CS520 Homework 2 - Partial Sensing

Arun Subbaiah • as3590

Agent 1 - Blindfolded. The agent can only learn about an obstacle by bumping into it.
Agent 2 - Can ‘see’ in 4 primary directions.
Agent 3 - Example inference agent. Can ‘sense’ the number of blocks in the 8 neighboring directions

Agent 3 Implementation:

When the agent moves into a new cell and finds it to be blocked or unblocked, it’s updated on the agent’s memory grid (initially the agent’s memory will be empty). Whenever any cell in the agent’s memory is updated to *True* (unblocked) or *False* (unblocked), the *updateSensing()* method is called on all its 8 neighbors which updates the sensing variables (c(x), b(x), e(x) and h(x)). This is because its neighbors are the only ones whose sensing variables might change when a cell is found to be blocked or unblocked. Then updateInference() (Inference engine) method is called on all these neighbors to try to solve for any possible solutions. If we are able to get a confirmed *True* or *False* value on any cell through the inference, then updateSensing() is called on all this cell’s neighbors. And the loop continues until there’s nothing that can be solved.

Plot Generation:

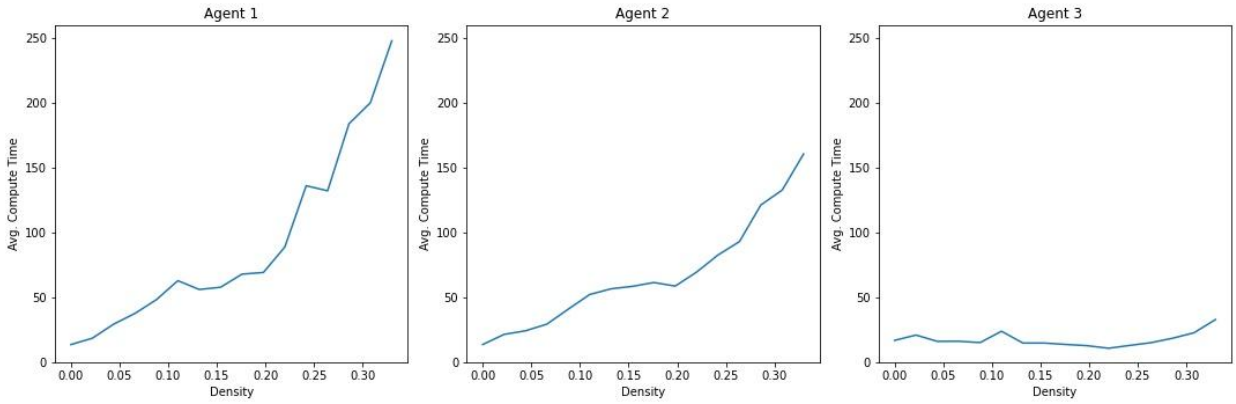
At specific values of density, two plots are generated for each agent.

1) Avg. computational time vs Density
2) Avg. path length vs Density

These plots are generated using the average value of 20 trials that are run with different gridworlds at each density.

Performance Comparison:

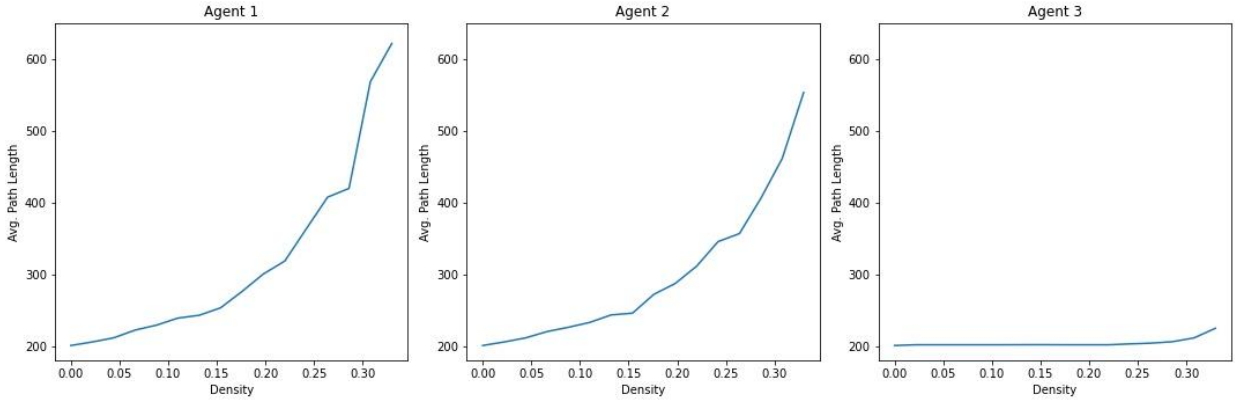
Average computational time vs Density:



The average computational time for these 3 agents varies greatly, agent 3 being the best performing. At the highest density of 0.33, the computational time for agents 1,2, and 3 are 247, 160, and 33 seconds respectively. Agent 3 is 8 times faster than agent 1 and 5 times faster than agent 2. Hence, we can conclude that an agent which can sense the number of blocks and infer is far superior to an agent that can see 4 sides.

An interesting thing to note, is agent 3's computational time stays almost constant with very few fluctuations through increasing density while the computational time of the other 2 agents increases with density.

Average path-length vs Density:



Agent 3 performs significantly better in terms of path length as well. At the highest density of 0.33, the computational time for agents 1,2, and 3 are 621, 553, and 225 cells respectively.

Agent 3's path length is very impressive considering the fact that the shortest path length in an empty grid is 199. But agent 3 is able to solve the grid with density = 0.33 at a path length of 225. Similar to the computational time graph, the agent 3's performance almost stays the same with increasing density with a very slight increase.