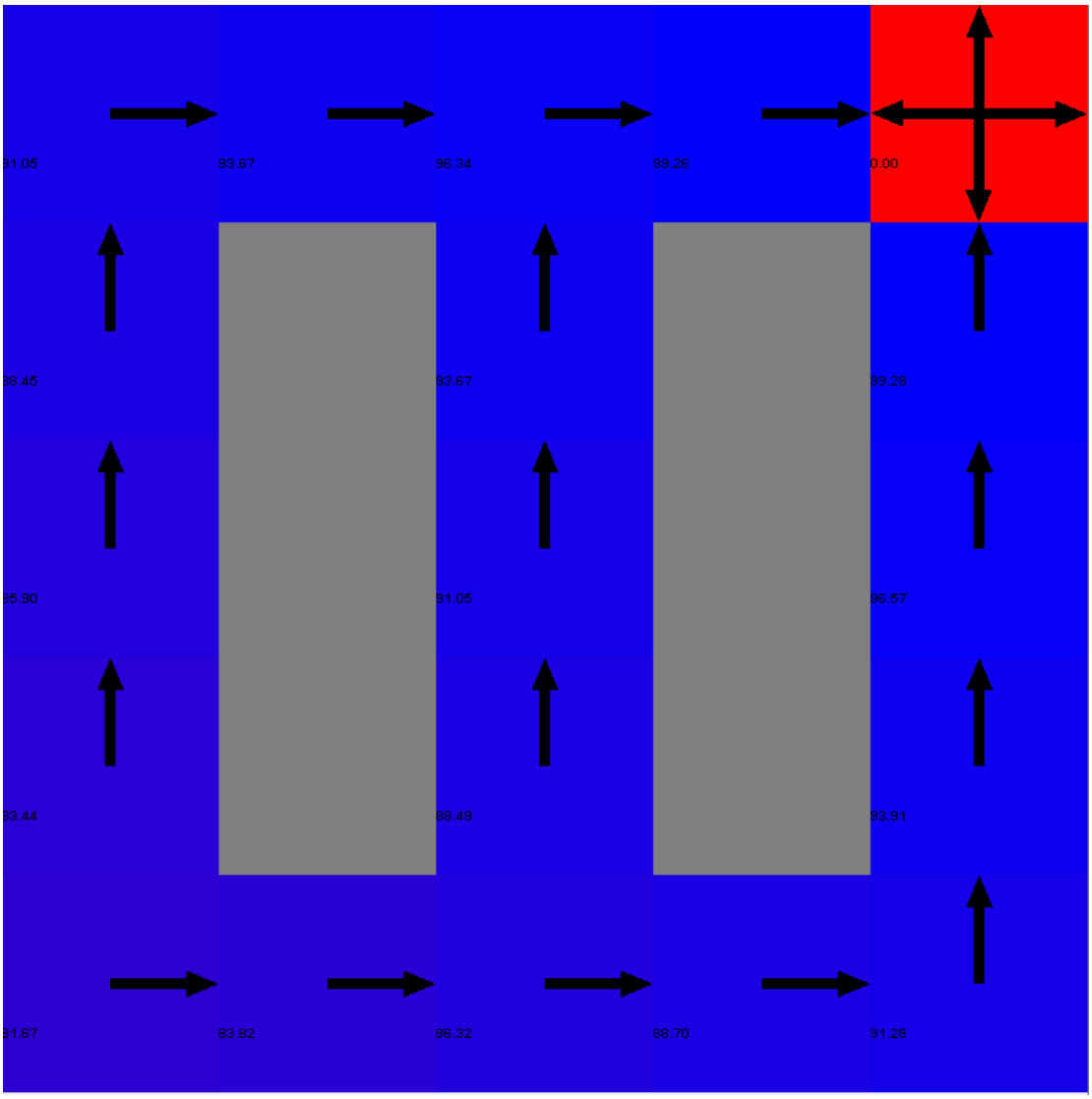
Nick Anderson
OMSCS 7641
Fall 2016
Assignment 4

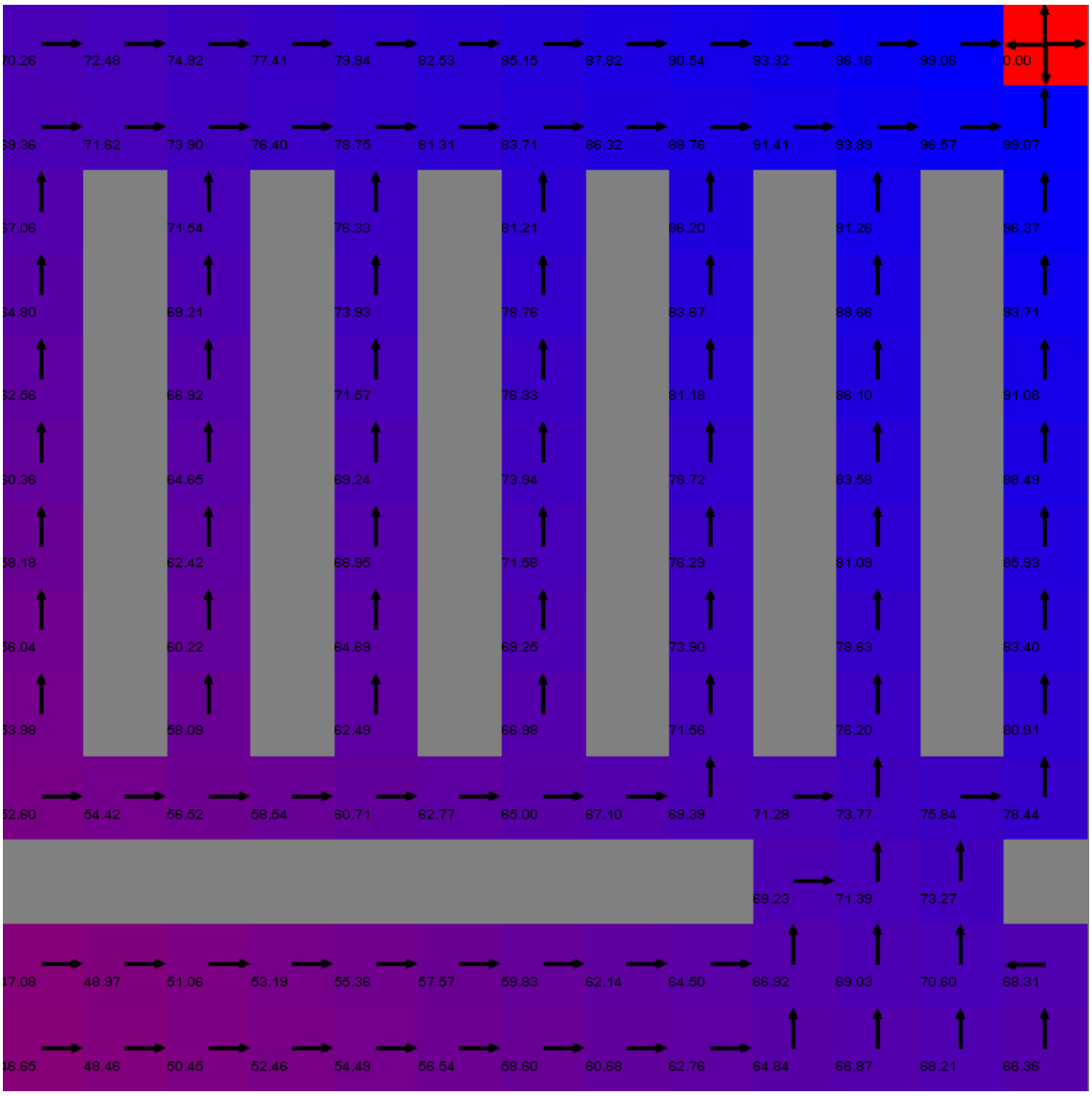
For my MDP problems, I chose two grid world problems, both with some grounding in a real situation. I work for a large manufacturing company and in my prior role, I was working at one of the largest operational warehouses in the world. It's no secret that automation and machine learning will become a bigger part of our operation in facilities such as those, so I was interested in applying what I learned about reinforcement learning to grid worlds that mimic subsections of our warehouses. Below is my grid world with a "small" number of states.



