CS:440 Intro to Artificial Intelligence

Assignment 2
Trivial Pursuit

Project Report

Group Members:
Ruiqi Wang: (rw555@scartletmail.rutgers.edu) RUID: 192001525
Vivek Rajyaguru: (vpr11@scarletmail.rutgers.edu) RUID: 195003544
Divay Tomar: (dt578@scarletmail.rutgers.edu) RUID: 195004196

TA: Dhruv Metha Ramesh: dm1487@scarletmail.rutgers.edu

Part 1 - Implementation

A) For the most efficient implementation of this environment, we used a combination of linked lists and arrays. First, we created a linked list of 40 nodes, then we put each node according to their data into the according array slots. The node has an array of size of 3 containing up to 3 edges and the number representing itself allows us to access every node with time of O(1). To make sure the extra edges don't exist between the two nodes, we simply created a 2D array to mark pre-existing numbers. For example, if we have Node 10 connected to Node 20, we will make the 2D array as list[10][20] = true and list[20][10] = true. This represents that we already have these 2 edges created, so the program will skip over the combination of these 2 numbers and generate another combination of 2 numbers to add a random connecting edge.

B) As Agent 0 does not have any steps involved in it, Agent 0 will have probability 1 of catching the target as long as there is no limit on the number of steps required. This is because the Agent is stagnant and does not move from its spot. The only way to catch the target is for the target to appear on the same node as the agent. In doing so, the target randomly navigates around the map, jumping from node to node. In a certain amount of time or steps, the target will eventually end up on the same node as Agent 0. This is true for all cases where no upper limit is set on the number of steps because the only dependent variable in this situation is how the target moves in each time step.

C) The results of Agent 3 do not surprise us because they are almost the same average as Agent 0. Their behavior is nearly identical, in the fact that they sit at or examine the same node, all while the target is moving randomly around the map. They both wait for the target to reach the node that they are waiting at, and then catch the target. The results are solely up to the behavior of the target, so the average results are not that surprising.

D) Yes, Agent 6 will eventually capture the target. Although the Agent never knows where the target is or how it is moving around the map, it uses an approach similar to that of Agent 4, in which it utilizes a belief system. For agent6 we used the Kamal filtering to better update the belief state, which helped in accurately tracking the target's position and subsequently identifying the node with the highest probability of containing the target. Furthermore, given enough steps, Agent 6 eventually captures the target since it can examine any node on the map at any given time.

E) In situations where the agent does not seem to be catching the target, we should take a step back and analyze step-by-step where the agent is moving, where the target is moving, and why the agent is not able to find the target. We can also ensure that the agent's belief state is accurately updated based on the observation and the filtering techniques. The better the belief state, the better the agent can predict the target's movements and plan its moves accordingly.

Part 2 - Analysis

Design:

For Agent 2, we followed a similar approach to Agent 1. Both agents found the shortest path to the target and moved one spot in that direction. However, Agent 1 was already incredibly fast at finding the target. Averaging only 5 steps on 100,000 trials was hard to beat. We used a probabilistic style for Agent 2, to speed up the process and have the Agent move quickest to the node it believes the target will move to, rather than moving straight to the node that the target is in. In doing so, the agent and target will both move into the same node, rather than the agent moving into the node with the target, saving, on average, 1 step across 100,000 trials.

To implement Agent 5, we first needed to do a better distribution of probabilities. It had to be better than Agent 4, as Agent 4 just brutally distributes the probabilities equally if it does not find the node. Agent 5 realizes that if a node does not have the target, its neighbor will have less probability of being the node that the target will move into, so it's simply an improvement of probability distribution than Agent 4. Agent 5 utilizes the filtering method, which is a more sophisticated approach to probabilistic distribution, taking into account the neighboring nodes' information and previous beliefs. This enhancement allows Agent 5 to make more informed decisions and significantly improves its ability to estimate the target's position.

In Agent 7, our goal was to enhance Agent 6. As Agent 6 examines nodes with the highest probability of containing the target, updating its belief state, and then moving along the shortest path, we modeled Agent 7 around that. However, the improvement that we made to Agent 7 is that every time the agent moves, it updates the probabilities of each node and belief state, singles out the nodes that are neighboring the examined node, and minimizes their probabilities if the target is not found, and then moves along the shortest path to reduce the distance. In doing so, Agent 7 quickly and accurately finds that target relative to Agent 6.

Part 2 - Analysis (continued)

Data:

In order to get an accurate reading and interpretation of how well our agents are performing, we ran 100,000 trials for each agent against the target in individually different, random environments, and found the average steps that it takes for each agent to reach the target. We chose to run 100,000 trials to minimize the effect that outliers will have on the data. In the table below, we have listed the average steps for each agent:

Agent Name	Average Steps to Find Target (100,000 Trials)
Agent 0	58 steps
Agent 1	5 steps
Agent 2	4 steps
Agent 3	59 steps
Agent 4	38 steps
Agent 5	35 steps
Agent 6	28 steps
Agent 7	21 steps

Part 2 - Analysis (continued)

Results:

The results that we got do make sense and agree with our intuition. We can see that Agent 0 and Agent 3 are ideally the same in nature, so we would expect the average number of steps to be relatively identical. Agent 1 was incredibly fast, and we knew this would happen since the agent can search the grid for the shortest route to the target, and then move along it. Similarly, we expected Agent 2 to be slightly faster than the already-fast Agent 1, since Agent 2 also takes into account the probability of the nodes the target could move into next, then makes an enlightened decision for the next step based on the shortest route. Agent 4 and 5 worked exactly how we had imagined them to be. They should be slower than Agents 1 and 2 because they do not know where the target is, but can only make an assumption based on constantly updating their belief state. Agents 6 and 7 also performed as we expected them to. They were a combination of the methods used in Agents 1 and 2 and Agents 4 and 5, so we expected them to be faster than Agents 4 and 5, but slower than Agents 1 and 2. Since these agents used a belief system but also navigated on the path that reduced the distance between them and the believed state of the target, they were significantly faster than Agents 4 and 5.

All Agents 2, 5, and 7 stacked up well against the other agents, specifically their counterpart agents. The reason why these agents worked so well is that we improved Agents 2, 5, and 7 based on the performance of their counterparts. For Agent 2, we saw that Agent 1 was always finding the current position of the target, and searching for the fastest route to that node and moving along it. To enhance and improve on this agent, we decided to have Agent 2 keep track of the probability of the nodes surrounding the target. By keeping track of not only where the target is but also where the target could potentially move and the probabilities for those nodes, Agent 2 more efficiently moves through the loop, and ends up moving into the same node as the target simultaneously, rather than moving into the node containing the target. For Agent 5's implementation, we refined the probability distribution process, and considering neighboring nodes' information, Agent 5 demonstrates a more intelligent and practical approach to estimating the target's location. The ability to make informed decisions based on probabilistic inferences significantly enhances the agent's performance, leading to a superior overall search strategy. As a result, Agent 5 sets a higher standard for efficiency and accuracy in the Catch target simulation, promising broader implications in similar applications and domains. For Agent 7, we saw that Agent 6 distributes the probability equally to every node if it doesn’t find the target, but Agent 7 made an improvement on that. If it does not find the target, it will not only lower the node that does not contain the target but also lowers its neighbors since one of their edges is confirmed to not contain the target. This results in a lower probability of being the node the target goes to. And for Agent 7 the reason why it succeeded in running.

If we have more time and resources we could implement a heuristic system into this type of algorithm, and we will store previous data when using Agent 7 so we have a probability where the node is most likely to be on certain steps.