Alexis Miranda

COMP 157

10/9/2016

Pathfinding Algorithm

Input size:

n x n matrix

Pseudo code:

#BFS

bfs(arr[0..n-1][0..m-1])

currX = startX

currY = startY

fail = False

**while** True:

**if** arr[currX - 1][currY] != "\*":

**if** arr[currX - 1][currY].visited == False:

arr[currX - 1][currY].visited = True

arr[currX - 1][currY].parentX = currX

arr[currX - 1][currY].parentY = currY

**if** arr[currX - 1][currY] == "e":

eposX = currX - 1

eposY = currY

**break**

queueX.append(currX - 1)

queueY.append(currY)

**if** arr[currX + 1][currY] != "\*":

**if** arr[currX + 1][currY].visited == False:

arr[currX + 1][currY].visited = True

arr[currX + 1][currY].parentX = currX

arr[currX + 1][currY].parentY = currY

**if** arr[currX + 1][currY] == "e":

eposX = currX + 1

eposY = currY

**break**

queueX.append(currX + 1)

queueY.append(currY)

**if** arr[currX][currY - 1] != "\*":

**if** arr[currX][currY - 1].visited == False:

arr[currX][currY - 1].visited = True

arr[currX][currY - 1].parentX = currX

arr[currX][currY - 1].parentY = currY

**if** arr[currX][currY - 1] == "e":

eposX = currX

eposY = currY - 1

**break**

queueX.append(currX)

queueY.append(currY - 1)

**if** arr[currX][currY + 1] != "\*":

**if** arr[currX][currY + 1].visited == False:

arr[currX][currY + 1].visited = True

arr[currX][currY + 1].parentX = currX

arr[currX][currY + 1].parentY = currY

**if** arr[currX][currY + 1] == "e":

eposX = currX

eposY = currY + 1

**break**

queueX.append(currX)

queueY.append(currY + 1)

currX = queueX.pop()

currY = queueY.pop()

**if** currX == NULL

fail = True

**break**

**if** fail == False:

**return** eposX, eposY

else:

**return** False

Basic Operation:

comparison (checking if a position is traversable)

Efficiency Class:

