**Sicily 1151 魔板 解题报告**

*13331231*

*孙圣*

1. 题目大意

有一个由8个数码组成的魔板。给定初始状态，将终止状态和最大步数作为输入，输入-1代表结束。有三种对魔板的操作：A，交换上下行；B，将每一位数右移，最右的数移到最左边；C，中间的四个数顺时针旋转一格。如果能在最大步数内将魔板移到终止状态，则输出步数和操作符序列；否则，输出-1。

1. 算法思想及主要的数据结构

这题给定初始态要寻找目标态，所以是一道搜索的问题。但由于数据量巨大，用深度优先搜索很有可能浪费大量的时间和空间，因此并不合理。所以主要利用广度优先搜索来求解问题，主要使用的数据结构为队列。在每一个节点处，将三个操作后形成的不同状态压入栈中，因此这是一棵三叉树。如果不进行判断是否重复，很有可能进入死循环浪费时间，所以要判断重复从而减少时间的消耗。对于重复的判断可以使用set，也可以利用数组。

1. 解题思路

主要利用康托展开来实现对存储空间的压缩。

由于8个数码并没有重复，可以将它们视为全排列。8个数的全排列一共有8! = 40320种情况。所以我们可以使用一个布尔类型的大小为40320的数组来帮助我们判断某一状态是否已经出现过。康托展开的公式：X = a[n] \* (n-1)! + a[n-1] \* (n-2)! +...+ a[i] \* (i-1)! +...+ a[1] \* 0!，其中a[i]是指比处在i下标处的数小的且之前未出现的数的个数。因此，对于每个排列，我们都可以根据康托展开得到一个唯一的数来作为整个排列的标识。

利用栈来实现广搜，对于每一个节点，如果其为目标状态则直接跳出循环输出结果，否则分别调用三个操作得到新的状态节点。如果新的状态节点在之前已经出现过，则不将其放入队列；如果之前并未出现，则压入队列。不断的循环，直到步数超过规定步数或者找到目标状态。

1. 算法描述

定义结构体state：

struct state {

short num[8];

string op;

};

保存8个数码的状态和操作符

进行A操作的函数，返回新的state结构体：

struct state opa(struct state& input)；

进行B操作的函数，返回新的state结构体：

struct state opb(struct state& input)；

进行C操作的函数，返回新的state结构体：

struct state opc(struct state& input)；

进行康托展开的函数，输入为储存8个数码的数组：

int cantor(short a[])；

int factorial[8]; 储存7!到0!的值

bool b[8]; 储存第i位数是否被访问过

count 储存比a[i]小且尚未出现的数的个数

返回康托展开的值

main()

1. 程序注释清单
2. 测试数据
3. 分析与优化