The rewards and policy shown on the grid above are those calculated by value iteration. This layout is meant to loosely mimic the rows of shelves where parts are stored in the warehouse (the resemblance will be more evident in the "large" state example). Having the ability to automatically determine optimal routes throughout our warehouse can result in significant cost savings, given that we pick millions of parts per year and our buildings and shelving structures are always being moved for optimization purposes.

Nick Anderson
OMSCS 7641
Fall 2016
Assignment 4

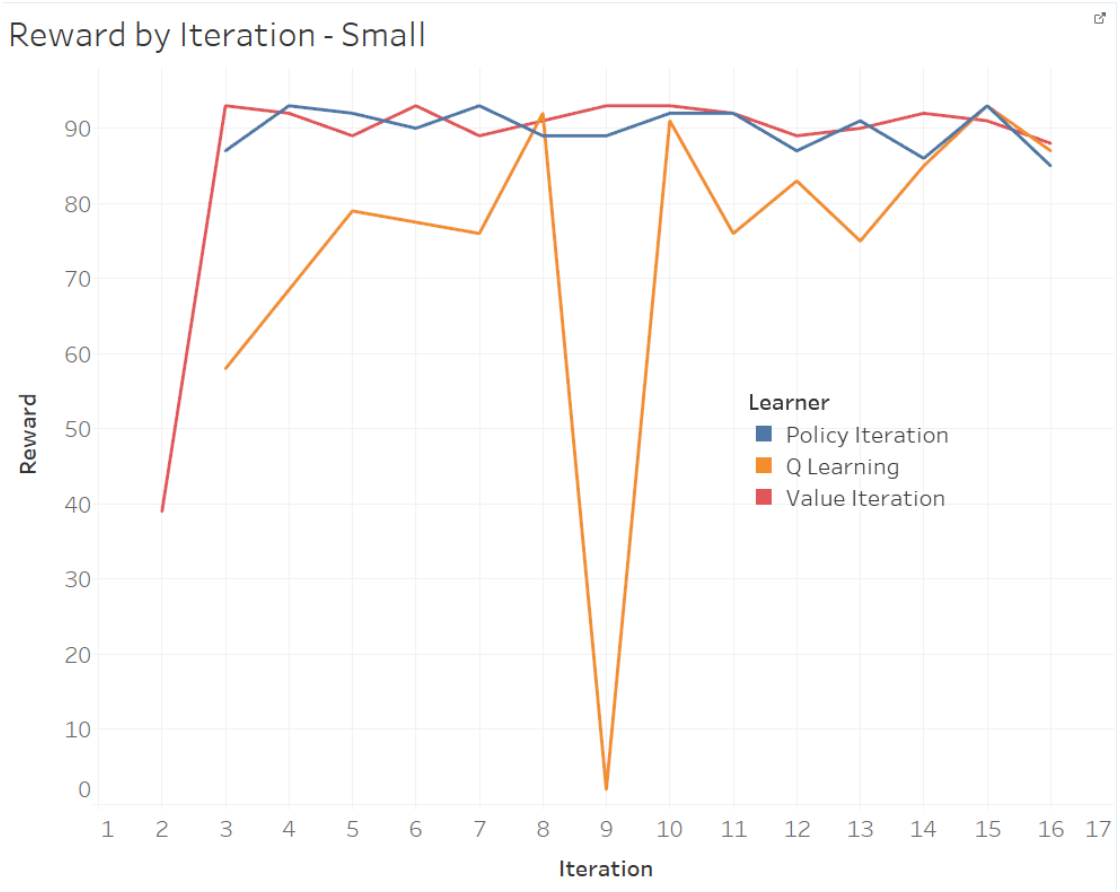
Below is the "large" state MDP problem layout. The rewards and policy on this visual are that of policy iteration as opposed to value iteration for the small state problem.



