CS7641 Assignment 4 Jacob Kilver 11/30/14

I. Introduction

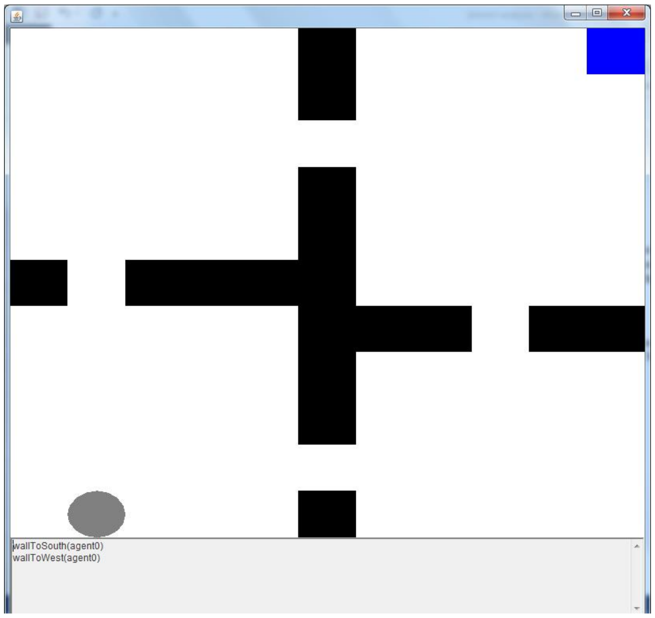
For this assignment two Markov Decision Processes (MDPs) were explored, one with a small number of states and one with a large number of states. The Brown-UMBC Reinforcement Learning and Planning (BURLAP) java code library was used to solve the MDPs using three different techniques: value iteration, policy iteration, and Q-learning. Finally, the results were analyzed and compared.

II. Description of MDPs

Both MDPs were traditional grid worlds where a single agent can move north, south, east, or west in search of the goal/terminating state.

A. Small MDP

The small grid world was the Four Rooms Grid World delivered with BURLAP. It is pictured in Figure 1. It is an 11x11 grid with four rooms, each with two doorways that the agent could exit or enter through. The goal was in the upper right corner while the agent began in the lower left corner. The agent had to navigate through the rooms to find the goal.



```
wallToSouth(agent0)
wallToWest(agent0)
```

Figure 1: Four Rooms Grid World

B. Large MDP

The larger MDP was similar to the small MDP, but with a larger grid (50x50) and with a different layout. The most interesting parts about this grid world were the two terminating states that had no additional value in the upper left and lower right corners that were easy to reach while the terminating state with a large reward was more difficult to reach. See Figure 2. This configuration was selected so that the value of the reward function could be varied so that different behaviors could be produced.

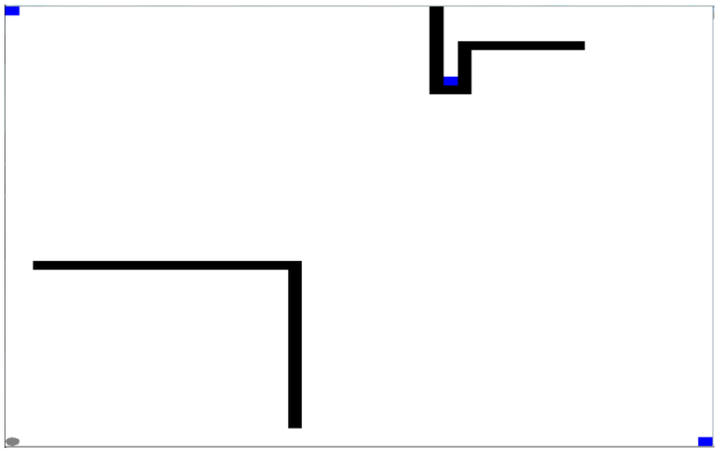


Figure 2: Large MDP Grid World

III. Solving MDPs using Value Iteration

A. Small MDP

Value iteration was run twice on the small MDP. The first time there was no stochasticity, so everything was deterministic. The results can be seen in Figure 3 where the utilities and the policy have been overlaid on the grid world. Notice that the policy always directs the agent towards increasing utility values. Since everything was deterministic, some paths had the same utility, namely when the minimum distances to the goal were the same. In these cases, value iteration allowed the agent to go in any of the directions with the greatest utility.

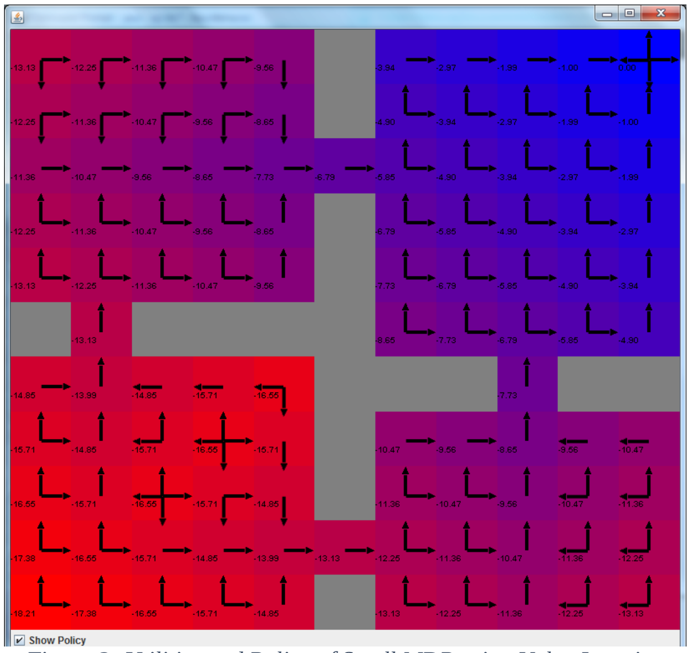


Figure 3: Utilities and Policy of Small MDP using Value Iteration with no stochasticity

