Tên : Nguyễn Quý Văn

Lớp: s22-61th1

Mssv : 1951065673 Số thứ tự : 29

Buổi 8 ngày 9/3/2022

Môn Trí Tuệ Nhân tạo

Map

2 5

3

1 7

8

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Số LL | Đỉnh | Đỉnh kề | Open | close |
| 0 |  |  | A |  |
| 1 | A | D,B | (D,1) (B,2) | A |
| 2 | D | F | (B,2) (F,8) | A,D |
| 3 | B | C,D | (F,8)(C,5) | A,D,B |
| 4 | C | {} | (F,8) | A,D,B,C |

Code BFS

import queue

from mymap import myMap

from modules import showResult

def BFS(start, goal):

q = queue.deque() # khởi tạo queue

q.append(start) # enqueue

previous = [] # khởi tạo previous kiểu dictionary

for city in myMap.keys():

previous.append((city, None))

previous = dict(previous)

while 1:

curCity = q.popleft() # dequeue

for city in myMap[curCity].keys(): # Các điểm có thể đi đến được từ điểm hiện tại

if previous[city] == None: # Nếu chưa đi qua điểm này

q.append(city) # Thêm vào hàng đợi

previous[city] = curCity # Lưu lại dấu vết

if city == goal:

print('Success!')

showResult(previous, start, goal)

return True # Tìm thấy goal

if len(q) == 0:

print('Fail!')

return False # Không thể đi đến goal từ start

start = "A" #điểm bắt đầu

goal = "C" #điểm kết thúc

BFS(start, goal)

File mymap.py

myMap = {

"A": {

"B": { "cost": 2 },

"D": { "cost": 1 }

},

"B": {

"C": { "cost": 5 },

"D": { "cost": 3 }

},

"C": {

"B": { "cost": 9 },

"F": { "cost": 7 }

},

"D": {

"A": { "cost": 1 },

"B": { "cost": 3 },

"F": { "cost": 8 }

},

"F": {

"D": { "cost": 8 },

"C": { "cost": 7 }

}

}

File module

def showResult(previous, start, goal):

curCity = goal # Truy vết từ điểm đích về điểm bắt đầu

print('Result:', curCity, end=' ')

while curCity != start:

curCity = previous[curCity]

print('->', curCity, end=' ')

def showResultWithAttr(previous, start, goal, attr='total\_cost'):

curCity = goal # Truy vết từ điểm đích về điểm bắt đầu

print('Result:', (curCity, previous[curCity][attr]), end=' ')

while curCity != start:

curCity = previous[curCity]['from']

print('->', (curCity, previous[curCity][attr]), end=' ')

def showStep(counter, q, previous, attr='total\_cost'):

print('%d. {' % counter, end=' ')

i = 0

lenQ = len(q)

for v in q:

i += 1

if (i < lenQ):

print((v, previous[v][attr], previous[v]['from']), end=', ')

else:

print((v, previous[v][attr], previous[v]['from']), end=' ')

print('}')

def aweSomeSort(array, previous, sortBy='total\_cost'): # QuickSort (python version)

less = []

equal = []

greater = []

if len(array) > 1:

pivot = previous[array[0]][sortBy]

for city in array:

cost = previous[city][sortBy]

if cost < pivot:

less.append(city)

if cost == pivot:

equal.append(city)

if cost > pivot:

greater.append(city)

return aweSomeSort(less, previous, sortBy) + equal + aweSomeSort(greater, previous, sortBy) # toán tử nối mảng

else:

return array

result:

