A5 – CS4300 Lab Report – 11/9/17

An Agent Using Monte Carlo Probability Prediction to Navigate Wumpus World

1. Introduction

Each iteration of the agents of AI past seemed to be stricken by ails of one kind or another. My random agent was easy to code but had no idea where he was going or even why he was going there. My RTP agent was cold and calculating, but often found himself lost in thought about where pits could be for far too long. This time, I aimed to see if using Monte Carlo probability prediction could both keep him up and on his feet while simultaneously keeping those feet out of pits and the mouths of Wumpuses. Monte Carlo methods use mass simulation to predict the possibilities of one single action. In my case, CS4300_MC_agent was to randomly generate a large number of boards in order to find boards which would match its own known board (i.e., the known percept values of its board). Once it had done that, it would sample the actual hazards of the generated boards and come up with actual probabilities of those hazards existing in various locations in its own board. The known weaknesses of this method for our problem are:

1. There are many cases where the probability of pits in two neighboring squares are actually theoretically equal. In cases like these, the only option the agent has is to take a 50/50 risk.
2. As the board becomes more explored, it becomes exponentially less likely that a randomly generated board will match the known board's percept values. Thus, it was take exponentially more time to generate such a board as the agent progresses.

A few relevant questions I aim to answer in this report are:

1. Does increasing the number of matched boards in the Monte Carlo estimate function increase the potential to beat a board
2. Does putting a limiting factor in the Monte Carlo estimate function to reduce time spent decrease the potential to beat a board?

2. Method

Our agent used the hybrid agent model, that is, it produced plans of action based on priority in the following order (each successive plan is only formed if the previous one was not formed):

1. If gold is found, pick it up and return home.
2. Use A* pathfinding to explore an unexplored known safe location.
3. Try to kill the Wumpus if a stench has been smelled and you still have an arrow.
4. Use A* pathfinding to explore the least unsafe location.

Our agent used the Monte Carlo method to attempt to estimate where the Wumpus might be as well as which locations were more and less likely to have pits. Because Monte Carlo's precision can potentially affect the accuracy of its predictions, I made the number of correctly generated boards a variable with which to test against. As mentioned earlier as well, a large number of known percepts will make

randomly generated boards much less likely to match the known board. Thus, I also put a limit on the number of total boards to be generated and decided to test how this affected performance as well. Note that in some cases, a matching board could not be generated within the limit of boards to generate. In cases like these, I forced the estimate function to at least generate one matching board before it was allowed to quit.

3. Verification

For verification of my program, I provide various example boards with hand-worked theoretical probabilities of pits and Wumpuses existing, as well as the estimates calculated by my Monte Carlo function using 50 trials:

B = Breeze • S = Stench • 0 = Clear • U = Unknown • ! = Breeze & Stench

Board 1: Percepts:

U	U	U	U
U	U	U	U
U	U	U	U
B	U	U	U

P(Pits) Theoretical

0.2	0.2	0.2	0.2
0.2	0.2	0.2	0.2
0.55	0.2	0.2	0.2
0	0.55	0.2	0.2

P(Pits) Monte Carlo

0.2	0.18	0.24	0.28
0.18	0.24	0.16	0.22
0.62	0.22	0.2	0.14
0	0.52	0.22	0.14

P(Wumpus) Theoretical

0.08	0.08	0.08	0.08
0.08	0.08	0.08	0.08
0	0.08	0.08	0.08
0	0	0.08	0.08

P(Wumpus) Monte Carlo

0.06	0.08	0.04	0.06
0.06	0.06	0.08	0.08
0	0.06	0.06	0.1
0	0	0.08	0.1

Board 2: Percepts:

U	U	U	U
U	U	U	U
U	U	U	U
S	U	U	U

P(Pits) Theoretical

0.2	0.2	0.2	0.2
0.2	0.2	0.2	0.2
0	0.2	0.2	0.2
0	0	0.2	0.2

P(Pits) Monte Carlo

0.12	0.14	0.32	0.22
0.24	0.26	0.28	0.18
0	0.24	0.12	0.18
0	0	0.2	0.16

P(Wumpus) Theoretical

0	0	0	0
0	0	0	0
0.5	0	0	0
0	0.5	0	0

P(Wumpus) Monte Carlo

0	0	0	0
0	0	0	0
0.5	0	0	0
0	0.44	0	0

5. Data

Statistical Data on the Scores of CS4300_MC_agent for 250 Different Boards Using Variable Monte Carlo Trials

	Ad Hoc Agent	50 trials	100 trials	200 trials
Mean	-936.196	381.116	352.776	321.516
Variance	678624.343	808748.9	816817.1	848930.5
Standard Dev	823.7866	899.3047	903.7794	921.3743
Margin of Error	102.1176	111.479	112.0337	114.2147
95% Confidence Interval	-936.196 ± 102.1176	381.116 ± 111.479	352.776 ± 112.0337	321.516 ± 114.2147

* Note that each trial was limited to a maximum 20,000 boards generated for each Monte Carlo function call

Statistical Data on the Scores of CS4300_MC_agent for 250 Different Boards Using Variable Monte Carlo Limits

	Limit 10000	Limit 20000	Limit 30000
Mean	289.448	381.116	329.496
Variance	875484.7	808748.9	832886.1
Standard Deviation	935.6734	899.3047	912.626
Margin of Error	115.9873	111.479	113.1303
95% Confidence Interval	289.448 ± 115.9873	381.116 ± 111.479	329.496 ± 113.1303

* Note that each trial performed up to 50 trials for each Monte Carlo function call.

6. Interpretation

What this data seems to indicate is that keeping the Monte Carlo estimate function restricted to 50 trials in our problem is the optimal way to go about solving boards. Due to the large margin of error on the data, it's impossible to conclusively say that this will lead to the best scores, but I know as the programmer that it certainly ran much faster and yielded similar results. In any case, the Monte Carlo agent was much better than the Hybrid Ad Hoc agent that I constructed for A4. While it may have taken significantly longer, the mean indicates that the Monte Carlo agent retrieved the gold over 50% of the time it ran for each variable number of trials. The data on variable limits seems to suggest conclusively that 10,000 is not a high enough limit to achieve optimal results, but that going beyond 20,000 will not continue to yield better results. A cutoff point was definitely expected in this area, but this data doesn't accurately show what that cutoff point was, only its existence.

7. Critique

My biggest critique of this experiment is, like usual, the amount of time it took to run each experiment. Had it not taken upwards of an hour to run the agent on the 250 boards for some parameters, it would have been more feasible to collect more precise data. In terms of my own experimental data, I think it would be helpful in the future to determine what the cutoff point for limit in our problem is. The three data points that I collected are too far apart to be of much use in determining the precise location of this point. One additional experimental variable that might have yielded additional valuable information would be to measure the effect of board exploration on time taken to run the Monte Carlo estimate function for various boards.

8. Log

- 1 hour to write the Monte Carlo estimates function
- 3 hours to write and do initial bug fixes of CS4300_MC_agent
- 5 hours to tweak CS4300_MC_agent to keep it from crashing in corner cases
- 6 hours spent across several different sittings to collect data
- 2 hours to write up this lab report