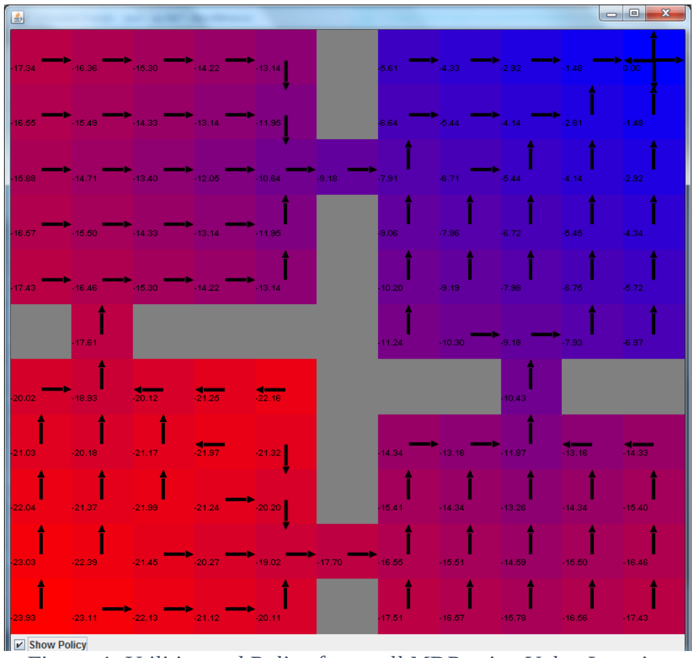


Figure 4: Utilities and Policy for small MDP using Value Iteration with Success Rate = 80%

The second time the small MDP was run, a stochasticity factor of 0.8 was introduced. One out of five moves resulted in a random direction. The results were similar to when there was no stochasticity, except that now none of the paths had the same utility, so the agent had no choice which action to take when in a certain state. This is a reasonable result. Since the process was stochastic, it was virtually impossible for the same sequence of actions to result in the same path through the grid world. As such, the utilities along paths that were an equal number of steps away from the goal in a deterministic world did not have the same utility as in a stochastic world.

B. Large MDP

The large MDP was run four times, varying whether the MDP was deterministic or stochastic and the value of the reward function. The first test was a deterministic MDP with a slightly negative reward function for each step and a very large reward at the location (31,41). Figure 5 presents these results. Since there were significantly more states in this MDP than in the small MDP, the policy was not visible when overlaid on the map. As such, please rely on the color scheme. Deeper blue states have greater utility, while deeper red states have less utility. The darkest red states are terminating states. As can be seen, the color of the states start from red in the lower left corner and fade into blue as the goal at (31, 41) is approached. Value iteration successfully found the optimal policy to the goal with the large reward with the agent taking about 100 steps to reach the goal.

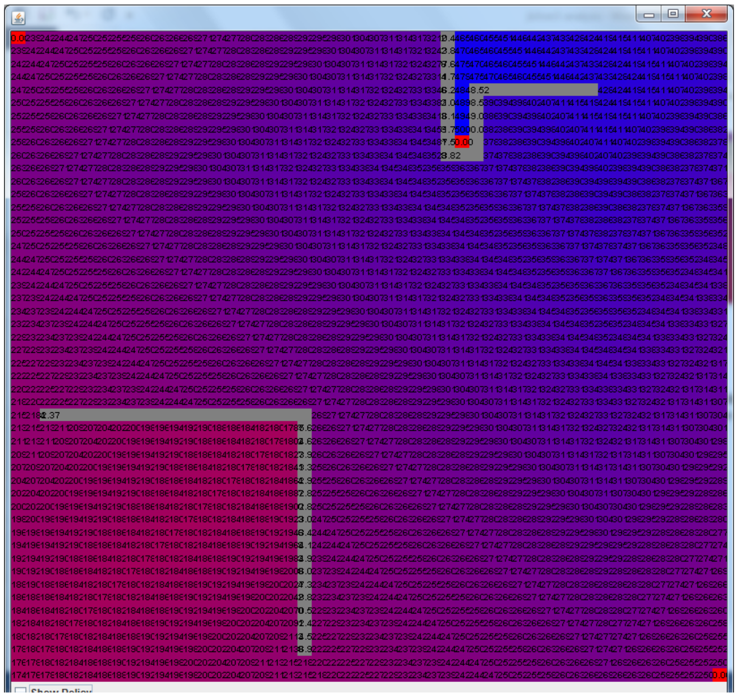


Figure 5: Utilities of states for Large MDP using value iteration with small negative reward function and large final reward

