Title: Learning Exercises
Authors: Corey Proscia and Alexander Wang, CSC 242, University of Rochester
Date: April 21, 2006

Abstract:

Intelligent agents can maximize utility in repeated trials through learning from past experiences. We compared tabulation and formula approximation as methods of using direct utility estimation with an online exploring agent. We experimented with different maps in both stochastic and deterministic environments. We used the Java Quagent API designed by Aaron Rolett, Jason Freidman, and Chris Tice and the learning package designed by Bo Hu to simulate these learning methods in the Quagent world.

Keywords:

Quagent, Quake2UR, learning, policy iteration, Gridworld, exploration, direct utility estimation, tabulation, formula approximation, stochastic, deterministic.

Background, Motivation, and Problem:

Our initial goal was to create an intelligent agent that would build on its experiences to maximize its expected rewards, or utility, without any human interaction, in a stochastic environment. However, this required an understanding of some basic learning methods in a basic world. We looked to start with an offline method that would determine optimal strategies in a small stochastic world; the sample world we chose was nicknamed the Gridworld. We also wanted to analyze how learning methods worked with increasing complexity. As stated in Russell and Norvig (2003), we predict runtime of policy iteration in the Gridworld will increase significantly with additional number of states.

Our later goal was to Russell and Norvig (2003) define an exploration function as a function that "determines how greed (preference for high [utility]) is traded off against curiosity (preference for … actions that have not been tried often)." We sought how to best balance curiosity and greed, as well as a method to use the results from past experiences to achieve the optimal policy quickly.

Prior Work:

Teaching assistants Jason Freidman and Aaron Rolett distributed a very extensive package of Java classes that they created along with Chris Tice. This package handles socket connections and manages communications between the client and Quagent. For the Quagent problems, we modified their package to create the Quagent. We generated maps based on the map generator by Pawlicki. This generator was originally designed to handle mazes but works effectively for open maps as well.

We also used the environment simulator of the Gridworld as well as policy and value iterators provided by Bo Hu.

Gridworld Experiment Procedure:

Our first experiments were with an agent in a 3x4 (rows x columns) environment, described in Russell and Norvig (2003) as the Gridworld. Each state is a unique y,x coordinate combination. The environment contains two terminal states at (3,4) and (2,4) with rewards +1 and -1 respectively. All other states have a reward of -0.04. The location at (2,2) is inaccessible. The environment is stochastic, as the agent's movement does not always function as intended. There is a 10% chance that the agent moves in the direction 90 degrees clockwise from intended,

a 10% chance that the agent moves in the direction 90 degrees counterclockwise from intended, and an 80% chance that the agent moves as intended.

In this assignment we used the environment simulator and policy iterator created by Hu (2004). The goal was to test the average total reward received per run from a given starting state using policy iteration. We tested every possible starting state ten times. Our next experiment was to compare runtime of environments of increasing size. We tested the run time of the policy iterator in the original 3x4 environment, as well as environments of size 4x4, 4x5, and 5x5. Each of these environments also contained two terminal states and one inaccessible state; therefore we tested on environments with 11, 15, 19, and 24 accessible states. Since one execution of the policy iterator was too fast to be measured on the smaller environments, we measured 10,000 executions of solve() for each environment. Each environment was tested multiple times and we used the average of those tests in the comparison.

Gridworld Experiment Results:

We were given the following data of utilities of the states in the 3x4 environment by Russell and Norvig (2003). Utility of a state was defined as the expected sum of rewards given an optimal policy starting at that state.

