A8 – CS4300 Assignment A8 Lab Report

Policy Iteration

By Haochen Zhang & Tim Wei
11/28/2017

1. Introduction

In this assignment, we were to find out the behaviours of a policy iteration algorithm on a fully observable Wumpus World. We implemented such a function, *CS4300_MDP_policy_iteration* to study about the policy evaluated by it. In this report, we want to answer the following point of interests:

- What are the policies calculated under different reward values?

2. Method

We have implemented 2 main functions in this lab which is *CS4300_MDP_policy_iteration* and *CS4300_policy_evaluation*, and used *transition_probability_table* from A7 for testing. The agent is to learn a policy for the following Wumpus world:

0 0 0 G
0 0 P 0
0 0 W 0
0 0 P 0

CS4300_MDP_policy_iteration is the implemented policy iteration algorithm from p. 657 of the book. We made a modification where when the policies from the previous iteration are exactly the same with the current iteration, the function also returns.

*CS4300_policy_evaluation*is is the following equation from p. 657 of the book implemented:

$$U_{i+1}(s) \leftarrow R(s) + \gamma \sum_{s'} P(s'|s,\pi(s))U_i(s')$$

We ran *CS4300_MDP_policy_iteration* with the reward value of a clear cell ranging from -2000 to 2000 to find how many different policies will be generated.

3. Verification of Program

For *CS4300_MDP_value_iteration*, as you can see in the following table with max iteration to be 1000 and eta to be 0.1, the closer the cells is to the gold, the higher the utility it will have. On contrary, the closer the cell is to the pits or Wumpus, the lower the utility it will have. Interesting part is that the starting cell has negative utility, which means that agent want to go the the terminal state as soon as possible and refuse to turn back even if it ends up at cell (2,1) after taking action forward.

0.4548	0.5346	0.6292	1.0000
0.2260	■	0.4127	-1.0000
-0.2009	0.1511	0.2522	0.0521

4. Data and Analysis

With the reward value of a clear cell from -2000 to 2000, *CS4300_MDP_policy* will generates about 300 different policy sets. Due to the amount of the policy sets we will not show them in the report. The human readable form of the policy sets is *policies_s* generated by *CS4300_policies_with_different_rewards*.

5. Interpretation

In this section, we will answer the question we posted for this study:

- What are the policies calculated under different reward values?

Each time *CS4300_policies_with_different_rewards* runs, there are different numbers of unique policy sets, and it is always around 300. We will try to categorize most of them:

a. R(s) < -1622

The world is so painful that the policy is to get to a terminal state as soon as possible, even if it means death.

```
">"   ">"   ">"   "G"
">"   ">"   "x"   "^"
">"   ">"   "x"   "<"
">"   ">"   "x"   "<"
```

b. -1 < R(s) < 9

This set of policy is probably ideal for an agent to work with. The world is bearable so it does not take any risk, and will try to get to the gold.

```
">"   ">"   "^"   "G"
"^"   "<"   "x"   ">"
"^"   "<"   "x"   ">"
"^"   "<"   "x"   ">"
```

c. 9 < R(s)

Once the rewards are larger than 9, the policies became uncertain. Agent will still try to avoid terminal state but action taken in cell (2,1), cell (2,2) and cell (2,3). Action in cell (3,3) will taken from Up and Right randomly and action taken in cell (4,3) will be Up and Right randomly. The behavior in cell (4,1) and cell (4,2) are still the same which is always Right.

```
"+"    "+"    "+"    "G"
"+"    "+"    "x"    "+"
"+"    "+"    "x"    ">"
"+"    "+"    "x"    ">"
```

6. Critique

This assignment was largely based on the work from A7, thus making the progress incredibly faster than the previous assignments. A problem we encountered, although was solved relatively quickly, is that the matrices are transposed. Having a set direction of matrices can prevent this from the future.

7. Log

Haochen Zhang(Section 1, 3, 5)
A total of 2 hours was spent performing the experiment in Matlab.
A total of 2 hours was spent performing the experiment in writing the report.

Tim Wei (Section 2, 4, 6)
A total of 2 hours was spent performing the experiment in Matlab.
A total of 2 hours was spent performing the experiment in writing the report.