Artificial Intelligence Laboratory 2: Search Algorithms

Jazib Ahmed, Rasmita Samantaray

Task 1 Path Planning:

1) In the first step we implemented the BFS (Breadth First Search) Algorithm by expanding all the neighbors of the starting node that are neither explored before and valid and putting each of them in the Priority Queue and every time we extracted the first node from the Queue (FIFO) operation and repeated the same step until the queue is empty or the goal state have reached. We kept the track of total cost for moving from start to goal and its route by using a dictionary (dist) which is being updated in every step. Also, we calculated the no expanded nodes to measure the performance of the Algo at the end.
2) In second step we Implemented DFS (Depth First Search) Algorithm by expanding all the neighbors of the starting node that are explored before and valid and and putting each of them in the list and every time we extracted the last node from the Stack (FIFO) operation. Basically, we used the two lists open and closed the closed list keep track of the already explored nodes and open list is the stack itself and the program continuity has the same conditions as BFS whether the stack is empty, or the goal state is reached.
3) In the third step we implemented Greedy Search Algorithm for this purpose we used a class Node which has attributes (parent: which tells about the parent node of a node or its previous node and position: which tells about is position in the map). We used again two lists open and closed which have the same functionality as DFS. In this algorithm we only considered those neighboring nodes not previously being explored who have a better heuristic value as compared to the previous ones. The loop stops when the goal is reached, or all the nodes are being explored. The greedy search will give us better results than BFS and DFS.
4) In the fourth step we implemented A star Euclidean distance algorithm. This algorithm is almost the same as greedy algorithm but here we use heuristic value plus graph value rather than using only heuristic value.
5) In the fifth step we implemented A star Manhattan distance algorithm. This algorithm is almost the same as previous algorithm previous one the only difference is the formulae used for calculating heuristic value.
6) In the last step we implemented the A star customized algorithm. This algorithm is same as the other heuristic function, except it compares whether the goal is nearer from the up or down direction with checking the H-shape obstacle coordinates info and search from the up/down direction

Task-2 Poker Game

1. The same successor function 'get_next_states()' is used to generate the successor states of the poker game.
2. Here the agent wins more than 100 coins within 4 hands. The agent uses the fixed hard cards. Most of the time the optimum solution comes at 2nd hand.

(i) Random search- It selects the search algorithm from DFS, greedy and astar algorithm randomly. If the agent doesn't win 100 coins within 4-hand in 10000 steps, it skips without proceeding further.

(ii) Both BFS and DFS are implemented. The explored nodes for BFS is very high as compared to other search algorithms.

(iii) Greedy search: The performance of greedy search is better compared to BFS and DFS.

(iv) We have attempted to implement Astar algorithm. Here the performance varies from one run to another. At one instance it gives the optimum solution very quickly exploding 40 nodes but, in another instance, it crosses 400.

- All of the solution finds the solution within 4 hands, except random search which varies from run to run, as there is a condition to find the solution within 10000 steps.

- The search algorithm is same as the task-1.

- The print() statement prints the number of nodes expanded for each search method.