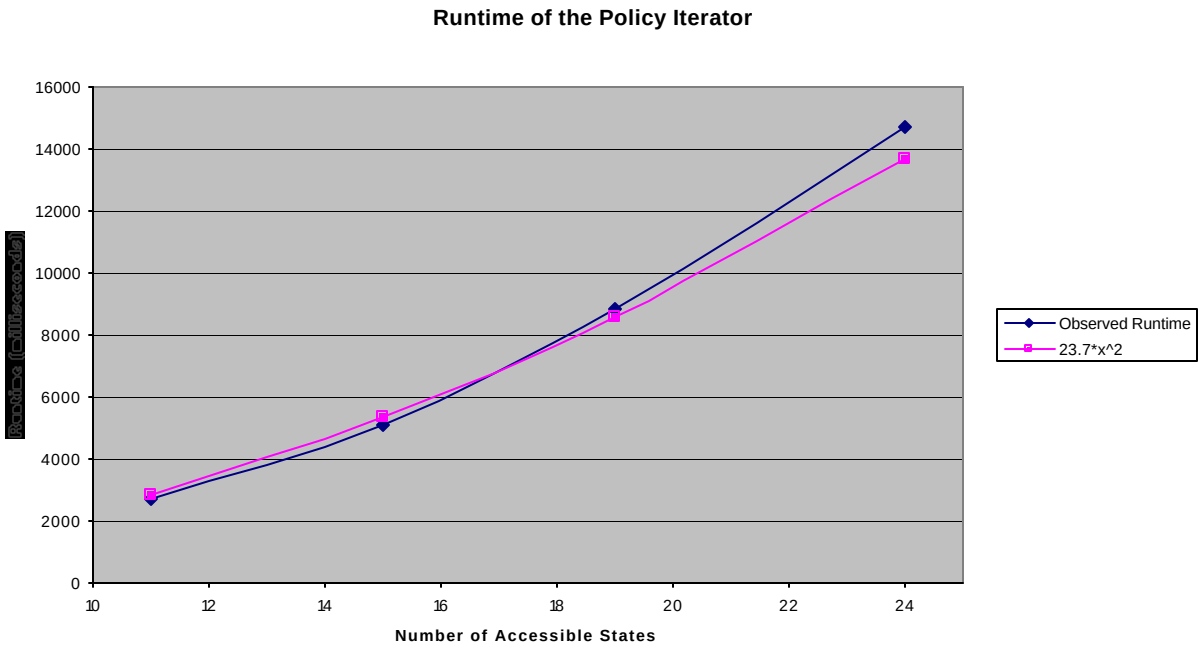
.812	*(3,1)*	.868	*(3,2)*	.918	*(3,3)*	1.0	*(3,4)*
.762	*(2,1)*	Not accessible	*(2,2)*	.660	*(2,3)*	-1.0	*(2,4)*
.705	*(1,1)*	.655	*(1,2)*	.611	*(1,3)*	.388	*(1,4)*

Our data of actual rewards was as follows:

.848	.88	.94	1.0
.776	Not accessible	.912	-1.0
.756	.456	.436	.612

The most noticeable differences came at (1,2) and (1,3). In our testing, we ended up at the terminal state with negative rewards one time each with these starting states and never with the other starting states. If we were to remove the results with negative rewards, the averages were .658 and .649 respectively – both much closer to the expected sum. Another aberration was at (1,4) where our agent performed significantly better than the expected policy. However, the rewards at (1,4) were lower than the (1,3) non-negative rewards, which would be expected as 80% of the time the agent moves from (1,3) to (1,4) and 10% of the time the agent moves from (1,3) to (2,4).

Figure 1: The observed runtime of the policy iterator in milliseconds was very similar to a function that depended on the number of accessible states squared.



Our data indicated that the policy iterator's observed runtime increased with the number of accessible states. The data resembled a function dependent on the number of accessible states squared, or a $O(N^2)$ pattern, as is visible in Figure 1.

This supports the hypothesis that runtime would increase significantly with complexity.

Quagent Procedure:

We sought to use reinforcement learning policies with a Quagent in an environment similar to the Gridworld. However, as we discovered in our previous experiment, dealing with a more complex world would result in significant increases in runtime and memory usage.

Our first project was to simulate the Gridworld experiment in the Quagent world. We had the policy generated using the policy iterator from the Gridworld experiment. We stored the optimal policy in an array, then had the Quagent adapt the movement commands of the Gridworld environment – moving the intended direction 80% of the time and an incorrect direction 20% of the time. The Quagent would access the cell in this array corresponding to its location in the Gridworld and intend to move according to the optimal policy stored in that cell.

There were a few changes that we needed to make to translate the Gridworld to the Quake world. We needed a map, which we easily designed using the Pawlicki (2003) maze generator. We also needed some way to translate coordinates in Quake to the coordinates of the Gridworld. A 64x64 pixel tile represents each Gridworld state. The coordinate system in Quake is relative in pixels to the northeast corner of the center-most tile. We used these two facts about the Quake world to generate a formula that converts a Quake coordinate to the Gridworld coordinate. Additionally, the Quagent does not move exactly the distance it is told to walk; it only is able to walk in the multiples of a given step size. We decided not to correct for this as we have in previous Quagent projects. In previous Quagent projects, precision was required. In this project, we have incorporated the lack of precision in a Quagent's walk into the Gridworld environment, as it adds an element of unpredictability.

