CS 312 Assignment 1 Report; Team - 21

M V Karthik (200010030), Josyula V N Taraka Abhishek (200010021)

December 26, 2021

1 Introduction

The goal of this task is to teach Pacman how to intelligently find his food by using breadth-first search, depth-first search, and DFID in the state space. The state-space consists of an m x n grid. The start state is (0,0). The goal state is the position of (*) in the grid. The Pacman is allowed to move UP, DOWN, LEFT and RIGHT (except for boundary).

We compare above three alogorithms on:

1. Path Length
2. Number of states explored
3. Dependence of result on the order of neighbours added.

2 Pseudo Codes

Pseudo code for alogorithms is given below.

2.1 Function MoveGen(state)

Algorithm 1: MoveGen(state)

```
Function $MoveGen(state)$ ;
$StatesList \leftarrow [\ ]$;
for $NeighbourNofstateinorder(Down, Up, Right, Left)$ do
    if $Nisnotboundary$ then
        $StatesList.append(N)$
    end
end
$return\ StatesList$
```

2.2 GoalTest(state)

Returns True if input state is goal state

Algorithm 2: GoalTest(state)

```
Function $GoalTest(state)$ ;
if $state.value ==' *'$ then
    $returnTrue$
end
$returnFalse$
```

2.3 BFS()

Implementation of Breadth First search

Algorithm 3: BFS()

```
Function $BFS()$;
if $source \&\& goal == InGraph$ then
    $Q = [\ \ ]$;
    $Q.append(source)$;
end
while $Q.NotEmpty$ do
    $statesExplored + = 1$;
    $current = Q.popleft()$;
    if $GoalTest(current)$ then
        $makePath(source, goal)$;
        $returndistance$;
    end
    $Q.append(MoveGen(current))$
end
```

2.4 DFS()

Implementation of Depth First Search

Algorithm 4: DFS()

```
Function $DFS()$;
if $source\&\&goal == InGraph$ then
    $S = [\ ]$;
    $S.append(source)$;
end
while $S.NotEmpty$ do
    $statesExplored+ = 1$;
    $current = Q.pop()$;
    if $GoalTest(current)$ then
        $makePath(source, goal)$;
        $returndistance$;
    end
    $Q.append(MoveGen(current))$
end
```

2.5 DFID()

Implementation of Iterative Deepening Depth First Search

Algorithm 5: DFID()

```
Function $DFID()$;
if $source\&\&goal == InGraph$ then
    if $Depth == 0$ then
        $return - 1$;
    end
    $distancefromsource(current)$
end
if $GoalTest(current)$ then
    $makePath(source, goal)$;
    $returndistance$;
end
$statesExplored+ = 1$;
$MoveGen(current)$;
$Depth = 1$;
while $True$ do
    $DFID()$;
    if $GoalTest(current)$ then
        $break$
    end
    $Depth+ = 1$
end
```

3 Comparison

The following Table corresponds to Order: Down> Up>Right>Left

Algorithm	Grid	States Explored	Path Length
BFS	2*2	13	10
DFS	2*2	14	10
DFID	2*2	62	10
BFS	4*4	44	28
DFS	4*4	35	32
DFID	4*4	645	32
BFS	6*6	117	38
DFS	6*6	71	46
DFID	6*6	2469	44
BFS	8*8	137	88
DFS	8*8	194	102
DFID	8*8	6906	102
BFS	10*10	348	126
DFS	10*10	257	148
DFID	10*10	29647	148

Table 1: Order: Down> Up>Right>Left

The following Table corresponds to Order: Right>Left>Down>Up.

Algorithm	Grid	States Explored	Path Length
BFS	2*2	13	10
DFS	2*2	13	10
DFID	2*2	63	10
BFS	4*4	42	28
DFS	4*4	48	28
DFID	4*4	578	28
BFS	6*6	117	38
DFS	6*6	93	38
DFID	6*6	2161	38
BFS	8*8	137	88
DFS	8*8	187	88
DFID	8*8	5657	88
BFS	10*10	346	126
DFS	10*10	247	126
DFID	10*10	24925	126

Table 2: Order: Right>Left>Down>Up

4 Conclusion

The dependence of states explored and path Length are summarized below.

Table 3: Dependence on Order of Neighbours

Algorithm	States Explored	Path Length
BFS	True	False
DFS	True	True
DFID	True	True

Inference:

1. The number of states explored by BFS and DFS are generally different but do not vary substantially among themselves.

2. The number of states explored by DFID is relatively larger when compared to those explored by BFS and DFS due to the fact of finding optimal path in DFID with cycles within the graph may require visiting a node multiple times.

3. The order of visiting adjacent cells does affects the result obtained. It is evident from the plots of cumulative analysis seen above, though the difference in the path lengths obtained and number of states explored in each case is not substantially large.

4. BFS always yields paths of shorter (or equal to) lengths than those yielded by DFS.