

E01 Maze Problem

18340052 何泽

2020 年 9 月 3 日

目录

1 Task	2
2 Codes	2
3 Results	4

1 Task

- Please solve the maze problem (i.e., find the shortest path from the start point to the finish point) by using BFS or DFS (Python or C++)
- The maze layout can be modeled as an array, and you can use the data file `MazeData.txt` if necessary.
- Please send `E01_YourNumber.pdf` to `ai_2020@foxmail.com`, you can certainly use `E01_Maze.tex` as the \LaTeX template.

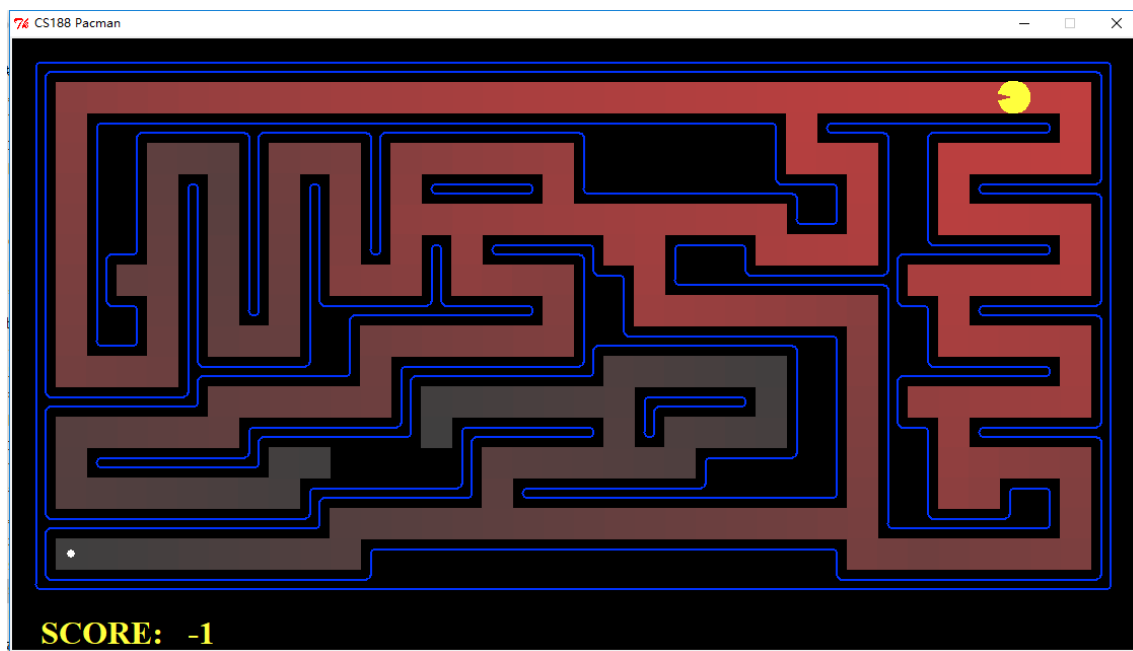


图 1: Searching by BFS or DFS

2 Codes

```
1 def DFS(map, x, y, used):
2     global res
3     if len(res) and len(used) > len(res):
4         return
5     if x == -1 or y == -1 or x == len(map) or y == len(map[0]) or
6         map[x][y] == '1' or (x, y) in used:
7         return
8     elif map[x][y] == 'E':
9         print( '\033[1;30;46m %d \033[0m' % (len(used) + 1), end=
10              " ")
```

```

9         if len(used) + 1 < len(res) or len(res) == 0:
10             res = used
11             res.append((x, y))
12             return
13         else:
14             used.append((x, y))
15             DFS(map, x + 1, y, used[:])
16             DFS(map, x - 1, y, used[:])
17             DFS(map, x, y + 1, used[:])
18             DFS(map, x, y - 1, used[:])
19         used.remove(used[-1])
20
21
22 def print_result(map, path):
23     print(' ')
24     print('\033[1;30;46m          最短路径长度: %d
25           \033[0m' % len(path))
26
27     print('图示: ')
28     for i in range(len(map)):
29         for j in range(len(map[i])):
30             if (i, j) in path[1:-1]:
31                 print('\033[1;32;43m \033[0m', end="")
32             elif map[i][j] == "1":
33                 print('\033[1;33;44m \033[0m', end="")
34             elif map[i][j] == "S":
35                 print('\033[1;30;41mS \033[0m', end="")
36             elif map[i][j] == "E":
37                 print('\033[1;30;45mE \033[0m', end="")
38             else:
39                 print(" ", end="")
40
41         print("")
42     print("")
43
44 if __name__ == "__main__":
45     print('\033[1;30;46m          何泽-18340052-人工智能实验一: Maze
46           \033[0m')
47
48     print('\033[1;30;44m 蓝色是墙 \033[0m', end="")