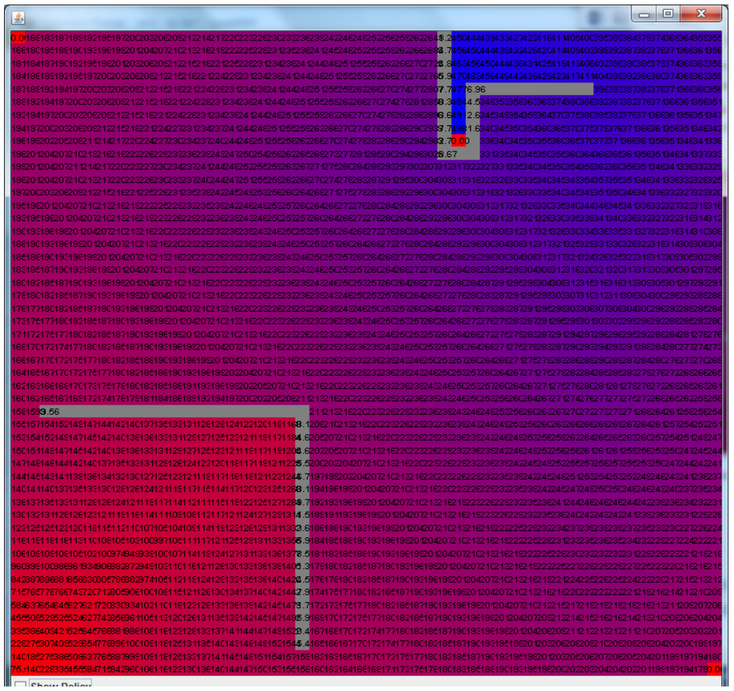


Figure 6: Utilities of states for Large nondeterministic MDP using value iteration with small negative rewards and a large final reward

The second test used the same MDP environment, but the agent's action was changed to result in a random direction 20% of the time. The results can be seen in Figure 6, which closely resembles Figure 5. The colors of the states transition from dark red in the lower left to dark blue around the terminal state in the upper right. The only difference is that the utilities were noticeably less for all the states. For instance, the state one step away from the goal in the upper right had a utility of 5000 (the value of the goal state) in the deterministic case and 4981 in the nondeterministic case. This was reasonable since in the nondeterministic case there was some probability that even if the agent wanted to go south, the action that resulted would not be south. As such, the expected utility cannot be 5000. The same argument holds for the other states in this MDP.

For the next two tests, the reward function was changed so that rather than a reward of -1 for each state transition, a reward of -500 was dispensed instead. Since the terminating state with the large reward was so far away that the steps necessary to reach it would have more than canceled out the reward for reaching that state, it was expected that the agent would instead seek to end the world as quickly as possible, thus seeking out either of the terminating states in the upper left or lower right corners. These assumptions were confirmed as can be seen in Figure 7 and Figure 8. There are blue states around all the terminating states. While states around the goal in the upper right are the darkest blue, they are somewhat isolated from the agent's starting point. In other words, the agent could not reach the terminating state in the upper right because it would have to follow a path of decreasing utility, which essentially would be not maximizing the agent's long term expected reward.

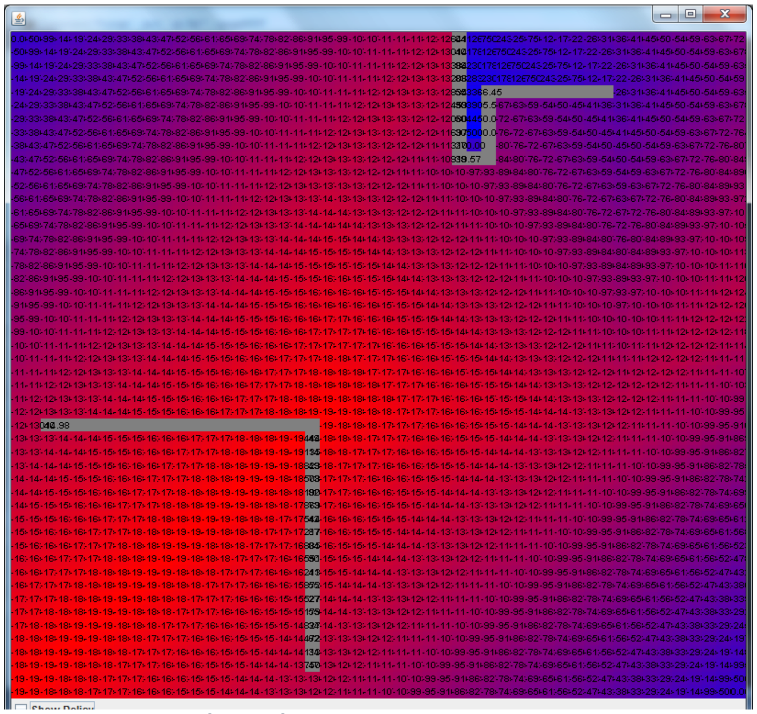


Figure 7: Utilities of states for deterministic Large MDP with large negative reward function

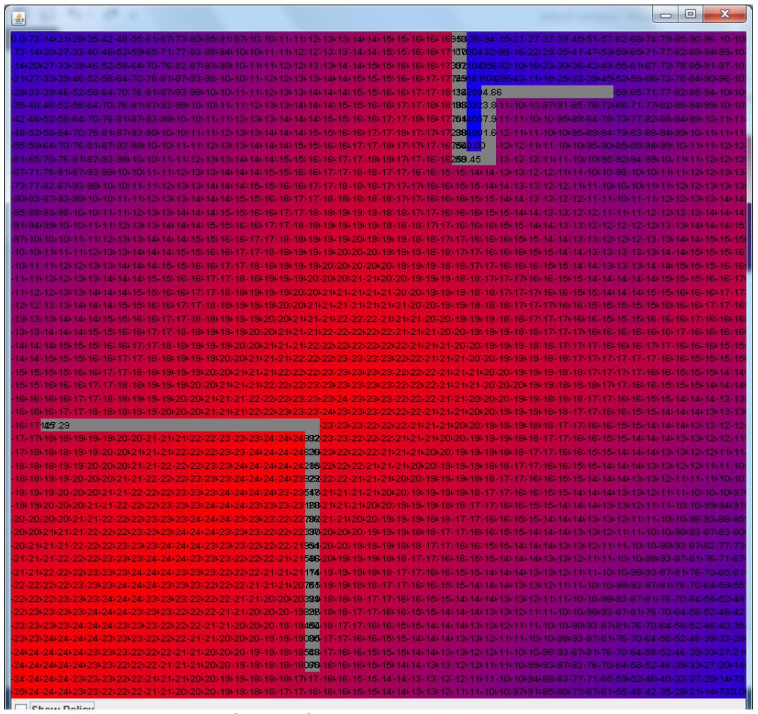


Figure 8: Utilities of states for nondeterministic Large MDP with large negative reward function

