CS 4641: Markov Decision Processes

Bradley Reardon

April 14, 2019

1 Introduction

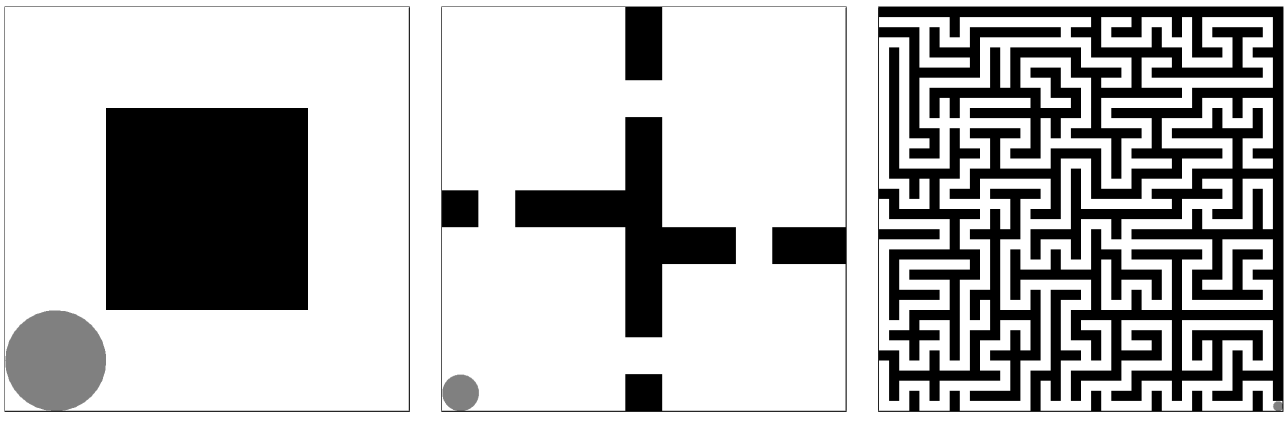
In this paper, we will be exploring the use of Markov Decision Processes and Reinforcement Learning to model and find optimal solutions for specific problem spaces. For the purpose of analysis, this report will cover applying various algorithms to two problems, which are completing a level of a classic Block Dude game, and navigating through a grid world. In exploring Markov Decision Processes, we will evaluate both problems using both value iteration and policy iteration. Afterward, we will explore each algorithm further, by applying Reinforcement Learning to the algorithms using Q-Learning.

2 Grid World

2.1 Domain

We will begin by discussing the Grid World problem. In the context of a Markov Decision Process, this problem space is relatively small. For each given position on the grid, there are a maximum of only 4 possible actions, which are movements to the north, south, east, and west. In addition, the goal is fixed and only requires movement actions to transitition to the goal state. Lastly, the grid's wall positions are fixed and do not change as a result of any transition.

Because of the properties of this problem space specifically, the Grid World problem can be hypothesized to converge to an optimal solution relatively quickly. Figure 2.1 shows the three grid world "levels" that will be tested for the grid world domain.



(a) Level 1: Single block **(b)** Level 2: Four rooms **(c)** Level 3: Maze

Figure 2.1: Grid World levels

This problem is particularly interesting due to its relationship to path-finding. Since we're evaluating the best way to go through a known world, this problem has connections to robotics and artificial intelligence.

2.2 Value Iteration

Each of the three levels in the grid world experiment were tested using value iteration. Value iteration was run with a termination reward of 5.0, a step reward of -0.1, and a discount of 0.99. Table 2.1 shows the statistics collected for each level.

	L1: Single Block	**L2: Four Rooms**	**L3: Maze**
States	12	104	800
Iterations	11	23	188
Steps Taken	7	29	212
Runtime	33ms	123ms	1.451s

Table 2.1: Grid World: Value iteration results

2.3 Policy Iteration

After being run with value iteration, the experiment was run again using policy iteration. Policy iteration was run with the same parameters as value iteration, with a termination reward of 5.0, a step reward of -0.1, and a discount of 0.99. These values were chosen to evaluate consistency across the two algorithms, and to see which algorithm performs better under the same conditions. Table 2.2 shows the results for this run.

	L1: Single Block	**L2: Four Rooms**	**L3: Maze**
States	12	104	800
Iterations	4	7	15
Steps Taken	8	22	204
Runtime	185ms	696ms	11.066s

Table 2.2: Grid World: Policy iteration results

2.4 Q-Learning

Table 2.3 shows the results for running Q-Learning on all three levels. In this case, the chosen learning rate was 0.1, the chosen qInit was 100, and episilon was 0.1. All three world models were trained over 2500 episodes. Interestingly, the maze world failed to complete the experiment and would stall when attempting to plan a path using the model. As such, these results will be denoted "DNF".

	L1: Single Block	**L2: Four Rooms**	**L3: Maze**
Steps Taken	6	25	DNF
Runtime	178ms	426ms	DNF

Table 2.3: Grid World: Q-Learning results

2.5 Discussion

Interestingly, these results suggest that value iteration is much faster on average at higher degrees of state complexity than policy iteration. However, most noticably, the maze level has a more optimial solution when policy iteration is used (204 steps) as opposed to value iteration (212 steps). Though the solution is marginally better for the maze world in the case of policy iteration, the 7.62-times increase in the runtime for the same world appears not to be worth the marginal improvement in solution optimization. This note will be revisited after experimenting with the Block Dude problem.