Starting spaces in the Quagent world were set up using the quagent.config. When we wanted a random starting space, we adjusted the quagent.config file appropriately. In order to

facilitate making quagent.config files for later larger worlds, we created a file that would automatically create a quagent.config that would change the starting location of the Quagent each time a new one was created.

We then introduced function approximation and tabulation into our Quagent. Our goal here was to keep track of the results of a Quagent with a given policy and to store the information. With tabulation, the information stored would be accessed later when the Quagent needed to make a decision. With function approximation, the information stored would be used to generate a function capable of predicting utility at unvisited states. We noticed when using a large value of alpha, or learning constant, in function approximation such as .5, the data diverged from zero, so we used smaller alpha values. We chose to go further in depth with both these methods later in the experiment.

After testing out what a Quagent would do with a given policy, we expanded into the possible actions without a given policy. We tested an exploring agent that, for the first few trials, attempted a random path and stored the utility for the function approximation and in a table. The Quagent would then use this information in future trials. This method of exploration was similar to the greedy in the limit of infinite exploration (GLIE) scheme. It also advanced our work with the function approximation and tabulation, as we would now be keeping track of and using the information from past trials. We also decided to replace the Gridworld scheme of rewards with a Quagent-based reward system. The Quagent's reward was now its energy, with a 10-energy battery replacing the positive terminal state and kryptonite replacing the negative terminal state. Quagent energy started at 1000 and was reduced one unit per second when far from kryptonite and a higher reduction rate when close to kryptonite. This continued to emphasize short distances

as an optimal path, but also added a slight emphasis to maintaining a straight line, as turns would consume time and thus energy.

Next, we attempted to create our own exploration function using tabulation for direct utility estimation. We decided to base the Quagent's curiosity on how many times a state was visited and not how many times an action was taken. This reduced the number of visits we had to keep track of to 11 states from 36 actions (four possible movements out of each non-terminal state) movements. We figured that this would yield very similar results to the exploration function suggested in Russell and Norvig. Our Quagent would initially explore randomly, then record its energy at previous states once it reached the terminal state. For states visited multiple times, energy was averaged. In later trials, our Quagent selected one of unexplored adjacent states. An unexplored state was defined in this case as a state that had been explored less than two times. If all adjacent states had been explored, the Quagent would visit the state with the highest expected reward.

We chose to use a 10x10 world, with the starting state in the bottom left corner (1,1) and the terminal state in the upper right corner (10,10), to test the exploration function. For some added complexity, we added a piece of kryptonite at the middle of the right wall (5,10). The 10x10 world revealed an error that we did not encounter in the smaller Gridworld. One of the flaws from having the Quagent select the area with the highest utility is if there are two states that are each others' best successors. We decided to fix this by re-adding the implementation model. However, we soon experienced some difficulties in testing as a group of four states in a corner were each others' best successors, as well as the best successors of the five states surrounding this group. We decided it was necessary to incorporate some method of the Quagent telling that it was looping back and forth. We did this by having the Quagent keep track of its

current reward and include it as part of the expected reward. This meant if a Quagent got stuck in one of these loops, eventually the utility of the looped states would be lower than the non-looped states and the Quagent would explore elsewhere. This also had the advantage of making these looped states less desirable for future trials. We de-activated the transition model here as it was no longer needed as a temporary fix.

We also tested function approximation as a method of direct utility estimation. This would allow us to estimate utility for unvisited states with a formula derived from the visited states – a useful ability given the 100 states of the 10x10 world. Our 10x10 world was set up well to handle function approximation, since the terminal state (10,10) is on one end of the map and the farther away you are from the terminal state, the more energy it should take to reach the terminal state. Since the world is limited to x and y coordinates, we were able to determine the utility of a state at the coordinates (y,x) as $U(y,x) = \theta_0 + \theta_1 y + \theta_2 x$. The θ values were updated following each trial based on the results of past trials.