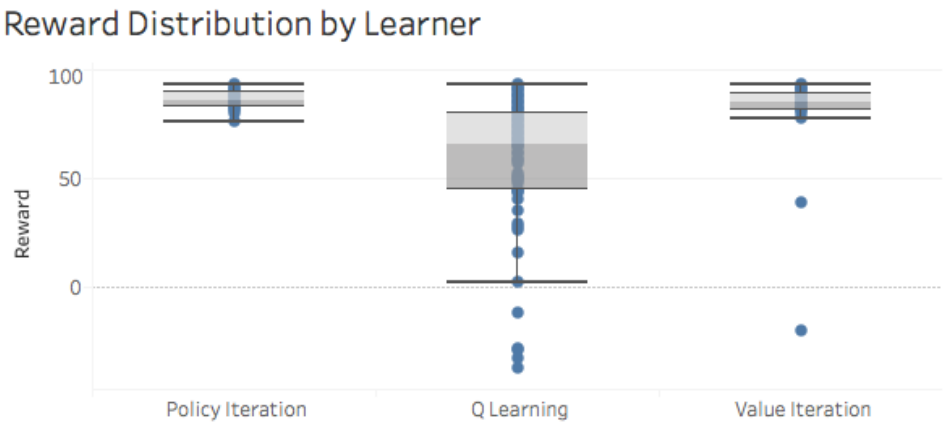
In this grid, it's easier to see the characteristics of a warehouse —starting from the bottom left, the horizontal "wall" could be seen as a wall between different buildings with the gap being the door to get from one to another. The parallel vertical lines in the upper 2/3 of the grid are "shelves" where parts are stored. One interesting thought I kept in the back of my mind while doing this analysis was: "Does the optimal path necessarily create the optimal traffic patterns?"

Next, we'll look at how my policy iteration, value iteration, and Q-Learning performed on both the large and small grid worlds.