As for Q-Learning, it outperformed policy iteration for levels 1 and 2, though it did fail to complete for the maze level. In the case of level 1, the solution contained the fewest steps yet, while in the case of level 2, the solution only outperformed that of value iteration. Overall, however, reinforcement learning via Q-Learning seems to be a

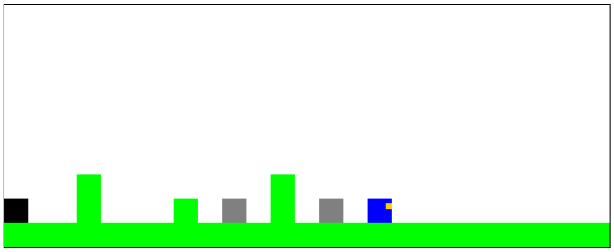
reasonable choice for finding quick solutions to the Grid World problem, as long as the state space isn't too large. However, as a larger state space indicates larger complexity overall, and my computer has performance bottlenecks due to its age, it is impossible for me to make a determination as to whether Q-Learning is a good choice for runs on larger state spaces such as the Grid World maze.

3 Block Dude

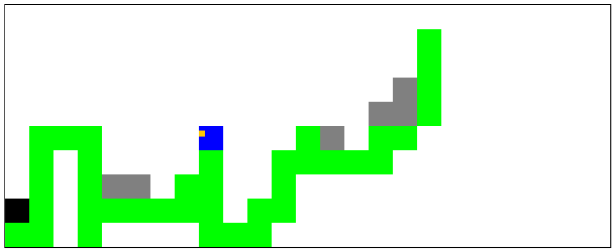
3.1 Domain

Next, we will evaluate the Block Dude game in the context of a Markov Decision Process. In this world, the "block dude" character has the ability to move around east, west, and up. In addition, gray blocks in the world are able to be picked up and placed down again in other locations. The addition of changing the world, in addition to the interest in the path-finding nature of the Grid World problem, makes the Block Dude game more complex than Grid World.

In this section, we will be evaluating the same algorithms in two different Block Dude levels, which can be seen in Figure 3.1.



(a) Level 1: Easy



(b) Level 2: Hard

Figure 3.1: Block Dude levels

As demonstrated by both levels, the nature of this game requires that the player change the terrain in order to reach the goal, making this a very interesting problem to explore in this context.

3.2 Value Iteration

Table 3.1 shows the results for running value iteration on the easy and hard levels of Block Dude. It is instantly apparent based on the overall number of states that even the easy level of Block Dude exceeds the maze level of Grid World in terms of overall complexity. Value iteration in this case was again run with a termination reward of 5.0, a step reward of -0.1, and a discount of 0.99.

	L1: Easy	**L2: Hard**
States	841	14200
Iterations	230	230
Steps Taken	19	94
Runtime	4.016s	66.637s

Table 3.1: Block Dude: Value iteration results

3.3 Policy Iteration

Next, policy iteration was run. The overall choice in parameters was again the same to keep consistent results in evaluating policy iteration versus value iteration. Table 3.2 on the following page shows the results for this test.

	L1: Easy	L2: Hard
States	840	14199
Iterations	20	52
Steps Taken	19	94
Runtime	13.076s	940.171s

Table 3.2: Block Dude: Policy iteration results

3.4 Q-Learning

Unfortunately, for the Block Dude problem, a similar problem to the one above from the Grid World maze level was encountered during Q-Learning. As a result, no result was obtainable from either the easy or hard levels of the Block Dude problem.

3.5 Discussion

In the end, the results show that the Block Dude is a very single-approach game in terms of the required steps to arrive at a solution. In both successful tests using value and policy iteration respectively, both algorithms resulted in equally-optimal solutions for each level. However, the main difference between the two approaches is that per-iteration runtime is much higher in the case of policy iteration, so much so that the hard level required nearly 16 minutes to reach a solution. In this case, value iteration stands out as the clear winner in terms of overall performance, as nothing is gained from the major performance hit from taking a policy-iterative approach in solving the level.

4 Discussion

For the two problems explored in this report, value iteration was the clear-winning approach in all cases. Though in some tests policy iteration or Q-Learning resulted in solutions that took fewer steps (mainly in the case of Grid World), the performance tradeoffs of switching to a different approach in the context of these problems was not worth it.

However, this isn't to discredit the efficacy of the policy iteration and Q-Learning algorithms. Policy iteration takes many fewer iterations to converge to a solution on average than value iteration itself, suggesting that for some problems with much larger complexity spaces that policy learning may have an advantage in converging faster over time. However, this was not the case in any of our tests.

For Q-Learning, though three of the five tests failed to complete with the testing implementation used for this project, the reinforcement learning algorithm did show similar results to the other algorithms on two of the Grid World tests. With further parameter tuning and a better understanding of the implementation of this algorithm in BURLAP, this may have been able to be avoided. However, with the constraint of time, this was not able to be explored further.

5 Conclusion

In closing, I have attempted to evaluate Markov Decision Processes and Reinforcement Learning in the context of two interesting problems. Value iteration and policy iteration, despite their simple approach, proved effective in all cases explored in this report, though the performance tradeoffs of using policy iteration in this case are a point of further discussion.

Future explorations of this topic may lead to an attempt to iteratively optimize the parameters of each of these methods for learning, however the parameters used throughout the various sections of the report did prove sufficient for experimentation.