IV. Solving MDPs using Policy Iteration

A. Small MDP

As with value iteration, policy iteration was run twice, once completely deterministically and once with stochasticity. For the first test, a deterministic world was used and the results using policy iteration were exactly the same as with value iteration. The utilities and the final policies were identical. Compare Figure 9 and Figure 3. This was reasonable since the two grid worlds were identical and both algorithms were planning algorithms, so both had information of the whole grid world when creating a policy. The policies and utilities using different methods resulted in the same answers.

With stochasticity introduced into the grid world, it was expected that the results from policy iteration would closely match those of value iteration but not match entirely since the both worlds contained randomness. However, in comparing Figure 4 and Figure 10, it can be seen that the results were again identical. The same utility values and the same policies result from both value iteration and policy iteration.

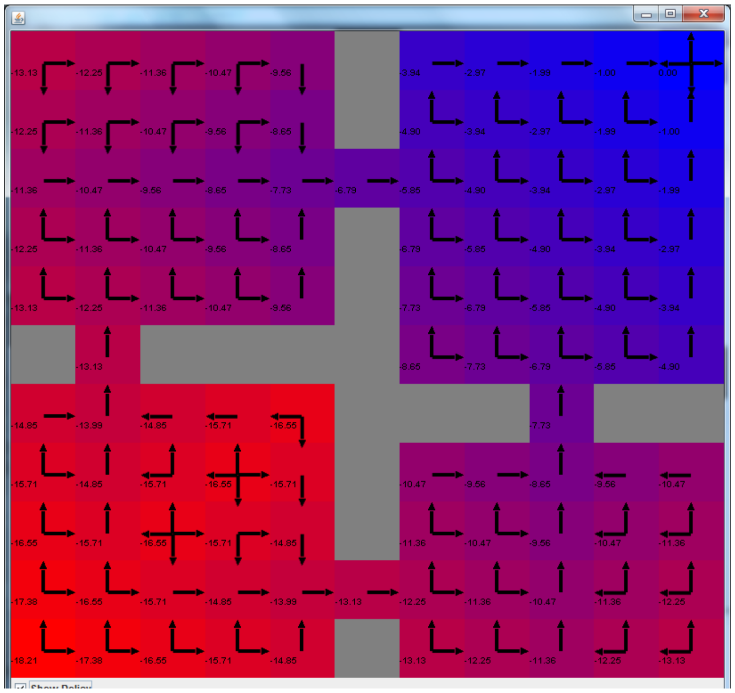


Figure 9: Utilities and Policy for deterministic Small MDP using policy iteration

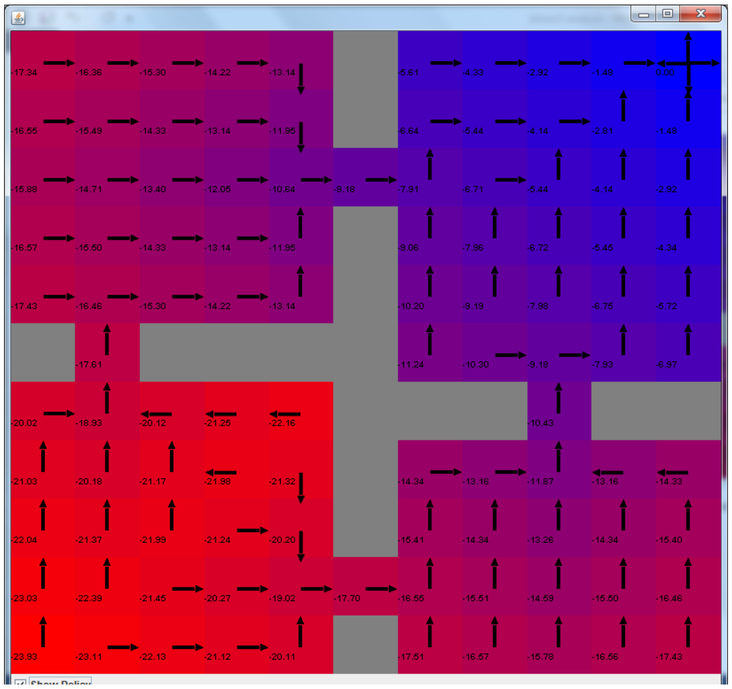


Figure 10: Utilities and Policy for nondeterministic Small MDP using policy iteration with Success Rate = 80%

B. Large MDP

The large MDP was run four times, varying whether or not the MDP was stochastic and the value of the reward function for each transition. The first test was a deterministic MDP with a slightly negative reward function for each step and a very large reward at the location in the upper right at position (31,41). The results obtained in Figure 11 were identical to those obtained using value iteration in Figure 5. The only difference was that policy iteration took more time to complete than value iteration did. Once again, the policy was unable to be displayed for the whole grid world, but the deeper the blue, the greater the utility the state possessed. As can be observed, the terminating state in the upper right corner is a "sink" that all the other states tend toward.

The second test used the same reward function but added a nondeterministic factor such that 20% of the agent's actions resulted in a random direction. These results can be seen in Figure 12. They were identical to those in Figure 6 that were created using value iteration. They were very similar to the results from the deterministic world, but all the states had a lower utility since the agent's actions were somewhat unpredictable.