Nick Anderson
OMSCS 7641
Fall 2016
Assignment 4

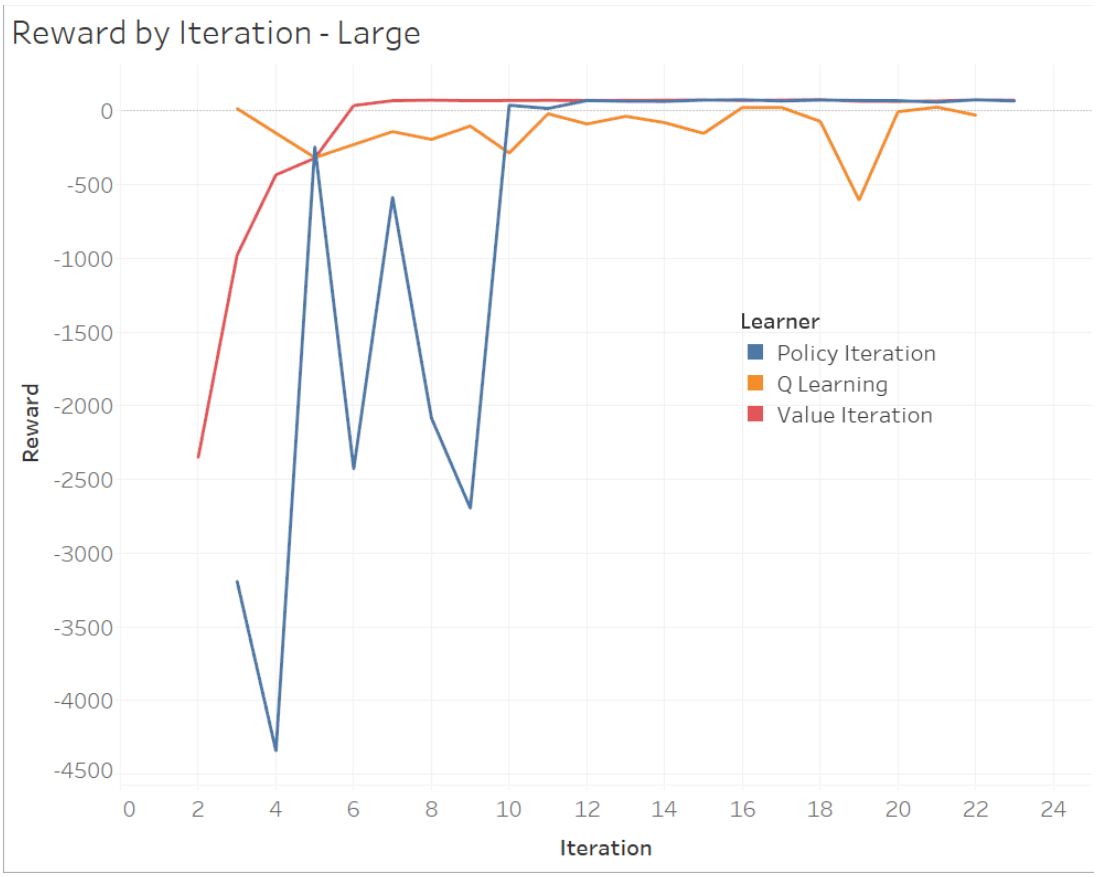


If we look at our small state MDP problem first, we can see that policy iteration and value iteration quickly converge above 90 within four and three iterations, respectively. After converging, these two learners exhibit a small amount of uncertainty, with a vast majority of observations having rewards between 85 and 93. Our reinforcement learning algorithm, Q Learning, converged after 8 iterations and showed much more variance throughout. Despite the differences in predictability, all three learners found the same maximum reward (93) throughout 500 iterations.

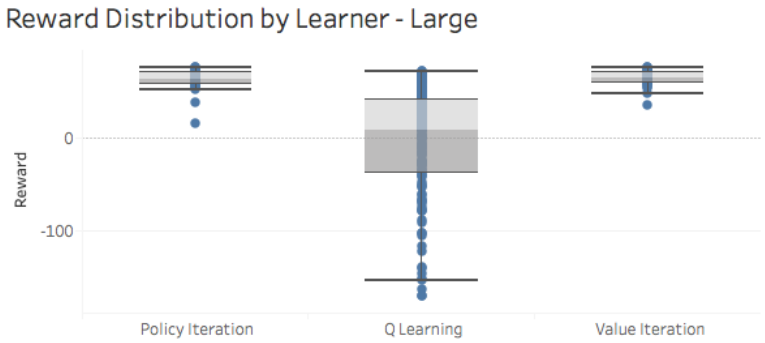


Nick Anderson
OMSCS 7641
Fall 2016
Assignment 4

Next, we'll look at the same information for the grid world with the higher number of states. Below is the reward by iteration for each learner.