Quagent Discussion and Conclusion:

Our work with online exploration and tabulation was done using a Quagent starting with 1000 energy on a 10x10 map. We tested the final energy of the Quagent over the course of 27 trials, each building on the information of past trials; our results are displayed in Figure 2. In the first trial, the Quagent was in full exploration mode – visiting a random state each time – and had a final energy of 678.38. Over the next three trials it explored gradually more unvisited states, increasing its final energy by 38, 56, and 64 points. It settled on an indirect path to the terminal very quickly. This happened because during the first trial, the Quagent revisited the starting state and above area, then on the second trial, the Quagent exited the starting state from the right and

did not revisit the above area. The utility calculated in the area above the Quagent was then significantly higher than the area to the right of the Quagent, creating a “wall” that the Quagent

Figure 2: Final energy per trial over 27 trials using the tabulation method. Energy started at 1000 in each trial. It decreased by approximately 1 unit per second when the Quagent was far from kryptonite and more as the Quagent got closer to kryptonite. The Quagent would pick up a battery worth 10 units of energy when close to the kryptonite.

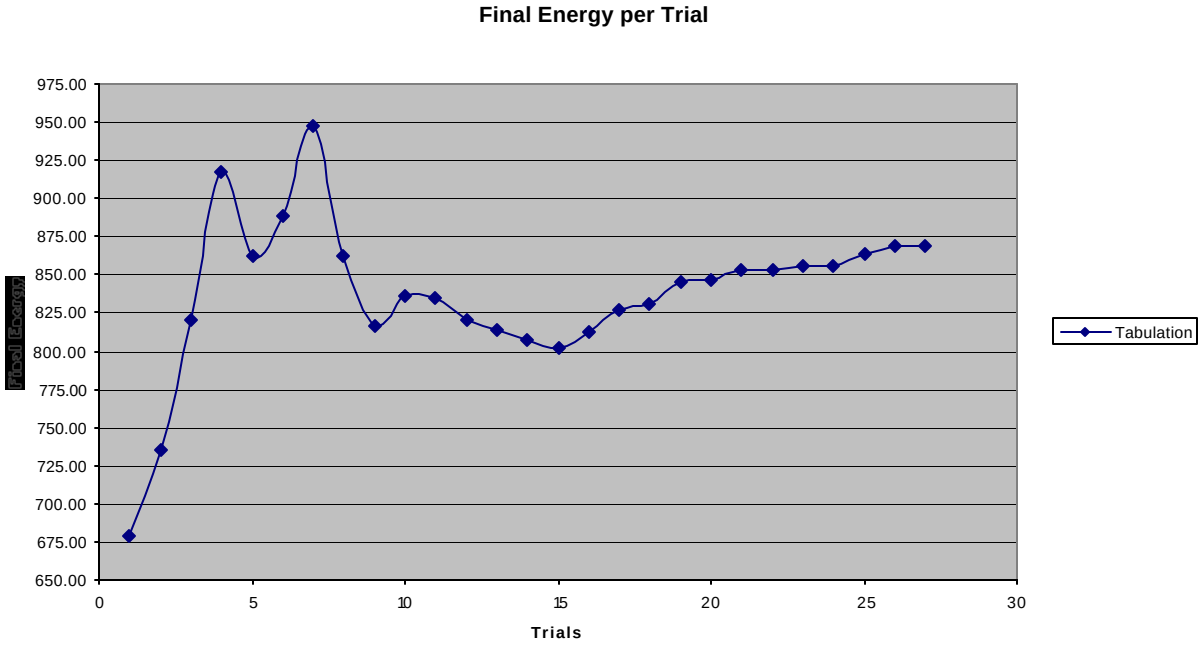
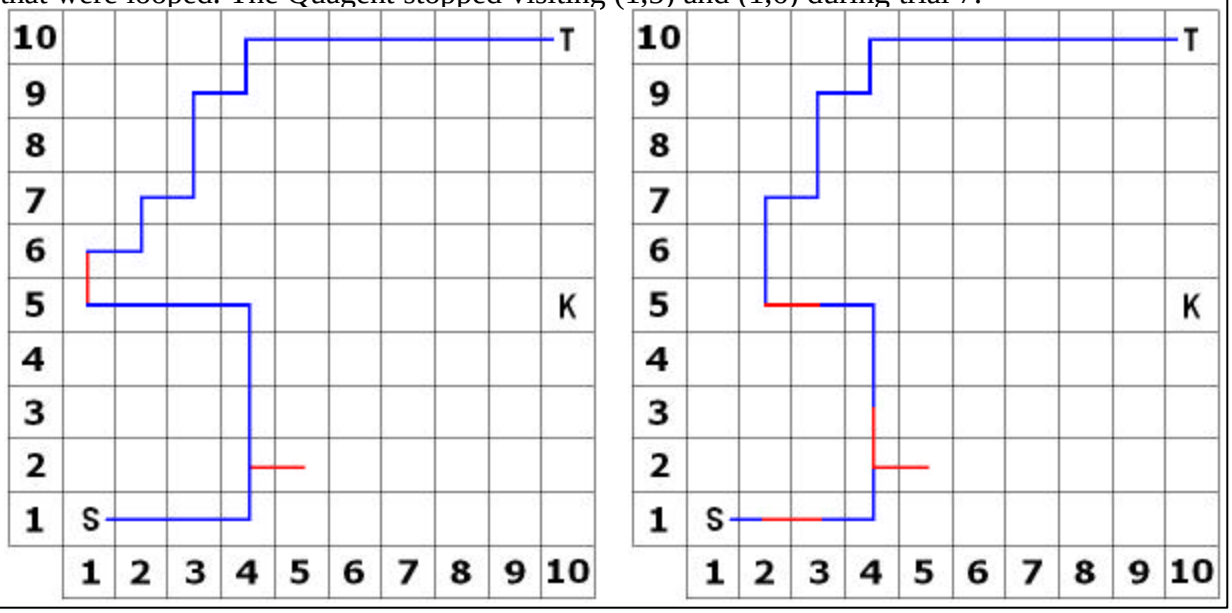
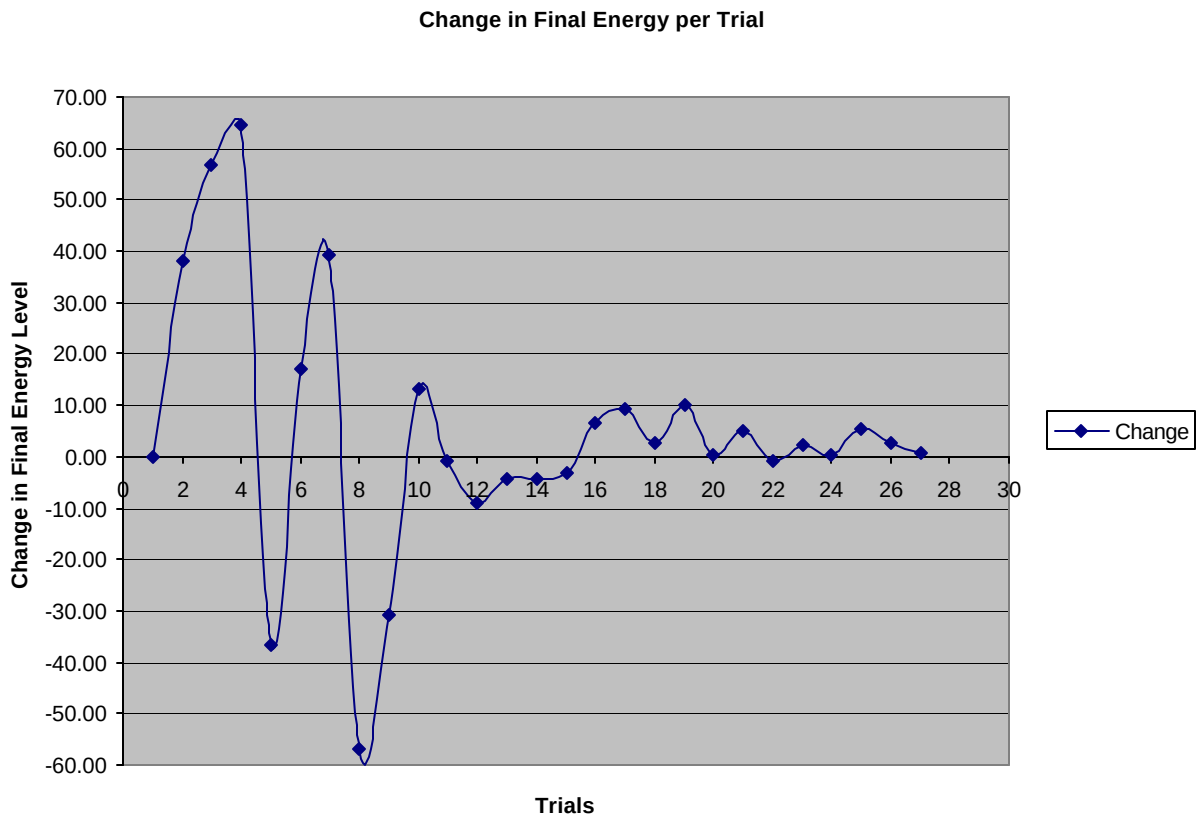