The third test reinstated the deterministic world but altered the reward function so that the agent received a very large negative reward upon each transition. The utilities (and accompanying policy) changed dramatically as compared to the original reward function. Now the agent simply wanted to end the world as quickly as possible by reaching one of the closest terminating states in the upper left and lower right corners. The cost in reaching the goal in the upper right corner would have greatly outweighed the reward for reaching that state. It is significant to point out that the results obtained in Figure 13 using policy iteration were identical to the results in Figure 7 using value iteration.

The final test involved making the grid world from the third test nondeterministic. These results are presented in Figure 14. They were identical to the results using value iteration seen in Figure 8. They were also very similar to the results from the deterministic world, only that the utilities were all less because of the uncertainty of the agent's actions.

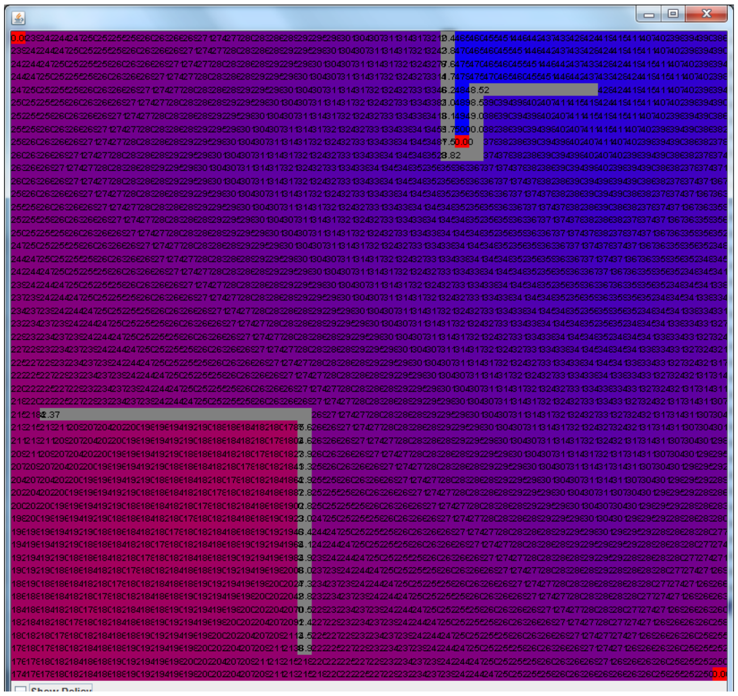


Figure 11: Utilities of states for deterministic Large MDP with small negative reward function

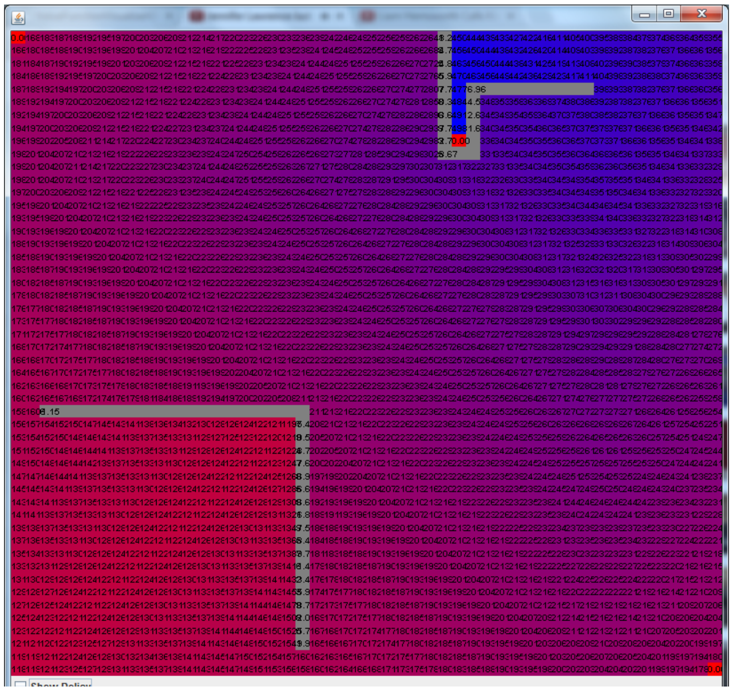


Figure 12: Utilities of states for nondeterministic Large MDP with small negative reward function

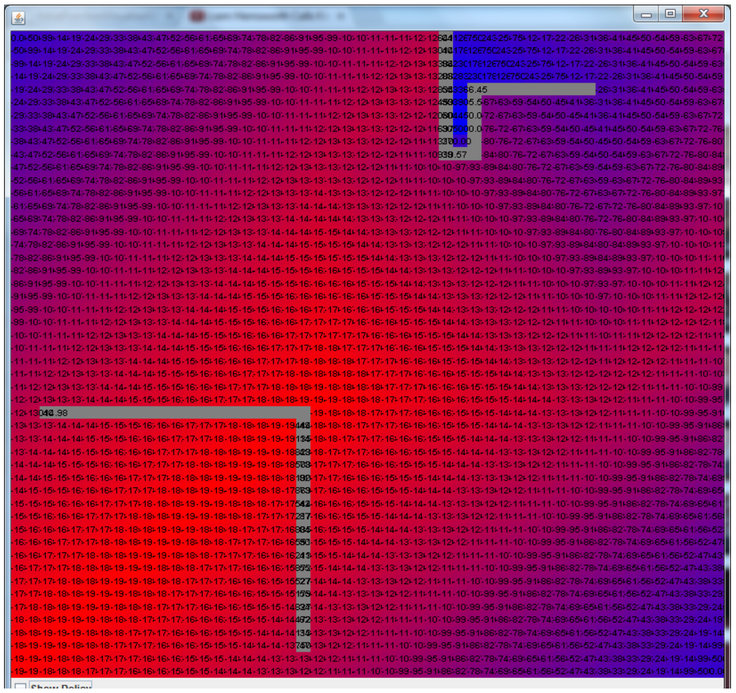


Figure 13: Utilities of states for deterministic Large MDP with large negative reward function