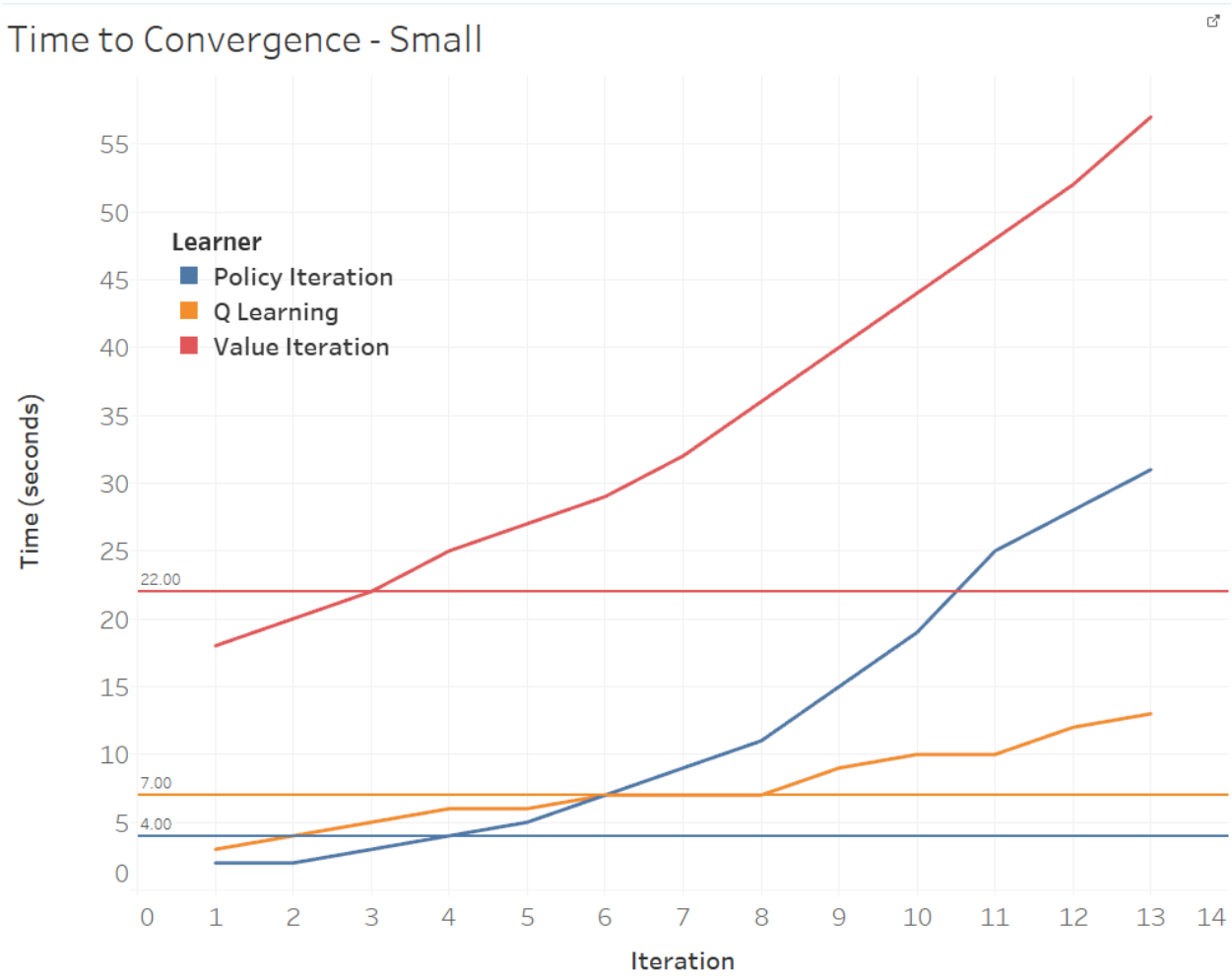


This output data is more interesting than the output from the small-state grid world for a few reasons. First, we see a difference in our policy iteration and our value iteration, however this is likely due to the different random starting places for each learner (the reward for iteration 1 of the policy iteration was so low that it was omitted because of being too low even for this y-axis). Value iteration converges to an answer around 6-7 iterations, whereas it takes policy iteration 12 iterations to converge to the same number. After 500 iterations, all learners did not converge to the same maximum rewards, either (Policy Iteration: 77; Q-Learning: 73; Value Iteration: 77).