```

```

45     print( '\033[1;30;41m 红色是起始点  \033[0m', end="")
46     print( '\033[1;30;45m 紫色是终点  \033[0m', end="")
47     print( '\033[1;30;43m 黄色是最短路径  \033[0m')
48     print( '\033[1;30;46m 历史路径长度:  \033[0m', end="")
49     Maze = []
50     res = []
51     with open("./MazeData.txt", 'r') as Maze_og:
52         for i in Maze_og.readlines():
53             if i[0] == '1' or i[0] == '0':
54                 Maze.append(i[:-1])
55
56     for i in range(len(Maze)):
57         for j in range(len(Maze[i])):
58             if Maze[i][j] == 'S':
59                 DFS(Maze, i, j, [])
60                 break
61
62     print_result(Maze, res)

```

3 Results

- 首先说明了迷宫各种颜色的含义
- 我的算法采用了深度优先搜索
- 每找到一条路径，就会记录下来当前路径长度并打印，并给出最短路径长度
- 在图示中用色块画出了迷宫和最短路径，可以看出找出的路径确实是最短的

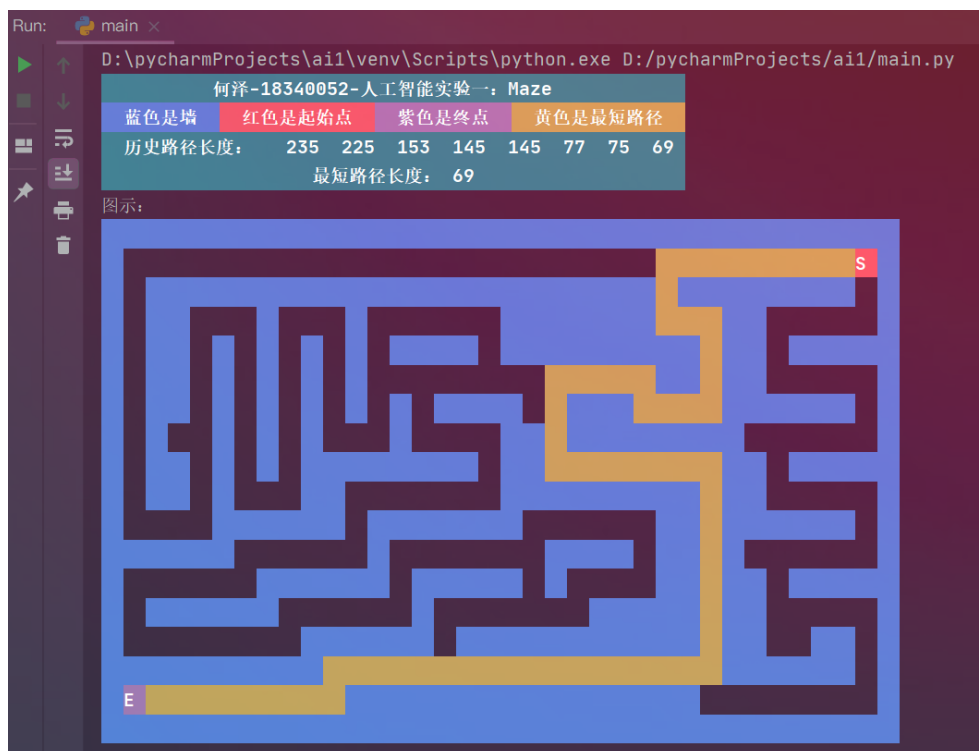


图 2: Result