

# Homework1

请在 3 月 27 日前于学在浙大提交

## 一、写在前面

经过课堂上的学习和课后的复习，相信大家对 8-puzzle 问题和一些简单的搜索方法已经有了一定的了解。为了进一步巩固学习成果，我们将分别使用这些搜索方法(DFS、IDDFS、BFS、UCS 和 A\*) 解决 8-puzzle 问题及其变体。

同时，本次作业也将帮助大家熟悉如何用 python 语言进行编程，并锻炼大家的代码阅读及书写能力。

## 二、题目描述

在基本的 8-puzzle 问题中，空白块能够与在它上下左右位置的数字块进行交换，且交换一次产生的代价为 1。

由于交换产生的代价均相同，在使用 BFS 和 UCS 算法对 8-puzzle 问题进行求解时，我们并不能够很好地区分这两个算法。因此，我们对交换所产生的代价做出如下修改：水平方向上的交换产生的代价为 1，竖直方向上的交换产生的代价为 5。

	1	2
3	4	5
6	7	8

图 1

3	1	2
	4	5
6	7	8

图 2

3	1	2
4		5
6	7	8

图 3

如从图 1 到图 2，空格与数字“3”进行了竖直交换，产生的代价为 5；从图 2 到图 3，空格与数字“4”进行了水平交换，产生的代价为 1。因此，图 1→图 2→图 3 的路径产生的代价为 5+1=6。

## 三、作业内容

在 src 文件夹中，给出了包含 DFS、IDDFS、BFS、UCS 和 A\*这 5 种搜索算法（图搜索）代码框架的文件，分别为 dfs.py、iddfs.py、bfs.py、ucs.py 和 astar.py，请你将**算法核心部分**补充完整。

在 src/utils.py 中，给出了一些有用的函数（启发式函数）、数据结构（队列、优先队列）以及棋盘、移动、搜索等 class，请你将**处理新的状态、搜索结果路径生成、自定义启发式算法**等部分补充完整。

需补充的地方有 TODO 和 NOTE 的提示（一共有 13 处）。

提示：建议先熟悉 utils.py 中的 State、Move、Search 类，并补充 TODO1&2；建议搜索算法的实现顺序为 bfs.py→ucs.py→astar.py，dfs.py→iddfs.py。

```

"""
TODO 1:
    请用合理的表达式替换上面代码段中的 'x_x', '+_+' 和 '=_' ,
    使得这段代码可以正确处理 new_state。
NOTE:
    new_state 为 cur_state 经过一步移动得到的状态,
    move_cost 为从 cur_state 移动到 new_state 所需要的 cost。
"""

```

#### 四、提交要求

请提交一个以 **hw1\_学号.zip** 命名的 zip 格式压缩文件，此压缩文件应当包含：

- 1) 补充完整的 utils.py；
- 2) 补充完整的 dfs.py；
- 3) 补充完整的 iddfs.py；
- 4) 补充完整的 bfs.py；
- 5) 补充完整的 ucs.py；
- 6) 补充完整的 astar.py；
- 7) 作业报告 report.pdf。

注意，请在你补充的代码中必要的地方写上注释。

作业报告要求：

- 1) 附上基本 8-puzzle 问题中 5 个搜索算法的运行结果（请设置 move\_type=1）；
- 2) 附上 8-puzzle 变体中 5 个搜索算法的运行结果（请设置 move\_type=2）；

```

if __name__ == "__main__":
    init_state = State(((1, 2, 3),
                        (4, 5, 6),
                        (7, 8, 0)))
    goal_state = State(((7, 3, 1),
                       (8, 0, 4),
                       (6, 5, 2)))
    solver = BreadthFirstSearch(init_state, goal_state, move_type=1)
    solver.solve()
    solver.print()

```

- 3) 对每个问题中的 5 个结果进行简要的比较和说明，了解对应的算法性质（最优性、探索的状态数、复杂度等）；
- 4) 阐述你设计的启发式函数，并使用该函数对应的代码（heuristic\_func2）再次在 8-puzzle 及其变体设置下运行 astar.py，比较该函数与原 utils.py 文件中给出的启发式函数（heuristic\_func1）的不同表现；
- 5) 报告的具体格式没有要求。

## 五、成绩评定

代码补充实现占 50%，report 占 50%。

注意：在完成相关作业时，切勿抄袭。

## 六、Q&A

### 1. 如何查看运行结果？

以 dfs.py 为例，在将对应的文件补充完整后，打开终端，在 src 目录下输入 `python dfs.py`，即可看到相应的打印结果。

### 2. 输出结果太长怎么办？

由于搜索算法的搜索策略不同，生成的解的路径长度也不一样，如 DFS 算法找到的路径长度为几百。对于过长的输出，请保留输出结果的最后 30 个 step，一个 step 的输出如下图。

```
Step 11
7 1 2
0 4 3
8 6 5
```

### 3. 有其他疑问怎么办？

请酌情使用搜索引擎或咨询助教。