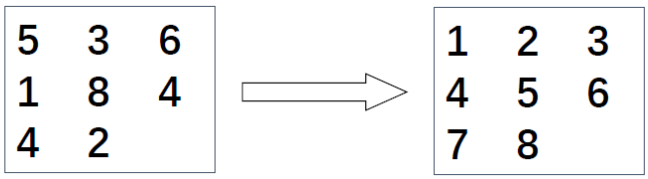
数字华容道 A*算法实验报告

一、 问题重述

数字华容道一般是指在3×3的九宫格上随意排列 1 至 8 共八位数字，通过利用剩余的一个空格进行数字的上下左右移动，最终达到数字按行有序排列的问题。其大致过程如下图所示：



类似的问题还有最终状态不同的八数码问题，但本质上都可以看作是从初始状态转化为有序终态的搜索问题，搜索空间是整个九宫格的各种可能排列的状态。

由此，数字华容道同样可以扩展为 4 阶 5 阶甚至任意更高阶的问题，当然越高阶的搜索空间越大，也会产生更大的时间和空间开销。

二、 A*算法求解概述

A*（AStar）算法是一种很常用的路径查找和图形遍历算法。它有较好的性能和准确度，可以被认为是 Dijkstra 算法的扩展，但由于借助启发函数的引导，A*算法通常拥有更好的性能。

对于每个可能的搜索状态，我们将其称作节点，A*算法通过函数$f(n)$来计算每个节点的优先级：

$$f(n) = g(n) + h(n)$$

其中$f(n)$是综合优先级，该节点的$f(n)$越小则优先级越大。$g(n)$是节点 n 距离初始节点的代价，$h(n)$是节点 n 距离终点的预计代价，也即 A*算法的启发函数。在运算过程中，每次选择优先级最高的节点进行遍历，并且 A*算法构建了 open 表和 close 表来分别表示待遍历的节点和已经遍历的节点。

在本题中，$h(n)$的定义一般有两种方式：

1. $h_1(n)$ =“不在位”的将牌数
2. $h_2(n)$ =将牌“不在位”的距离之和

可见第二种启发函数的定义更符合真实情况，同时由于数字华容道的规则，故选用所有“不在位”将牌的曼哈顿距离之和作为$h(n)$的定义。

本题算法的描述大致如下：

```
* 初始化 open_set 和 close_set；
* 将起点加入 open_set 中，并设置优先级为 0（优先级最高）；
* while(open):
    * 从 open_set 中选取优先级最高的节点 n：
    * 如果节点 n 为终点，则：
        * 从终点开始逐步追踪 parent 节点，一直达到起点；
```

```
        * 返回找到的结果路径，算法结束；
    * 如果节点 n 不是终点，则：
        * 将节点 n 从 open_set 中删除，并加入 close_set 中；
        * 遍历节点 n 所有的邻近节点：
            * 如果邻近节点 m 在 close_set 中，则：
                * 判断是否需要更新
            * 如果邻近节点 m 在 open_set 中，则：
                * 判断是否需要更新
            * 如果邻近节点 m 既不在 close_set 中也不在 open_set 中，则：
                * 设置节点 m 的 parent 为节点 n
                * 计算节点 m 的优先级
                * 将节点 m 加入 open_set 中
```

在本题中进行了高阶的扩展，对于自定输入的阶数，会在算法之前随机生成该阶数的棋盘作为初始状态，通过 A*算法进行搜索。但由于 4 阶及以上的搜索空间较大，所以程序在搜索速度上还有一定的改进空间。

三、 讨论

1. 数字华容道是否有解

首先按照行从上往下，列从左往右的顺序可以把棋盘上的数字排列为一串序列。按照游戏规则，在移动数字的过程中，棋盘数字序列的逆序数的奇偶性不会改变。由于终态是升序排列的，即可看做逆序数是偶数，所以只要初态数字序列的逆序数为偶数，则该问题一定有解，否则无解。在算法随机生成了棋盘之后，首先进行了逆序数的计算，由此可以得到是否有解，再进行后续搜索。