Figure 3: The 10x10 world, with starting state S, terminal state T, and kryptonite state K. The paths taken are from trials 4-6 (left) and 8-27 (right). The red areas indicate the coordinates that were looped. The Quagent stopped visiting (1,5) and (1,6) during trial 7.



would not pass over. The Quagent visited the same states – although not with the same frequency – in trials four, five, and six. It removed two of the states from its path in trial seven and stuck with this path thereafter. These paths are visible in Figure 3.

Figure 4: Changes in final energy per trial over 27 trials using the tabulation method.



The changes in final energy, displayed in Figure 4, fluctuated in the first seven trials with exploration. The changes after trial seven resulted from the Quagent finding multiple states to be each others' best successors and looping between those states. The fluctuations up and down were the results of multiple loops happening at different times; as some loops decreased in iterations, other loops increased their number of iterations. As evident from Figure 3, three loops were present at trial eight – one emerged between (1,2) and (1,3), another emerged and a small loop between (2,4) and (2,5) had been getting worse since trial four. The first loop started

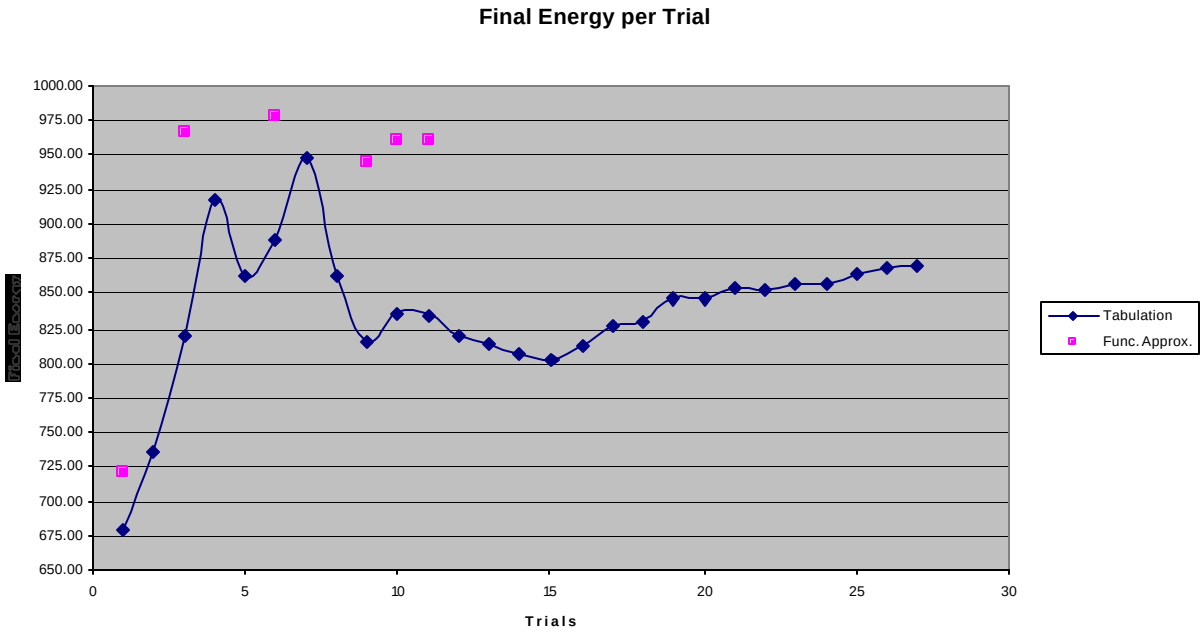
settling quickly, but the third loop continued to worsen – adding the position at (3,4) – and the second trial exploded at trial ten. This turned the positive benefits into negative benefits quickly. No new loops emerged after trial ten, and each of these loops all started gradually improving afterwards – although none were fixed by trial 27. We do expect the improvements to continue as each loop seemed to decrease iterations in the last few trials. These improvements should slow over time and approach zero.

As we can see in Figure 2, final energy had a global maximum of 947.02. Our 27th and final trial had a final energy of 869.43. Final energy increased in 10 of the last 11 trials and we expect that to continue, as mentioned before. We estimate that with the path this specific Quagent took, final energy would reach a number around 960 once all loops are eliminated. Our test with a Quagent following the optimal path – up along a wall and right along the other wall – yielded a final energy of approximately 980.