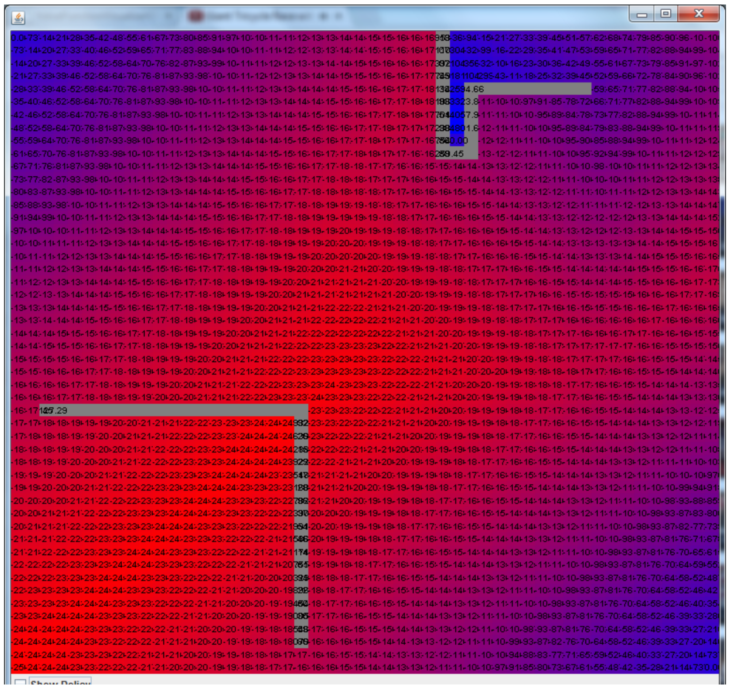


Figure 14: Utilities of states for nondeterministic Large MDP with large negative reward function

V. Solving MDPs using Q-learning

A. Small MDP

Q-learning was applied twice to the small grid world MDP. The first time the agent's actions were completely deterministic. Unfortunately the tool used did not offer a means of visualizing the Q values generated by Q-learning. However, by examining a plot of the number of steps taken in each episode as in Figure 15, the behavior of the agent during Q learning can be surmised. For early episodes, the agent took many steps before reaching the goal state. These early episodes constituted the exploration phase of the agent when it was gathering information about the world. As the agent gathered information, it started to exploit the information it had gained, and thus took fewer and fewer steps to reach the goal. The final policy did not exactly match that of value or policy iteration, but this was expected. In order for them to match, Q-learning would have had to visit each state infinitely many times. By the end, the agent trained via Q-learning took about 25 steps to find the goal, while the agent trained using value and policy iteration took 20.

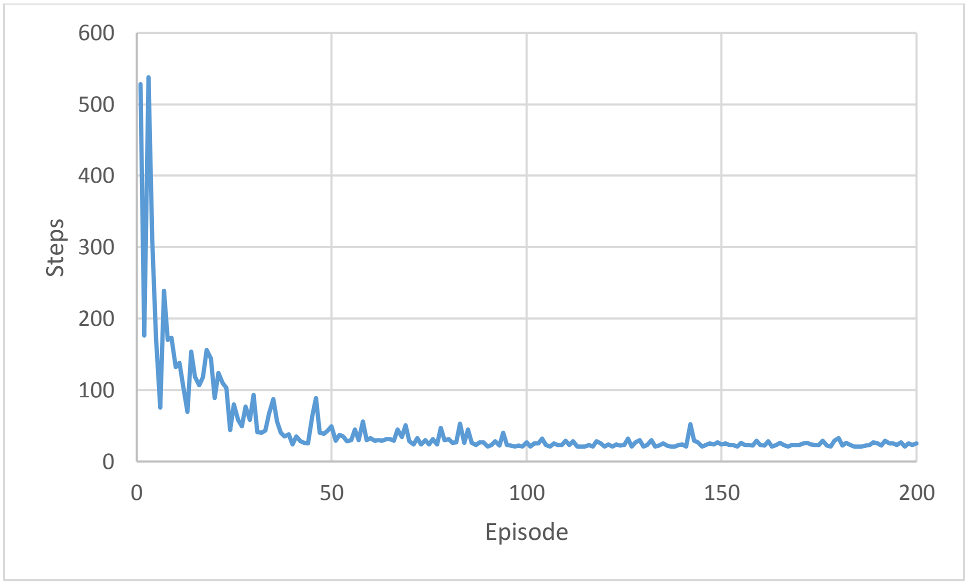


Figure 15: Number of steps taken to reach goal during Q-learning in deterministic small MDP

The second time Q learning was applied to the small grid world MDP the agent executed its desired action 80% of the time while the other 20% of the time a random action was executed. The number of steps taken for each of 2000 episodes can be seen in Figure 16. As can be seen, the number of steps does not converge as quickly nor as definitively as in the completely deterministic world. The average number of steps taken to reach the goal for the last several hundred episodes was 58, which is significantly higher than in the deterministic case.

It should be noted that Q learning took much longer than value or policy iteration to converge. This is reasonable since both value and policy iteration were planners, meaning they had knowledge of the world when creating a policy, while Q-learning did not have any knowledge of the world except that which it gained from taking an action and observing the result. Since Q-learning had this extra step (and it is quite a significant extra step) the additional time was reasonable.

