**AStar：数字华容道实验报告**

Kotoricyann

项目Github：https://github.com/Kotoricyann/RUC-AI-Course

1. 问题定义

数字华容道问题是让打乱的N\*N-1个连续数字在N\*N的方阵中移动，最终实现按顺序排列，空格在右下角的问题。

起始状态如下图所示



该图片展示的是一个3\*3的数字华容道问题，棋盘上有一个空位，8个数字，数字可以挪到空位上，通过移动棋子，让棋盘上的数字有序排列，最终达到下图的格式：



该问题可以看作是一个搜索问题，搜索空间是棋盘全部出现的可能情况。有些情况是不可能有解的，比如下面这种情况：

1. 2 3

4 5 6

8 7

如上这种情况是不可解的，因为数字排列的逆序数为奇数。为了避免这种情况，在生成初始状态的时候需要保证逆序数为偶数。

1. 算法概要

Astar（A\*）算法是一种启发式直接搜索算法，最初被用于静态路网中求解最短路径，也被用于解决许多搜索问题。算法中的距离估算值与实际值越接近，最终搜索速度越快。对于给定的状态x距离估计值f(x)由两部分组成：距离起始情况的距离g(x)，与距离最终情况的距离h(x)。f(x)=g(x)+h(x)。其中h(x)是状态x距离最终状态真实距离的下限，即h(x)小于等于状态x距离最终状态真实距离。h(x)越接近真实距离，搜索速度越快，搜索范围越小。离真实距离越远，搜索速度越慢，搜索范围越广。

在本题中，g(x)的可以定义为搜索状态的所在层数，也就是起始状态到本状态的步数。h(x)可以有如下两种定义：

1. 定义为不在位数字数；
2. 定义为每个节点到最终状态距离的步数和。

可见第二种定义计算出的评分会比第一种更大，第二种也更接近真实距离。所以本实现采用了第二种定义方法。算法具体的伪代码如下：

1. 随机生成起始起始状态a
2. 将起始状态a加入到openSet中（openSet为以评分函数为标准的优先队列）
3. While（！openSet为空）
4. x = openSet.top()
5. closeSet.insert(x)
6. 检查x的派生是否为最终状态，如果是
7. 结果展示
8. 附加题