2. 附加题：修改 A*算法，当问题存在多于 n 个解时，求解前 n 个最好的解

由于 A*算法的核心在于启发函数$f(n) = g(n) + h(n)$，该启发函数的设置会影响 A*算法的搜索过程。如果$h(n)$始终小于等于节点 n 到终点的代价，则 A*算法保证一定能够找到最短路径。但是当$h(n)$的值越小，算法将遍历越多的节点，也就导致算法越慢。在极端情况下，若$h(n)$趋于零，则将由$g(n)$决定节点的优先级，此时 A*算法退化为 Dijkstra 算法；若$h(n)$远大于$g(n)$，则此时只有$h(n)$产生效果，此时 A*算法趋近于最佳优先（Best First）搜索算法。

所以要想找到 n 个最好的解，一是要保证算法在搜索到第一个解之后要继续搜索下去，二是可以减小$h(n)$用于扩大搜索的范围。此外，还可以为于$h(n)$函数设置权重值$w(n)$，构造动态衡量启发式函数，即$f(n) = g(n) + w(n) * h(n)$，其中$w(n) \geq 1$。通过对$w(n)$值的控制，可以对搜索返回和搜索速度进行调整来进行优化。

四、 结果展示

- Case 1:

```
Please input the dimension of the grid:3
The init state is:
[3, 5, 0]
```

```
[6, 1, 8]
[2, 7, 4]
The problem has a solution, in searching...
Search times: 1076  Steps: 22
The each step of moving:
* STEP 0 :
[3, 5, 0]
[6, 1, 8]
[2, 7, 4]
F(n)= 18 , G(n)= 0 , H(n)= 18
--------------------
* STEP 1 :
[3, 0, 5]
[6, 1, 8]
[2, 7, 4]
F(n)= 21 , G(n)= 1 , H(n)= 20
--------------------
* STEP 2 :
[3, 1, 5]
[6, 0, 8]
[2, 7, 4]
F(n)= 20 , G(n)= 2 , H(n)= 18
--------------------
* STEP 3 :
[3, 1, 5]
[0, 6, 8]
[2, 7, 4]
F(n)= 21 , G(n)= 3 , H(n)= 18
--------------------
* STEP 4 :
[3, 1, 5]
[2, 6, 8]
[0, 7, 4]
F(n)= 20 , G(n)= 4 , H(n)= 16
--------------------
* STEP 5 :
[3, 1, 5]
[2, 6, 8]
[7, 0, 4]
F(n)= 19 , G(n)= 5 , H(n)= 14
--------------------
* STEP 6 :
[3, 1, 5]
[2, 6, 8]
```

```
[7, 4, 0]
F(n)= 18 , G(n)= 6 , H(n)= 12
--------------------
* STEP 7 :
[3, 1, 5]
[2, 6, 0]
[7, 4, 8]
F(n)= 19 , G(n)= 7 , H(n)= 12
--------------------
* STEP 8 :
[3, 1, 5]
[2, 0, 6]
[7, 4, 8]
F(n)= 20 , G(n)= 8 , H(n)= 12
--------------------
* STEP 9 :
[3, 0, 5]
[2, 1, 6]
[7, 4, 8]
F(n)= 23 , G(n)= 9 , H(n)= 14
--------------------
* STEP 10 :
[0, 3, 5]
[2, 1, 6]
[7, 4, 8]
F(n)= 24 , G(n)= 10 , H(n)= 14
--------------------
* STEP 11 :
[2, 3, 5]
[0, 1, 6]
[7, 4, 8]
F(n)= 23 , G(n)= 11 , H(n)= 12
--------------------
* STEP 12 :
[2, 3, 5]
[1, 0, 6]
[7, 4, 8]
F(n)= 22 , G(n)= 12 , H(n)= 10
--------------------
* STEP 13 :
[2, 3, 5]
[1, 4, 6]
[7, 0, 8]
F(n)= 21 , G(n)= 13 , H(n)= 8
```

```
--------------------
* STEP 14 :
[2, 3, 5]
[1, 4, 6]
[7, 8, 0]
F(n)= 20 , G(n)= 14 , H(n)= 6
--------------------
* STEP 15 :
[2, 3, 5]
[1, 4, 0]
[7, 8, 6]
F(n)= 23 , G(n)= 15 , H(n)= 8
--------------------
* STEP 16 :
[2, 3, 0]
[1, 4, 5]
[7, 8, 6]
F(n)= 24 , G(n)= 16 , H(n)= 8
--------------------
* STEP 17 :
[2, 0, 3]
[1, 4, 5]
[7, 8, 6]
F(n)= 25 , G(n)= 17 , H(n)= 8
--------------------
* STEP 18 :
[0, 2, 3]
[1, 4, 5]
[7, 8, 6]
F(n)= 26 , G(n)= 18 , H(n)= 8
--------------------
* STEP 19 :
[1, 2, 3]
[0, 4, 5]
[7, 8, 6]
F(n)= 25 , G(n)= 19 , H(n)= 6
--------------------
* STEP 20 :
[1, 2, 3]
[4, 0, 5]
[7, 8, 6]
F(n)= 24 , G(n)= 20 , H(n)= 4
--------------------
* STEP 21 :
```

```
[1, 2, 3]
[4, 5, 0]
[7, 8, 6]
F(n)= 23 , G(n)= 21 , H(n)= 2
--------------------
* STEP 22 :
[1, 2, 3]
[4, 5, 6]
[7, 8, 0]
F(n)= 22 , G(n)= 22 , H(n)= 0
```