If we increased the number of times a state needed to be visited in order to be considered explored from our original number of two, we predict that the first few results from final energy would be lower, but the Quagent would emerge with a better final path. The aforementioned "wall" would be lowered and potential paths would not be ruled out as strongly as they were. However, the completely random nature of the exploration mode could lead the Quagent to take less than optimal paths, as we saw in the experiment.

Function approximation yielded some unexpected results. The Quagent reached the terminal state in its first trial when we fed it starting theta values of –40, 2, 2. On its next trial, it got stuck in one of the non-terminal corners in another state. This is because the Quagent would reach the terminal state when its last two theta values were both positive – as x and y increased, so did the expected reward. When the Quagent had a positive θ_1 and negative θ_2, the Quagent

Figure 5: Final energy for function approximation compared to tabulation. While the Quagent using tabulation was able to reach the terminal state every time, the Quagent using function approximation only reached the terminal state during trials one, three, six, nine, ten, and eleven. Utility values were higher when the Quagent with function approximation did reach the terminal state, however, and approached the optimal final energy of 980 with a maximum of 978 energy.

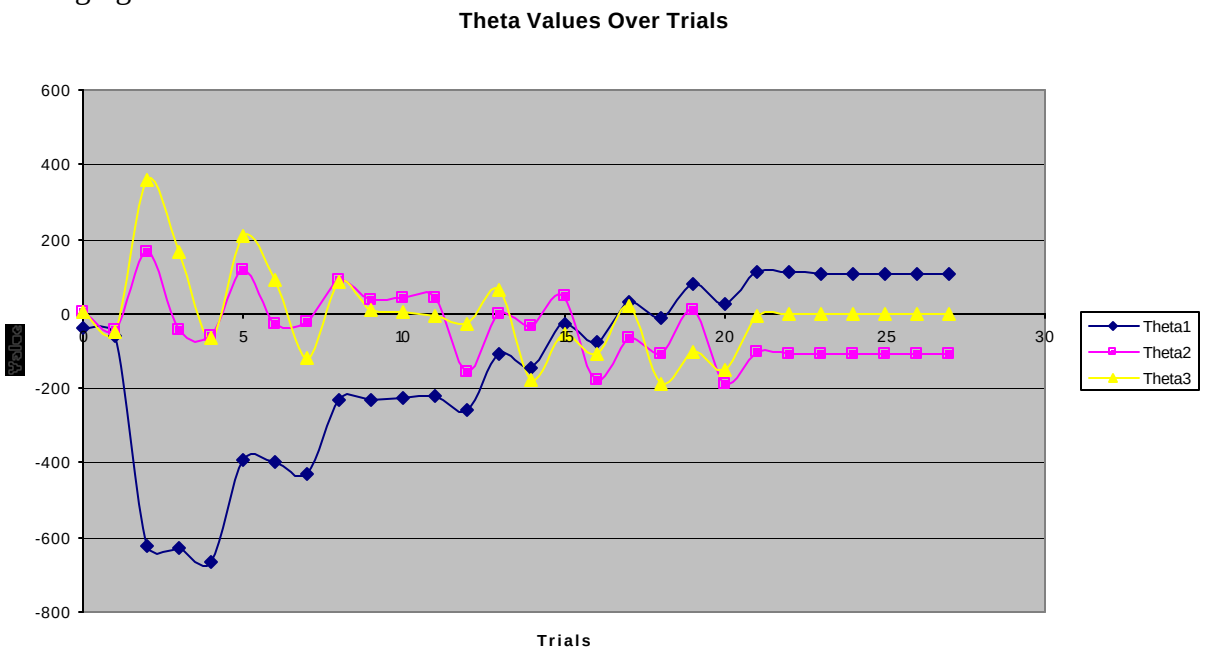


would move to the position (1, 10) as utility would increase with increased x and also increase with decreased y. Whenever the Quagent got stuck in a non-terminal corner state, it would lose energy at a consistent rate when walking between two corner states. Instead of leaving the Quagent to run until it ran out of energy, we took these rates and simply simulated the results using the rate information. We hoped that getting stuck in early trials would discourage going to that trial again in the future.