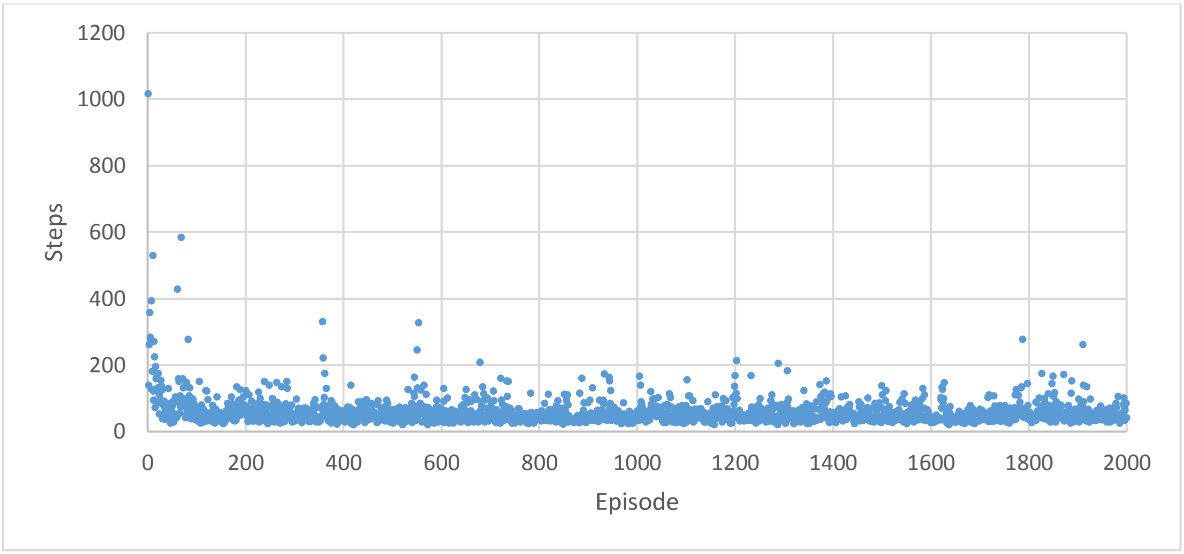


Figure 16: Stochasticity = 0.8

B. Large MDP

While the Large MDP was tested four times for previous algorithms, only two tests using Q-learning will be presented in detail here.

The first test using Q-learning placed a large positive reward at the upper right of the grid world and a small negative reward function for the other states. It was expected that the agent would find the goal in the upper right and would choose a policy that directed the agent to that state. However, the results obtained in Figure 17 do not reflect this. In order to reach the goal in the upper right the agent would have to take at least 100 steps, but the results in Figure 17 show the agent settled on around 50 steps. This indicates that the agent chose to terminate the world by reaching one of the other two terminating states. This is an instance of Q-learning overfitting; it trusted its findings too much and did not have a good balance between exploration and exploitation. The same test was conducted but with the success rate of the agent's desired action set to 80%, but the same results were obtained. The agent did not find the terminating state with the large final reward in the upper right corner.

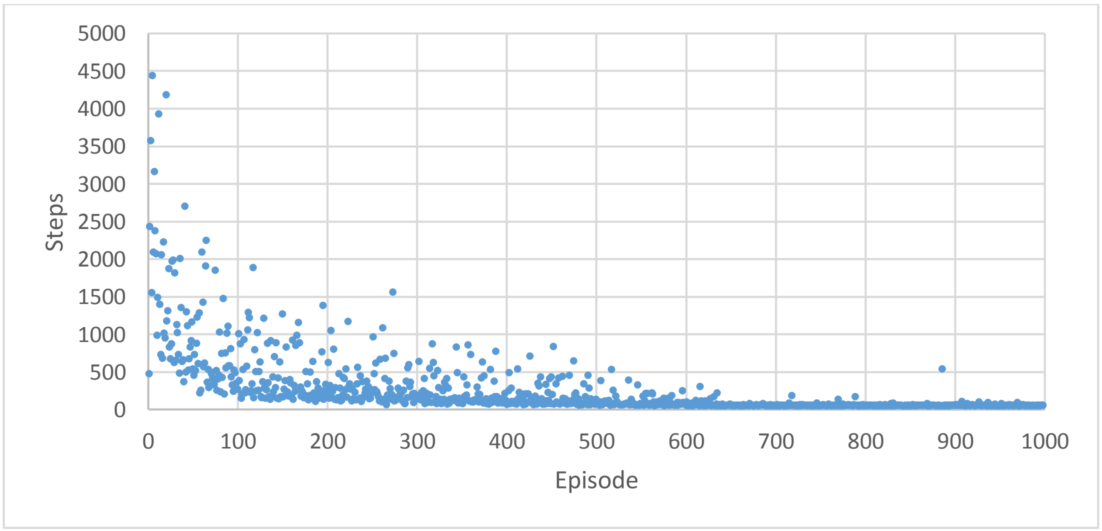


Figure 17: Deterministic Large MDP with large final positive reward

The second test still had a deterministic agent, but the reward function was changed so that each transition resulted in a large negative payoff. Thus, the results obtained in the first test were actually expected. The reward for finding the goal state with the large reward was outweighed by the cost to reach it so the agent was better off ending the game as quickly as possible. The same results were obtained when the agent's actions were nondeterministic, with the exception that more steps were required to reach the terminating state.