- Case 2:

```
Please input the dimension of the grid:3
The init state is:
[1, 2, 3]
[8, 4, 6]
[7, 5, 0]
The problem has a solution, in searching...
Search times: 257  Steps: 14
The each step of moving:
* STEP 0 :
[1, 2, 3]
[8, 4, 6]
[7, 5, 0]
F(n)= 4 , G(n)= 0 , H(n)= 4
--------------------
* STEP 1 :
[1, 2, 3]
[8, 4, 0]
[7, 5, 6]
F(n)= 7 , G(n)= 1 , H(n)= 6
--------------------
* STEP 2 :
[1, 2, 3]
[8, 0, 4]
[7, 5, 6]
F(n)= 10 , G(n)= 2 , H(n)= 8
--------------------
* STEP 3 :
[1, 2, 3]
[0, 8, 4]
[7, 5, 6]
F(n)= 11 , G(n)= 3 , H(n)= 8
--------------------
```

```
* STEP 4 :
[1, 2, 3]
[7, 8, 4]
[0, 5, 6]
F(n)= 12 , G(n)= 4 , H(n)= 8
--------------------
* STEP 5 :
[1, 2, 3]
[7, 8, 4]
[5, 0, 6]
F(n)= 13 , G(n)= 5 , H(n)= 8
--------------------
* STEP 6 :
[1, 2, 3]
[7, 0, 4]
[5, 8, 6]
F(n)= 14 , G(n)= 6 , H(n)= 8
--------------------
* STEP 7 :
[1, 2, 3]
[7, 4, 0]
[5, 8, 6]
F(n)= 13 , G(n)= 7 , H(n)= 6
--------------------
* STEP 8 :
[1, 2, 3]
[7, 4, 6]
[5, 8, 0]
F(n)= 12 , G(n)= 8 , H(n)= 4
--------------------
* STEP 9 :
[1, 2, 3]
[7, 4, 6]
[5, 0, 8]
F(n)= 15 , G(n)= 9 , H(n)= 6
--------------------
* STEP 10 :
[1, 2, 3]
[7, 4, 6]
[0, 5, 8]
F(n)= 16 , G(n)= 10 , H(n)= 6
--------------------
* STEP 11 :
[1, 2, 3]
```

```
[0, 4, 6]
[7, 5, 8]
F(n)= 17 , G(n)= 11 , H(n)= 6
--------------------
* STEP 12 :
[1, 2, 3]
[4, 0, 6]
[7, 5, 8]
F(n)= 16 , G(n)= 12 , H(n)= 4
--------------------
* STEP 13 :
[1, 2, 3]
[4, 5, 6]
[7, 0, 8]
F(n)= 15 , G(n)= 13 , H(n)= 2
--------------------
* STEP 14 :
[1, 2, 3]
[4, 5, 6]
[7, 8, 0]
F(n)= 14 , G(n)= 14 , H(n)= 0
```

