A.I. SEARCH

Bram Schönfeldt, SCHABR004

October 21, 2020

The 4 Corners Problem

a) Formal description

The 4 corners problems involves Pacman, our agent, navigating a maze with the goal of visiting each of the 4 corners of the maze. We are interested in achieving this goal in the shortest path possible. More formally, the 4 corners problem consists of:

1. *State space* - Each state is of the form $S = \{(x, y), (c_1, c_2, c_3, c_4)\}$ where (x, y) represents the x and y co-ordinates of the agent and c_i is a bit representing whether corner i has been visited yet* (0 - unvisited, 1 - visited).
2. *Actions* - $\{\text{North}, \text{South}, \text{East}, \text{West}\}$
3. *Successor function* - For a given state, this returns a set of *admissible actions*, the *updated position* and *number of corners visited* associated with each of these actions, and a cost of 1.
4. *Start state* - The start state is dependent on the maze, but in general takes the form $\{(x_0, y_0), (0, 0, 0, 0)\}$ where (x_0, y_0) is Pacman's initial position and $(0, 0, 0, 0)$ signifies that none of the corners have been visited so far.
5. *Goal state* - the goal state is that Pacman has already visited 3 distinct corners, e.g. $(0, 1, 1, 1)$ and has just arrived at the last distinct corner.

*In my implementation, instead of using bits I kept a set of the co-ordinates of the corners that had been visited

b) Heuristic

The heuristic I chose to implement was the maximum Manhattan distance to an unvisited corner, which can be written as:

$$h(S) = \max_{c_i} |x - x_{c_i}|^2 + |y - y_{c_i}|^2, \quad c_i = 0 \quad \text{(i.e. corner } c_i \text{ is unvisited)}$$

where (x, y) represent Pacman's current position and (x_{c_i}, y_{c_i}) are the co-ordinates of cornerc_i.

Since Pacman has to visit each corner, he has to *at least* travel to the furthest corner. We justify this heuristic further in the next section.

c) Consistency

In order for a heuristic to be *admissible*, the heuristic must be an 'optimistic' lower bound of the true cost of achieving the goal. Firstly, we note that the Manhattan distance between 2 points will always be an optimistic estimate of the true cost of the path between those 2 points. This is because Pacman cannot move diagonally, so at best (without any walls in the way), the distance of a path will be its Manhattan distance. Next, we note that the best case scenario for visiting 4 points will be that all 4 points are along the shortest path to the furthest of these 4 points. Thus, the distance to the furthest point (the maximum Manhattan distance) is an optimistic lower bound. To further show that this heuristic is *consistent*, we show that taking an action of cost c results in a decrease in the cost of the heuristic of at most c. This is trivial since each move has a cost of 1 and to move 1 position will either decrease or increase the Manhattan distance by at most 1.

The Eating-All-The-Dots Problem

a) Formal description

In the eating-all-the-dots problem, Pacman navigates a maze with a number of dots scattered through it with the goal of 'eating' every one of the dots.

1. *State space* - Each state is of the form $S = \{(x, y), \boldsymbol{D}\}$ where (x, y) is Pacman's current position and $\boldsymbol{D}$ is a matrix of bits where entry $d_{i,j} = 1$ if there is food at co-ordinate i, j of the maze.
2. *Actions* - $\{\text{North}, \text{South}, \text{East}, \text{West}\}$
3. *Successor function* - For a given state, this returns a set of admissible actions, the updated position and food matrix $\boldsymbol{D}$ associated with each of these actions, and a cost of 1.
4. *Start state* - $\{(x_0, y_0), \boldsymbol{D}\}$ where (x_0, y_0) is Pacman's initial position and $\boldsymbol{D}$ is the initial food matrix.
5. *Goal state* - the goal state is that for every position i, j of the food matrix, $d_{i,j} = 0$.

b) Heuristic

As before, we define the *distance to the furthest bite of food* as our heuristic. This time, we consider both the Manhattan distance, as well as the *actual maze distance*. The actual maze distance to the furthest dot can can be determined by running *uniform cost search* from the current position to each of the bites of food and returning the maximum path length. Note that determining the actual maze distance is computationally expensive since uniform cost search is exponential in effective depth (i.e. the cost of the actual path to the goal).

We investigate 2 admissible heuristics for the sake of showing how better heuristics typically mean expanding fewer nodes but doing more work per node to compute the heuristic itself

c) Consistency

The distance to the furthest bite of food is an *admissible* heuristic since, in the best case scenario, all of the food lies on the optimal* path to the furthest bite of food. We have shown above that the Manhattan distance is consistent. Using uniform cost search to determine the actual distance to the furthest bite of food is *admissible* since it will return the optimal path to the furthest point, i.e. an optimistic lower bound, and it is consistent since each action of cost 1 will *at most* decrease the cost of the actual distance to the furthest point by 1 (if the action is in the same direction as the optimal path).

* path with the least cost

Assume to the contrary that that there is a path P through all the food with a lower cost that the optimal path O to the furthest food. But then this contradicts that O is optimal.