Nick Anderson
OMSCS 7641
Fall 2016
Assignment 4

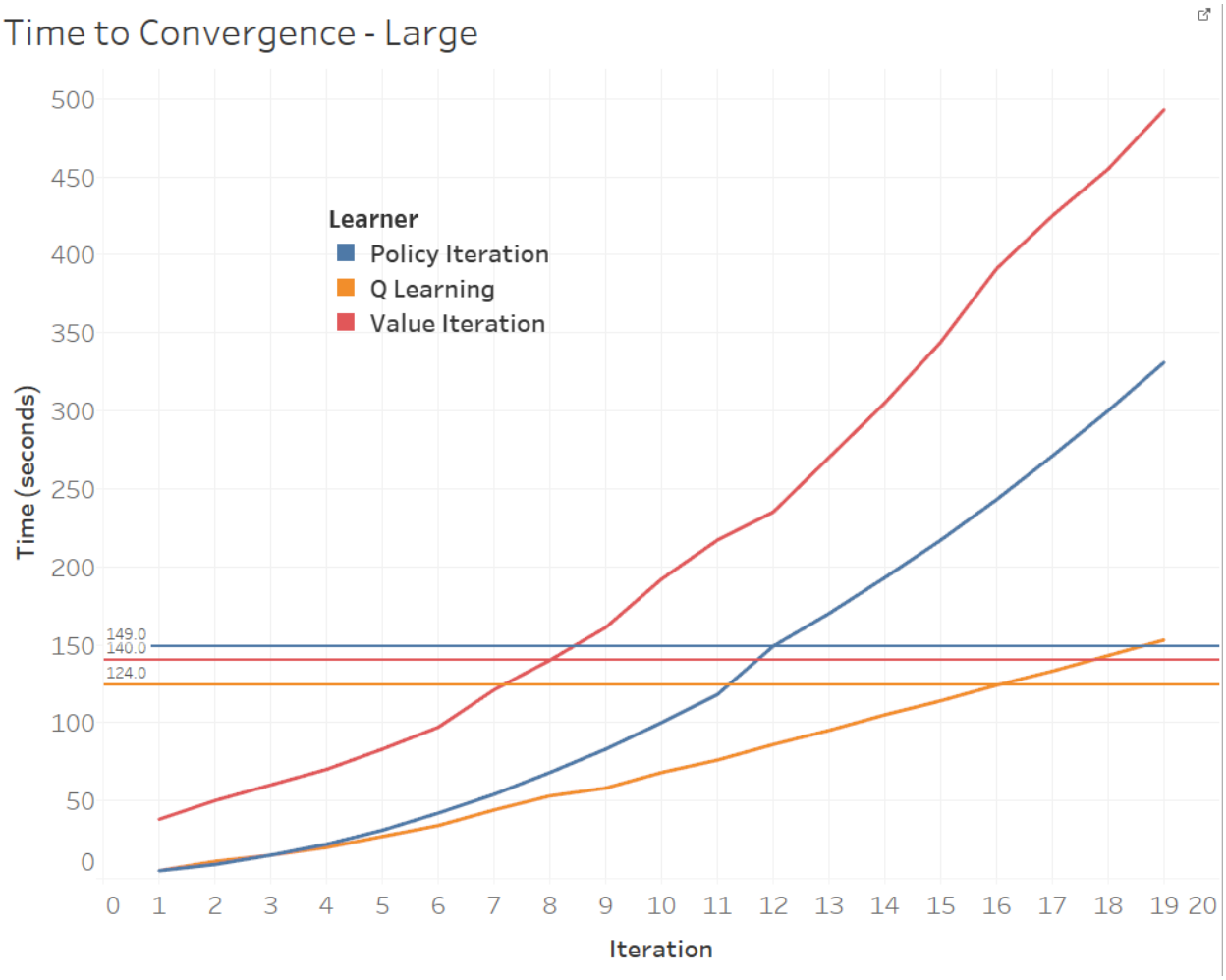
Most of our focus so far has been around how many iterations it takes for each learner to converge, but often times we are more worried about clock time from a practical, implementation perspective. First, let's look at our clock times to convergence for the small state grid world. The graph below shows the cumulative time by iteration for each learner. The horizontal lines each represent the time at which that learner (matched on color) converged.



When we first looked at the small-state problem, it appeared that policy and value iteration methods were roughly equals based on iterations to convergence. This graph shows a different story where value iteration took 5.5 times longer to converge (clock time) than policy iteration. Additionally, (and not surprisingly) Q-Learning performs relatively well and scales better than its counterparts as the number of iterations goes to infinity.

Next, we'll look at the clock times to convergence for our large-state problem:

Nick Anderson
OMSCS 7641
Fall 2016
Assignment 4



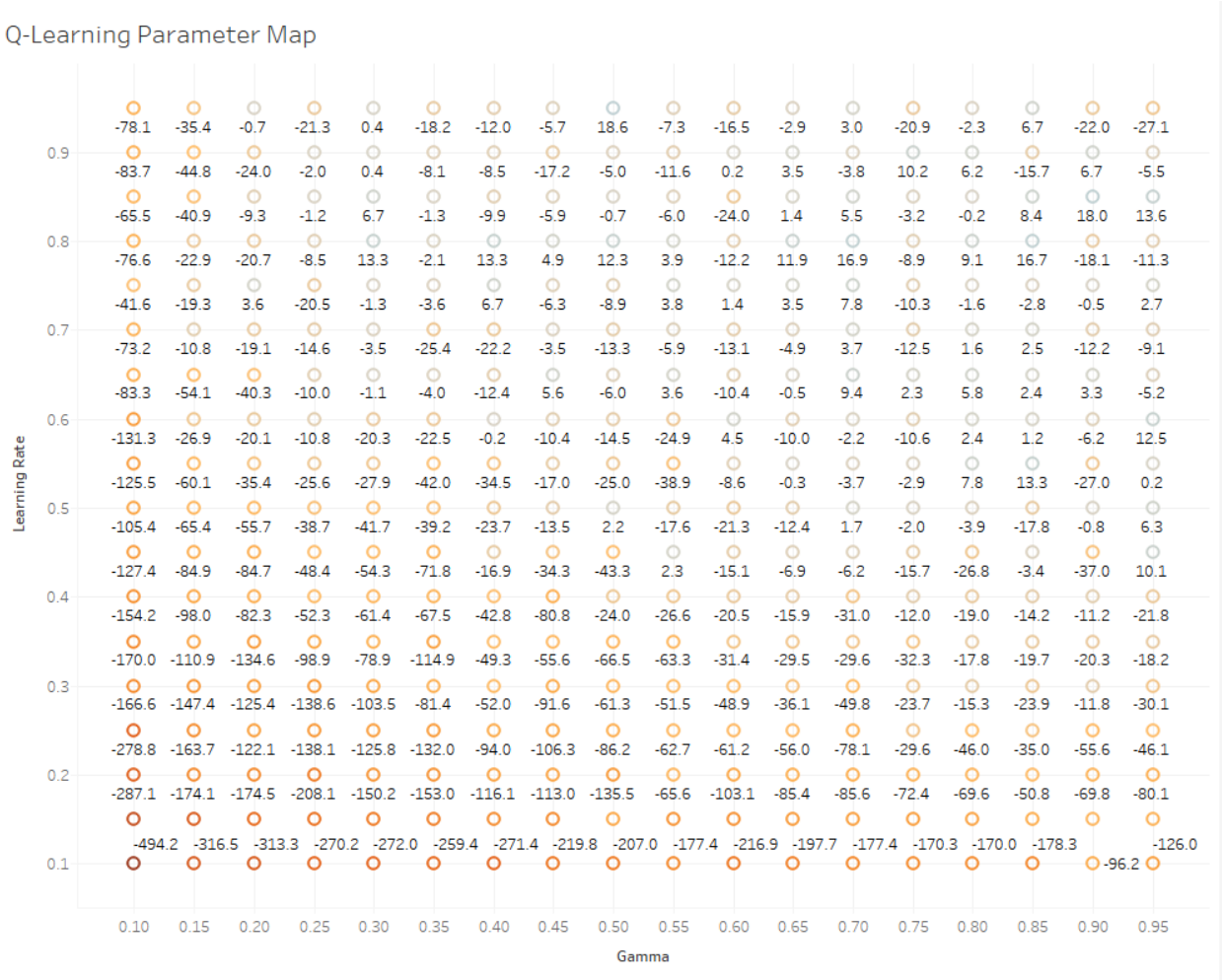
Our final clock times to convergence for each learner are Policy Iteration: 149; Value Iteration: 140; Q-Learning: 124. Value iteration appears to have a fixed cost at the beginning of execution, but then grow at the same rate as policy iteration. The times to convergence have converged (ha!) compared to the small-state problem, and the trade-off between learners starts to feel like a zero-sum game in the sense that a learner that is faster in iterating (e.g. Q-Learning) will need more iterations to catch the same amount of information as a slower algorithm that captures more information.

The main difference I see between the small and large state problems are the obvious iterations and time to convergence, and the fact that the time to convergence between the algorithms seems to converge as the number of states grows.

There is a much bigger observable difference between policy/value iteration and our reinforcement learning algorithm, Q-Learning, namely in the iterations to convergence, the large variance when it does converge (since it's constantly searching for different paths), and the fact that it often didn't hit the absolute maximum found by policy and value iteration in a reasonable number of iterations.

Nick Anderson
OMSCS 7641
Fall 2016
Assignment 4

I'd also like to address how I chose parameters for Q-Learning (gamma and learning rate). This turns out to have been a very uneventful process. Below you can see the average reward for different combinations of learning rate/gamma:



While there appear to be some higher averages, those turned out to be outliers and were not repeatable when I re-ran the program. I looked at the same visualization above by the maximum reward, and every point was at or very near the true max; when I did the same for the variance, it behaved very much the same as the averages view. I also graphed the learning rate by the reward value stratified by certain values of gamma, and this did not show any evident trend (they seemed to "converge" to their maximum state at the same rate). To me, this means the grid is easily solvable; this is partly because of the high number of iterations I allowed compared to the grid size, but even the for the grid size this should have been more easily solvable for Q-Learning due to the fact that the aisles only allow the agent to move forward (the right direction) or backwards (where it came from). Because of this, I kept both the learning rate and gamma at the default values of .99 for my analysis since those values were as successful as any other in the trials.