A Star Search

A\*搜索

问题：

对于的矩阵

点可以与上下左右的相邻点交换位置，除此之外不能随意改变位置，将该矩阵变成

求最少变换次数以及变化经过，若将矩阵的初始状态看作起点，最终状态看作终点，即搜索到的最短路径。

本问题的原型是“八数码问题”。

解法：

与之前问题不同，本问题将每种矩阵状态看作一个节点，是一种时间上的状态搜索。本文用A\*搜索来解决该问题，A\*算法是一种启发式搜索，与DFS和BFS这种无差别搜索不同，A\*算法设置一个评价函数来计算节点的搜索代价（到目标的距离），优先搜索那些离目标最近的点，从而提高搜索效率。

A\*算法的评价函数，其中是节点，表示点到end的评价距离，表示从beg到点的实际最短距离，表示从点到end的估算距离。在A\*算法的等待队列中，总是优先选取最小的点进行搜索。

在本问题中矩阵的估算距离为矩阵与end在对应位置（其中）上不同数字的数量之和，：

对于下面的矩阵，，：

与之前的问题不同，本问题中的节点是一种矩阵状态。之前的解法中我们用染色的方式来标记节点是否被访问过，编码实现时可以用数组来标记节点i的颜色。而矩阵状态是无法作为数组下标的，不过我们可以用哈希表来记录矩阵状态x是否被访问过，以及从节点beg到达x的节点距离。矩阵可以通过下面这两种方式分别映射为字符串或数字：

设置openqueue、closequeue和gscore分数表。gscore分数表是一个哈希表，用来存储每个节点的。openqueue是一个优先队列，与之前搜索算法中的队列功能相同，用于管理等待访问的节点，但是我们需要从openqueue中总是优先取出可能离end最近的节点，因此其优先级为。closequeue是一个哈希表，用于存储所有已经访问过的节点距离。

当x点与end点相同时，x点中每个位置的值都和end点相同。过程如下：

1. 将beg点加入open队列，g分数表中；
2. 从open队列中取出f值最小的（唯一的）点，将它加入close队列，该点的x与上下左右交换位置后的状态有，这3个状态可以看作该点的相邻点（和问题1中二维方格的上下左右4个相邻格子类似），这3个状态不属于open队列和close队列，将这3个点加入open队列并设置父节点都为并计算g值；
3. 从open队列中取出f值最小的点，将它加入close队列，该点的x与上下左右交换位置后的状态有2个点，其中属于close队列，跳过该点，不属于open队列和close队列，将该点加入open队列，设置父节点为并计算g值；
4. 从open队列中取出f值最小的点，将它加入close队列，该点的x与上下左右交换位置后的状态有2个点，其中属于close队列，跳过该点，不属于open队列和close队列，将该点加入open队列，设置父节点为并计算g值；
5. 从open队列中取出f值最小的点，将它加入close队列，该点的x与上下左右交换位置后的状态有4个点，其中属于close队列，跳过该点，不属于open队列和close队列，将这3点加入open队列，设置父节点为并计算g值；
6. 从open队列中取出f值最小的点，将它加入close队列，该点的x与上下左右交换位置后的状态有3个点，其中属于close队列，跳过该点，不属于open队列和close队列，将这2点加入open队列，设置父节点为并计算g值；
7. 从open队列中取出f值最小的点，将它加入close队列，该点的x与上下左右交换位置后的状态有3个点，其中属于close队列，跳过该点，不属于open队列和close队列，将这2点加入open队列，设置父节点为并计算g值；
8. 从open队列中取出f值最小的点（open队列中存在多个的点，但是该点h值最小因此优先选择），将它加入close队列，该点的x与上下左右交换位置后的状态有2个点，其中属于close队列，跳过该点，不属于open队列和close队列，将该点加入open队列，设置父节点为并计算g值；
9. 从open队列中取出f值最小的点，将它加入close队列，该点的x与上下左右交换位置后的状态有3个点，其中属于close队列，跳过该点，不属于open队列和close队列，将这2点加入open队列，设置父节点为并计算g值；
10. 从open队列中取出f值最小的点，将它加入close队列，该点的x与上下左右交换位置后的状态有2个点，其中属于close队列，跳过该点，不属于open队列和close队列，将该点加入open队列，设置父节点为并计算g值；
11. 从open队列中取出f值最小的点，将它加入close队列，该点的x与上下左右交换位置后的状态有3个点，其中属于close队列，跳过该点，不属于open队列和close队列，将该2点加入open队列，设置父节点为并计算g值；
12. 从open队列中取出f值最小的点，将它加入close队列，该点的x与上下左右交换位置后的状态有3个点，其中属于close队列，跳过该点，不属于open队列和close队列，将该2点加入open队列，设置父节点为并计算g值；
13. 从open队列中取出f值最小的点，将它加入close队列，该点的x与上下左右交换位置后的状态有3个点，其中属于close队列，跳过该点，不属于open队列和close队列，将该2点加入open队列，设置父节点为并计算g值；
14. 从open队列中取出f值最小的点，将它加入close队列，该点的x与上下左右交换位置后的状态有3个点，其中属于close队列，跳过该点，不属于open队列和close队列，将该2点加入open队列，设置父节点为并计算g值；
15. 从open队列中取出f值最小的点，将它加入close队列，该点的x与上下左右交换位置后的状态有4个点，其中属于close队列，跳过该点，不属于open队列和close队列，将该3点加入open队列，设置父节点为并计算g值；
16. 从open队列中取出f值最小的点，将它加入close队列，该点的x与上下左右交换位置后的状态有2个点，其中属于close队列，跳过该点，不属于open队列和close队列，将该点加入open队列，设置父节点为并计算g值；
17. 从open队列中取出f值最小的点，将它加入close队列，该点的x与上下左右交换位置后的状态有4个点，其中属于close队列，跳过该点，不属于open队列和close队列，将该3点加入open队列，设置父节点为并计算g值；
18. 每一步从open队列中取出f值最小的（f值相同时优先选h值最小的）点x，比较x点与end点，(a)若open队列为空，则没有找到end，算法结束，(b)若则找到end点，算法结束，(c)继续寻找，将x点加入close队列，x点相邻的其他节点y有以下可能：(a)若y点属于close队列则直接跳过该点，(b)若y点不属于close队列，不属于open队列，将y点加入open队列，父节点设置为x，计算、、值，(c)若y点不属于close队列，属于open队列，说明y点已经被访问过，重新计算以x点为父节点的y的、、值，若更小则更新y点的信息和父节点为x点，若并不比原本的路径更短则保持现在的y点不变；

对于八进制数码问题，A\*搜索在最坏情况下时间复杂度为。

八数码问题：

<http://www.d.umn.edu/~jrichar4/8puz.html>

<https://www.cs.princeton.edu/courses/archive/fall12/cos226/assignments/8puzzle.html>