Project

Team member1: Kaustav Basu
Team member2: Monojit Dey

August 2019

Introduction

In this project, Our Pacman agent will find paths through his maze world to reach a particular location. We will build general search algorithms and apply them to Pacman scenarios.

1 Problem Statement

Finding a Fixed Food Dot using Depth First search

DFS progresses by expanding the first child node of the search tree that appears and thus going deeper and deeper until a goal node is found, or until it hits a node that has no children.

Completeness:
yes,it is complete if it is finite; otherwise not(infinite).

Optimal :
No, because it searches depthwise.

Time complexity:

$$1 + [b^2] + [b^3] + - - - + [b^m]$$
$$O(b^m)$$

branching factor= b.
number of layer=m

Space Complexity:
O(bm)
branching factor=b
number of layer=m

Conclusion:
DFS may suffer from non-termination when the length of a path in the search tree is infinite

Algorithm:

2 problem statement

Finding a Fixed Food Dot using Breath First search

A simple strategy in which the root is expanded first then all the root successors are expanded next, then their successors. We visit the search tree level by level that all nodes are expanded at a given depth before any nodes at the next level are expanded.

Completeness:
yes, it is complete because, at least we can reach to the goal state.

Optimal :
yes, it is optimal because BFS always expands the shortest path from the root.

Time complexity:

$$1 + [b^2] + [b^3] + - - - + [b^s]$$

$$O(b^s)$$

branching factor= b.
number of layer=s

Space complexity:

$$O(b^s)$$

branching factor= b.
number of layer=s

Conclusion:
complexity si still a major problem.

Algorithm:

3 problem Statement:

Finding a Fixed Food Dot using Uniform Cost search

the algorithms starts by expanding the root, then expanding the node with the lowest cost from the root, the search continues in this manner for all nodes.

Completeness:
It is obvious that UCS is complete if the cost of each step exceeds some small positive integer, this to prevent infinite loops.

Optimal:
UCS is always optimal in the sense that the node that it always expands is the node with the least path cost.

Time Complexity:
If that solution costs C* and arcs cost at least e, then the "effective depth" is roughly C*/ e

$$O(b^c * /e)$$

Space Complexity:
The space complexity is

$$O(b^c * /e)$$

as the time complexity of UCS.

Conclusion:
UCS can be used instead of BFS in case that path cost is not equal and is guaranteed to be greater than a small positive value e.

Algorithm:

4 Problem Statement:

Finding a Fixed Food Dot using A* search
A* Graph Search, or simply Graph Search, removes this limitation by adding this rule: do not expand the same node more than once.
Heuristic Graph search is optimal only when the forward cost between two successive nodes A and B, given by h(A) - h (B) , is less than or equal to the backward cost between those two nodes g(A -¿ B). This property of graph search heuristic is called consistency.
Consistency: h(A) - h(B) g(A $\rightarrow B$)

Algorithm:

Objective:
$The objective of the game is to accumulate as many points as possible by eating dots, fruits, and blue ghosts.$
$When all of the dots in a stage are eaten, that stage is completed, and the player will advance to the next.$