The Quagent using formula approximation reached the terminal state in the third and sixth trials once each and then appeared to have established a pattern by reaching the terminal states in trials nine through eleven. However, that was the last time both values were positive. The two latter theta values started converging to negative numbers – the final trial showed theta2 and theta3 values of –109.3 and -.1, leading the Quagent to get stuck in the starting state instead

of the terminal state. This was confusing at first, but after analyzing the numbers more closely, this appeared to be an issue with the environment. The effect of walking towards the kryptonite apparently outweighed the benefits of picking up the battery at the terminal state. We expect if that this were not the case, the Quagent would reach the terminal state after a number of trials with higher utility than the tabulation method, particularly in such a straightforward and simple environment as this. The final theta value – the effect of x on expected rewards – started nearing 0 by halving each time. It is possible but unlikely that in another few trials, the Quagent could explore a different corner instead, affecting the assumption that moving towards the kryptonite outweighed the benefits of picking up the battery.

Figure 6: The theta values used in the function $U(y,x) = \theta_0 + \theta_1 y + \theta_2 x$ were recalculated after every trial. After 27 trials, the formula used was 107.76 – 109.3y -.1x. The data started converging after trial 20.



We made the Quagent world deterministic to test the exploration function on the large scale of the 10x10 world. A stochastic world, such as one with the Gridworld transition model, would add additional fluctuations to the results of the exploration. However, we believe the

Quagent will still eventually converge to the same results as it would in a deterministic environment. We also believe the Quagent may be effective in implementing these techniques in other more complex environments, such as a 10x10 world with more obstacles, a different terminal state, multiple terminal states, etc. However, the technique of function approximation might not work as well outside of a planar map, like the one we used here. Overall, the techniques we have developed here should be applicable to many different environments, with varying effectiveness.

Responsibilities:

Corey was the lead programmer, primarily working with the Java code. Alexander developed the quagent.config generator and was present for most of the Java coding. Alexander was the lead author of the writeup and README.

References:

Hu, B. *CSC 242 Learning Project Source Code*. http://www.cs.rochester.edu/~brown/242/assts/bh_learning/learning.html. 2004.

Russell, S. and Norvig, P. *Artificial Intelligence: A Modern Approach (2nd Ed.)*. New Jersey: Prentice Hall, 2003.

Various. *Quagents: Quake Agents.* http://www.cs.rochester.edu/research/quagents/. 2006.

Appendix: Data Results from Individual Experiments

Trial	Utility Using Tabulation	Trial	Utility Using Function Approximation	Theta1 Using Function Approximation	Theta2 Using Function Approximation	Theta3 Using Function Approximation
			Initial values of theta →	-40	2	2
1	678.38	1	721.76	-57.92	-44.94	-49.14
2	735.57	2	(never finished) 0.00	-623.25	163.64	358.86
3	820.44	3	967.11	-630.74	-44.04	167.90
4	917.23	4	0.00	-664.39	-57.15	-63.84
5	862.06	5	0.00	-393.54	115.13	206.01
6	887.93	6	977.91	-396.77	-27.39	89.07
7	947.02	7	0.00	-429.05	-24.28	-117.29
8	861.94	8	0.00	-229.32	90.43	82.44
9	815.85	9	944.41	-231.15	36.92	12.19
10	835.54	10	961.31	-226.76	41.68	3.69
11	834.14	11	960.51	-222.67	42.56	-5.63
12	820.52	12	0.00	-257.51	-153.68	-28.27
13	813.85	13	0.00	-105.55	-1.72	65.76
14	807.25	14	0.00	-147.30	-31.89	-176.85
15	802.65	15	0.00	-25.92	49.65	-55.48
16	812.55	16	0.00	-77.51	-175.56	-107.06
17	826.54	17	0.00	31.77	-66.28	20.44
18	830.64	18	0.00	-9.71	-107.76	-190.60
19	845.74	19	0.00	80.77	11.00	-100.12
20	846.04	20	0.00	28.26	-188.80	-152.63
21	853.74	21	0.00	112.47	-104.59	-5.97
22	852.54	22	0.00	110.10	-106.96	-3.01
23	855.74	23	0.00	108.90	-108.16	-1.52
24	856.04	24	0.00	108.30	-108.75	-0.77
25	864.14	25	0.00	107.99	-109.07	-0.39
26	868.33	26	0.00	107.84	-109.22	-0.20
27	869.43	27	0.00	107.76	-109.30	-0.10