* Laborty Practice II (Artificial Intelligence) - Group A - Experiment No. - 2

Name :- Kaustubh Shrikant Kabra.
Class :- Third Year Engineering.
Div :- A
ERP Number :- 38
Department :- Computer Department.
College :- AISSMS's IOIT.

Title :-
A - Star Algorithm

Aim :-
Implement A - star algorithm for any game search problem.

Objective :-
1. To understand abd learn A - star algorithm.
2. To implement A - star algorithm.

Theory :-
A* search algorithm -
A* search algorithm is one of the best and popular techrique used in path-finding and graph traversal. Informally speaking, A* search algorithm, unlike other traversals techniques, it has "brain". What it mean is that if is really a smarth algorithm which separates if from the other conventional algorithms.

This fact is cleared and it is also worth mentioning that many games and web-based maps use this algorithm to find the shortest path very efficiently.

Logic Explanation:-

Consider a square grid having many obstacles and we are given a starting cell and a target cell. We want to reach the target cell from starting cell as quickly as possible. Here A* search algorithm comes to force. At each step it picks the node according to a value 'f' which is a parameters equal to the sum of two other parameters - 'g' and 'h'.

At each step it picks the node/cell having the lowest 'f', and process that node/cell.

g = movement cost to move from starting point to given square on the grid, path generated to get there.

h = estimated movement cost to move from that given square on the grid to the final destination.

Algorithm:-

Create two list, 'Open' list and 'Closed' list.

1. Initialize the open list.
2. Initialize the closed list, Put the sorting node on the open list.
3. While the open list is not empty.
    a) Find the node with least and on the open list call it "q".
    b) pop q off the open list.

```
c) Generate q's 8 successors and set their parents to q.
d) for each successor
    i) if successor is the goal, stop search.
    ii) else compute both g and h for successor.
        successor.g = q.g + dist between successor and q.
        successor.h = dist from goal to successor.
        successor.f = successor.g + successor.h.
    iii) If a node with the same position as successor is in the
        open list which has a lower f than successor, skip this.
    iv) If a node with the same position as successor is in the
        closed list which has a lower and than successor, skip
        this successor otherwise, add the note to the open list.
        end (for loop).
e) Push q on the closed list.
    end (while loop).
```

Limitations :-

Although being the best path finding algorithm, A* search algo doesn't produce the shortest path always, as it relies heavily on heuristics to calculate h:

Heuristics can be calculated using two methods

a) Exact heuristics (time consuming)
b) Approximation Heuristics (less time consuming)
    i) Manhattan Distance
    ii) Diagonal Distance
    iii) Euclidean Distance.

Applications:-

1) They are used in games (for eg. Tower Defense)
2) Find shortest path from one point to another

Time Complexity:-

Worse case time complexity - O(E)
E -> no. of edges in the graph.

Conclusion:-

Thus we have studied the A* search algorithm and its applications and implemented it on a game search problem.