*Christoph F. Eick*

COSC 6368 (Fall 2017)

Project 2: Learning Paths from Feedback Using Q-Learning and SARSA

Individual Project

Fourth Almost Final Draft



Deadline: We., November 22, 11p (or Su., Nov. 19, 11p—you get a 5% bonus)

Last updated: October 27, 10:30a

In this project we will use reinforcement to learn and adapt “promising paths” in robot-style grid world. Learning objectives of Project2 include:

* Understanding basic reinforcement learning concepts such as utilities, policies, learning rates, discount rates and their interactions.
* Obtain experience in designing agent-based systems that explore and learn in initially unknown environments.
* Learning how to conduct experiments that evaluate the performance of reinforcement learning systems and learning to interpret such results.
* Development of visualization techniques summarizing how the agent moves, how the world and the q-table changes, and the system performance
* Development of path visualization and analysis techniques to interpret and evaluate the behavior of agent-based path-learning systems.



Figure 1: Visualization of the PD-World

In particular in Project2 you will use *q-learning/SARSA[[1]](#footnote-1) for the PD-Word* (<http://www2.cs.uh.edu/~ceick/ai/2017-World.pptx>), conducting 3 experiments using different parameters and policies, and summarize and interpret the experimental results. However, before running the experiment, you will develop path visualization techniques that are capable to shed light on what paths the learning system actually has learnt from obtained Q-Tables—we call such paths *attractive paths* in the remainder of this document.

In the three experiments, you conduct, the discount rate is assumed to be γ=0.5 and we assume that q values are initialized with 0 at the beginning of the experiment. The following 3 policies will be used in the experiments:

* **PRANDOM**: If pickup and dropoff is applicable, choose this operator; otherwise, choose an applicable operator randomly.
* **PEPLOIT**: If pickup and dropoff is applicable, choose this operator; otherwise, apply the applicable operator with the highest q-value (break ties by rolling a dice for operators with the same q-value) with probability 0.85 and choose a different applicable operator randomly with probability 0.15.
* **PGREEDY**: If pickup and dropoff is applicable, choose this operator; otherwise, apply the applicable operator with the highest q-value (break ties by rolling a dice for operators with the same q-value).

The experiments are as follows:

1. In Experiment 1 you use α=0.3, and run the Q-learning algorithm[[2]](#footnote-2) for 3000 steps with policy PRANDOM; then run PGreedy for 3000 steps[[3]](#footnote-3). Display and interpret the Q-Table you obtained in the middle and the end of the experiment.
2. In Experiment 2 you use α=0.3, and run the Q-learning algorithm for 6000 steps with policy PEXPLOIT—however, use policy PRANDOM for the first 200 steps of the experiment, and then switch to PEXPLOIT for the remainder of the experiment. Analyze the performance variables and summarize what was learnt by analyzing the q-table at different stages of the experiment.
3. In Experiment 3 you use α=0.3, and run the SARSA q-learning variation for 6000 steps with policy PEXPLOIT—however, use policy PRANDOM for the first 200 steps of the experiment, and then switch to PEXPLOIT for the remainder of the experiment. When analyzing Experiment 3 center on comparing the performance of Q-learning and SARSA. Also report the final q-table of this experiment.

For all experiment, if the agent reaches a terminal state, restart the experiment by resetting the PD world to the initial state, but do not reset the Q-table. Run each experiment twice, and report[[4]](#footnote-4) and interpret the results; e.g. utilities computed, rewards obtained in various stages of each experiment.

Assess which experiment obtained the best results[[5]](#footnote-5). Next, analyze the various q-tables you created and try identify attractive paths[[6]](#footnote-6) in the obtained q-tables, if there are any. Moreover, briefly assess if your system gets better after it solved a few PD-world problems—reached the terminal state at least once. Finally, briefly analyze to which extend the results of the two different runs agree and disagree in the 3 experiments.

Moreover,

* Students that provide good methods for visualizing attractive paths obtain extra credit.
* Make sure that you use different random generator seeds in different runs of the same experiment to obtain different results—having identical results in the 2 runs of the same experiment is unacceptable. It is okay just to report and interpret the Q-tables for the better of the two runs for each experiment, but you should report the performance variables for all six runs.
* Use the q-learning state space recommended in the PhD World slide show[[7]](#footnote-7)!
* You should use the traditional Q-learning and SARSA algorithm in the project and not any other Q-learning or reinforcement learning algorithm!
* As far as counting operator applications in performance measures is concerned, you should never count operators that are not applicable in a particular state.

Write a 5-9 page report that summarizes the findings of the project. Be aware of the fact that at least 25% of the points available for this project are allocated to the interpretation and comparison of the experimental results. Finally, submit the source code of the software you wrote in addition to your project report and be ready to demo the system you developed. Detailed instructions on what to submit should be available approx. Nov. 10.

**Academic Honesty:**

This project is an individual project. Collaborating with other students in COSC 6368 and using code your class mates wrote is strictly disallowed. Reporting false results or solutions that were not obtained by running your own programs is also a serious academic honesty violation.

Project2 Links:

<http://www-all.cs.umass.edu/rlr/domains.html>

<http://courses.cs.washington.edu/courses/cse473/15sp/assignments/project3/project3.html>

<http://ai.berkeley.edu/project_overview.html>

<https://github.com/kristofvanmoffaert/Gridworld>

<http://cs.stanford.edu/people/karpathy/reinforcejs/gridworld_td.html>

<https://mediatum.ub.tum.de/doc/1238753/1238753.pdf>

<http://archive2.cra.org/Activities/craw_archive/dmp/awards/2004/Coggan/FinalReport.pdf>

<http://www2.econ.iastate.edu/tesfatsi/RLUsersGuide.ICAC2005.pdf>

<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.113.7978&rep=rep1&type=pdf>

1. SARSA is a variation of Q-learning that uses the q-value of the actually chosen action and not the q-value of the best action! [↑](#footnote-ref-1)
2. You have the option to use the SARSA algorithm instead, if you prefer that! Make clear in your report which algorithm you used in Experiment1. [↑](#footnote-ref-2)
3. Do not reset the q-table after 3000 steps! [↑](#footnote-ref-3)
4. Additionally, report the following Q-tables for Experiments 2 (or Experiment 3 if you prefer that, in this case you will only need to report the final Q-Table of Experiment 2) in your report a) when the first drop-off location is filled (the eighth block has been delivered to it) and b) when a terminal state is reached and c) the final Q-table of each experiment.

The Q-table in the screenshot should be presented as a matrix, with *s* rows (states) and *t* columns (operators).  Thus, the Q-table for recommended state space has 25 x 2 rows and 6 columns; however, the q-values for the drop-off and pickup operators do not need to be reported. [↑](#footnote-ref-4)
5. Provide graphs that show, how the algorithm’s performance variables changed over the duration of the experiment in the three experiments. [↑](#footnote-ref-5)
6. A path going from (i,j) to (i’,j’) is attractive if, the q-values of the motion operators are high in comparison to other directions. Be aware of the fact tha different paths are attractive for agents who hold a block and for agents how do not hold a block. [↑](#footnote-ref-6)
7. See PD-World slide show. [↑](#footnote-ref-7)