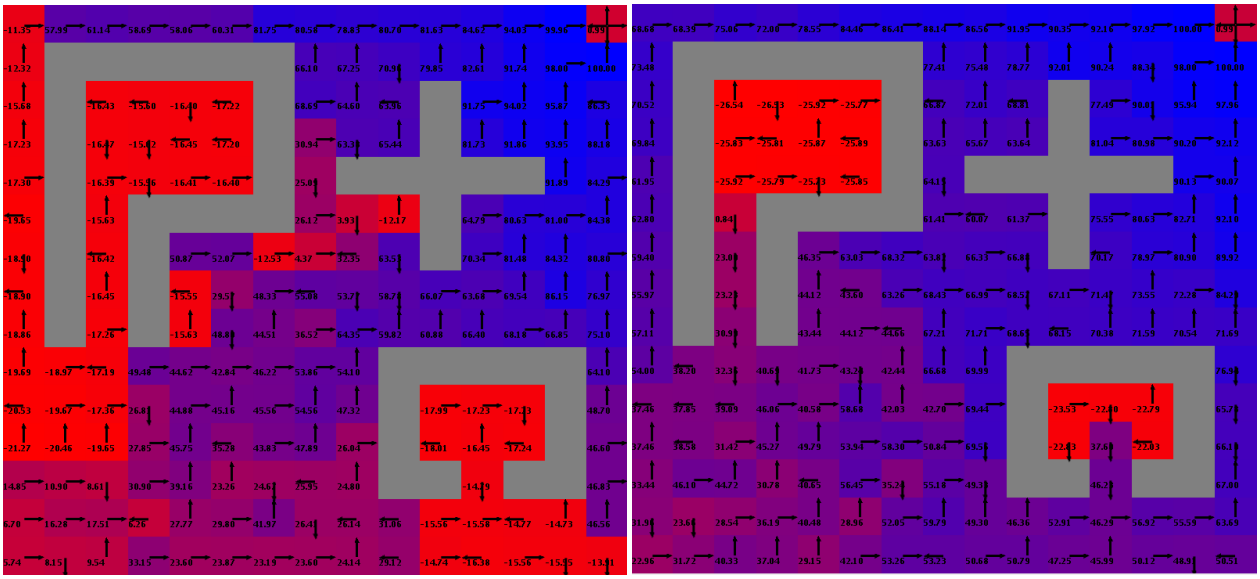
Figure 9: Optimal Policy After 100 Iterations

Large MDP continued on next page...

Q-learning

Iterations

In addition to Q-learning performed on the small and large MDPs in the previous sections, addition experiments were run to see if Q-learning could achieve comparable performance to value/policy iteration. Figure 10 below shows the same Q-learning algorithm applied to the Large Grid MDP problem using 100 iterations and 10,000 iterations. The results show that the 10,000 iteration solution policy is far better than the policy found using 100 iterations. This is expected, as a large MDP space requires a lot of trial-and-error for the agent to learn about the environment. However, it is interesting to note that even with 10,000 iterations, the policy found is still noticeably different than the single optimal policy found by value/policy iteration. Even at 10,000 iterations, the computation time for Q-learning is comparable to value/policy iteration at 100 iterations.



(a) Q-learning (100 iterations) (b) Q-learning (10,000 iterations)

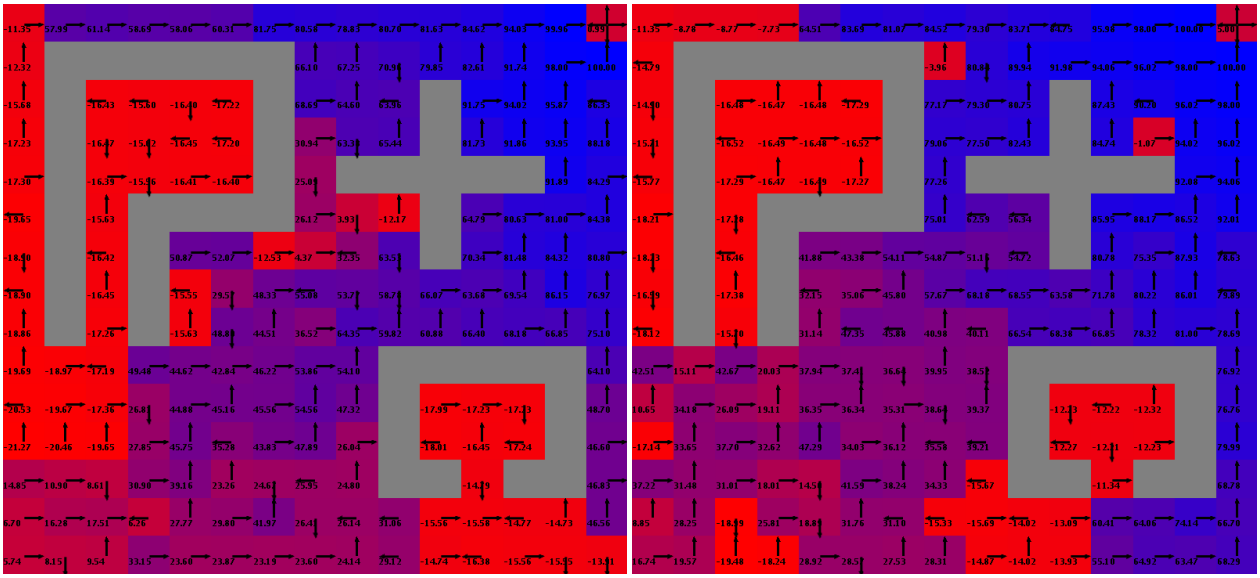
Figure 10: Q-learning 100 iterations vs. 10000 iterations

Q-learning continued on next page...

Q-learning

Exploration Strategy

In addition, various exploration strategies were chosen by tweaking the initial conditions (initial q-values). High initial values are optimistic and encourage exploration because no matter what action is selected, the update rule will cause the state value to decrease. This is an exploration-exploitation tradeoff. As shown below, the higher exploration preference (larger initial q-value) on the grid on the right in Figure 11 allow the agent to explore territory and find a more optimal solution given the smaller number of iterations (still only 100 iterations for both). Obvious improvements are shown in the lower right hand corner of the map as well as around the perimeter of the cross-shaped wall.



(a) initial q=0.99 (b) initial q=5

Figure 11: Q-learning comparing initial condition values

Q-learning continued on next page...

Conclusion

Practicality: In reality, the agent often do not have full knowledge about the environment. The stochastic transition function and reward for all state are usually not known. As such, it is not feasible to use value or policy iterations for these problems. Online learning methods such as Q-learning provide good approaches to learning about the environment while finding an optimal solution. Although Q-learning never surpassed value/policy iteration in my experiments, the resulting policies were often satisfactory and still very much usable.

Tradeoff: The benefit of Q-learning is that that we can trade off computation time with solution optimality. On average, Q-learning is much faster to run compared to value/policy iteration at the expense of better solutions. Running Q-learning for a larger number of iterations may take longer than value/policy iteration but the policy will continue to improve towards the optimal solution.