- Case 3:

```
Please input the dimension of the grid:5
The init state is:
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 17, 18, 19, 20]
[0, 21, 23, 24, 22]
The problem has a solution, in searching...
Search times: 8728  Steps: 20
The each step of moving:
* STEP 0 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 17, 18, 19, 20]
[0, 21, 23, 24, 22]
F(n)= 8 , G(n)= 0 , H(n)= 8
--------------------
* STEP 1 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
```

```
[11, 12, 13, 14, 15]
[16, 17, 18, 19, 20]
[21, 0, 23, 24, 22]
F(n)= 7 , G(n)= 1 , H(n)= 6
--------------------
* STEP 2 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 0, 18, 19, 20]
[21, 17, 23, 24, 22]
F(n)= 10 , G(n)= 2 , H(n)= 8
--------------------
* STEP 3 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 18, 0, 19, 20]
[21, 17, 23, 24, 22]
F(n)= 11 , G(n)= 3 , H(n)= 8
--------------------
* STEP 4 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 18, 23, 19, 20]
[21, 17, 0, 24, 22]
F(n)= 12 , G(n)= 4 , H(n)= 8
--------------------
* STEP 5 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 18, 23, 19, 20]
[21, 17, 24, 0, 22]
F(n)= 13 , G(n)= 5 , H(n)= 8
--------------------
* STEP 6 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 18, 23, 19, 20]
[21, 17, 24, 22, 0]
F(n)= 12 , G(n)= 6 , H(n)= 6
```

```
--------------------
* STEP 7 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 18, 23, 19, 0]
[21, 17, 24, 22, 20]
F(n)= 15 , G(n)= 7 , H(n)= 8
--------------------
* STEP 8 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 18, 23, 0, 19]
[21, 17, 24, 22, 20]
F(n)= 18 , G(n)= 8 , H(n)= 10
--------------------
* STEP 9 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 18, 0, 23, 19]
[21, 17, 24, 22, 20]
F(n)= 21 , G(n)= 9 , H(n)= 12
--------------------
* STEP 10 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 18, 24, 23, 19]
[21, 17, 0, 22, 20]
F(n)= 22 , G(n)= 10 , H(n)= 12
--------------------
* STEP 11 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 18, 24, 23, 19]
[21, 17, 22, 0, 20]
F(n)= 21 , G(n)= 11 , H(n)= 10
--------------------
* STEP 12 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
```

```
[11, 12, 13, 14, 15]
[16, 18, 24, 0, 19]
[21, 17, 22, 23, 20]
F(n)= 22 , G(n)= 12 , H(n)= 10
--------------------
* STEP 13 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 18, 0, 24, 19]
[21, 17, 22, 23, 20]
F(n)= 23 , G(n)= 13 , H(n)= 10
--------------------
* STEP 14 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 0, 18, 24, 19]
[21, 17, 22, 23, 20]
F(n)= 24 , G(n)= 14 , H(n)= 10
--------------------
* STEP 15 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 17, 18, 24, 19]
[21, 0, 22, 23, 20]
F(n)= 23 , G(n)= 15 , H(n)= 8
--------------------
* STEP 16 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 17, 18, 24, 19]
[21, 22, 0, 23, 20]
F(n)= 22 , G(n)= 16 , H(n)= 6
--------------------
* STEP 17 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 17, 18, 24, 19]
[21, 22, 23, 0, 20]
F(n)= 21 , G(n)= 17 , H(n)= 4
```

```
--------------------
* STEP 18 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 17, 18, 0, 19]
[21, 22, 23, 24, 20]
F(n)= 22 , G(n)= 18 , H(n)= 4
--------------------
* STEP 19 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 17, 18, 19, 0]
[21, 22, 23, 24, 20]
F(n)= 21 , G(n)= 19 , H(n)= 2
--------------------
* STEP 20 :
[1, 2, 3, 4, 5]
[6, 7, 8, 9, 10]
[11, 12, 13, 14, 15]
[16, 17, 18, 19, 20]
[21, 22, 23, 24, 0]
F(n)= 20 , G(n)= 20 , H(n)= 0
```

- Case 4:

```
Please input the dimension of the grid:3
The init state is:
[6, 5, 4]
[2, 7, 3]
[8, 1, 0]
No solution!
```

References

[1] https://github.com/roadwide/AI-Homework/tree/master/Search%20Algorithms/Astar

[2] https://zhuanlan.zhihu.com/p/54510444

[3] https://zhuanlan.zhihu.com/p/80707067

[4] https://blog.csdn.net/u011008379/article/details/40144147

[5] https://blog.csdn.net/denghecsdn/article/details/78778769