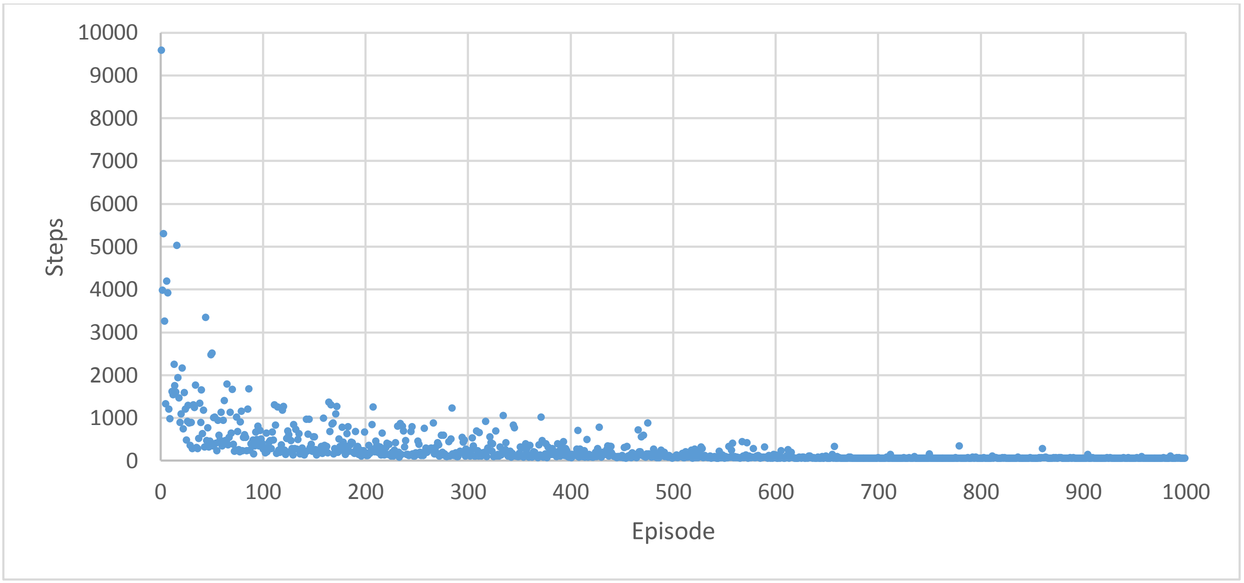


Figure 18: Deterministic Large MDP with large negative reward function for each transition

VI. Analysis

It was not surprising that value and policy iteration resulted in the same policy for all the MDPs tested. Value and policy iteration could have generated the same policy even with different state values. What was surprising was that value and policy iteration converged to the exact same values (at least to the number of significant figures the tool output). Upon closer examination of the values passed to the value and policy iteration algorithms, these results were reasonable. Both value and policy iteration used the same equation, the Bellman equation, to compute the utilities of the MDP states and the resulting policy. The terminating criterion passed to both these algorithms was the same, so identical results were reasonable.

Policy iteration took noticeably longer to converge than value iteration. This was unexpected since both value and policy iteration used the same Bellman equation. However, policy iteration had the additional step of iteratively improving the policy estimate. In contrast, value iteration simply computed the value of the states in each iteration. Once the values converged, then the policy was selected. Essentially policy iteration had an additional step at each iteration, so it was reasonable that it took longer to converge than value iteration.

As mentioned earlier, Q-learning took longer than value or policy iteration to converge because it had to learn the environment rather than knowing it beforehand. However, with the small MDP in particular (because the large MDP did not always converge on the proper state), Q-learning converged must faster and more definitively without stochasticity in the agent's actions than with stochasticity. While it was expected that the number of steps required to reach the goal with the agent's actions being stochastic would be greater than when they were entirely deterministic, it was not expected for the average number of steps to more than double. Furthermore, there were a number of cases that took several hundred steps to converge, even after more than 1000 iterations. Meanwhile, without stochasticity the agent took an average of 25 steps to reach the goal after only 100 iterations. It would appear that Q-learning did not make a global policy. This was not as apparent for the deterministic case because the agent did not venture away from where the ideal policy was generated. However, for the nondeterministic case the agent sometimes entered a state where the ideal policy had not been generated and wandered around a great deal before finally either finding the terminating state or landing in a state that had a well-defined policy. This was a side-effect of the limitations of this experiment. If all the states were visited infinitely often, the Q-values would have converged to the same values as in value and policy iteration and a global policy would have resulted. For obvious reasons, this was not possible, but could perhaps be approximated by varying the parameters passed to the Q-Learning algorithms.

The most interesting experiment was that of Q-learning with the Large MDP with the large positive final reward and the small negative reward function for the other states. The agent did not find the terminating state with the large reward, and so it sought to maximize its final reward by reaching one of the two terminating states that had no reward. The agent did not adequately explore the environment before exploiting what it had learned.

VII. Conclusions and Future Work

Value and policy iteration outperformed Q-learning both in terms of accuracy and execution time. However, they made assumptions that are not always available, namely having full knowledge of the environment before creating a policy. Q-learning was able to approximate the optimal policy in some cases, but had difficulty when the problem space was large.

Since value and policy iteration already gave optimal results, there is not much future work that can be performed with them. However, it would be interesting to change the terminating criteria for only one of these algorithms and see if they result in the same policy even if the utilities of the states are different.

In contrast to value and policy iteration, there are many future avenues of experimentation for Q-learning. For instance, the initial Q value, learning rate, and discount were held constant for all the experiments reported here. These parameters could be changed and the differences noted. Finding a way to have Q-learning converge upon the optimal policy for the Large MDP with large positive terminal reward and small negative reward for all other transitions should be of primary concern. One obvious solution would be to remove the other terminal states in the upper left and lower right that are easy to reach. However, it would be ideal to find a set of parameters to pass to the Q-learning algorithm that would allow the agent to converge upon the ideal policy even with the other terminal states in place. It may be the case that the exploration strategy used for the Q-learning algorithm needs to be modified to allow for more exploration.