

神犇(shenben)

“经历只有自己去体会，别人的眼中只有结果”

博客园

首页

新随笔

联系

管理

公告

Visitors

258,106

8,224

6,335

5,939

4,990

3,161

3,128

1,197

846

734

609

549

FLAG counter

昵称：神犇(shenben)

园龄：2年10个月

粉丝：146

关注：7

+加关注

< 2018年12月 >

日	一	二	三	四	五	六
25	26	27	28	29	30	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31	1	2	3	4	5

搜索

找我看

谷歌搜索

常用链接

我的随笔

我的评论

我的参与

最新评论

我的标签

我的标签

I do what I do(1)

随笔分类(1164)

OI致梦想(4)

STL系列(3)

拔高系列(7)

博弈(6)

常用思想——倍增(5)

常用思想——分数规划(2)

常用思想——莫队(5)

常用思想——排序整理(1)

动态规划——DP(124)

动态规划——概率dp(8)

多项式(1)

分块(5)

分治——CDQ/整体二分.....(8)

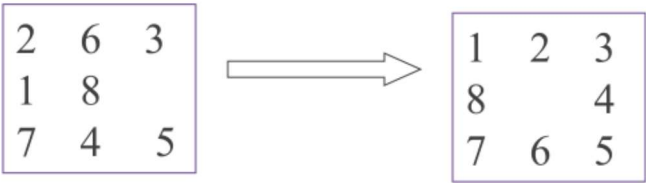
高等数学——FFT(1)

luogu P1379 八数码难题（A*算法入门详细讲解）

启发式搜索算法(A算法)

- 在BFS算法中，若对每个状态n都设定估价函数 $f(n)=g(n)+h(n)$ ，并且每次从Open表中选节点进行扩展时，都选取f值最小的节点，则该搜索算法为启发式搜索算法，又称A算法。
- $g(n)$ ：从起始状态到当前状态n的代价
- $h(n)$ ：从当前状态n到目标状态的估计代价

A算法的例子--八数码



定义估价函数：

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

$g(n)$ 为从初始节点到当前节点的步数

$h(n)$ 为当前节点“不在位”的方块数

高级数据结构——LCT(3)
 高级数据结构——Splay(9)
 高级数据结构——加权并查集(4)
 高级数据结构——树链剖分(9)
 高级数据结构——树套树(2)
 高级数据结构——线段树(23)
 高级数据结构——主席树(13)
 高级数据结构——左偏树(3)
 高精度(10)
 计算几何(8)
 计算几何——(离散化)扫描线(3)
 模板区、原理整理(27)
 数学推论——莫比乌斯反演(12)
 数学推论——逆元(5)
 数学推论——欧拉函数(7)
 数学推论——容斥原理(2)
 数学推论——数论区(36)
 数学推论——组合数学(9)
 搜索(55)
 套题(42)
 图论(7)
 图论——二分图(2)
 图论——网络流(53)
 歪门邪道——rope/pb_ds(4)
 线性函数——高斯消元(3)
 线性函数——矩阵乘法(15)
 在线评测——BZOJ(18)
 在线评测——codevs(307)
 在线评测——cojs(10)
 在线评测——HDU(9)
 在线评测——NOI.cn(20)
 在线评测——NOIP历年题目(54)
 在线评测——POJ(92)
 在线评测——洛谷(55)
 注意细节、一点小技巧(12)
 字符串处理——AC自动机(9)
 字符串处理——kmp/exkmp(5)
 字符串处理——后缀数组(16)
 字符串处理——回文串
 manacher(4)
 字符串处理——字符串哈希(7)

随笔档案(985)

2018年8月 (1)
 2017年5月 (18)
 2017年4月 (71)
 2017年3月 (101)
 2017年2月 (94)
 2017年1月 (89)
 2016年12月 (13)
 2016年11月 (43)
 2016年10月 (49)
 2016年9月 (65)
 2016年8月 (119)
 2016年7月 (103)
 2016年6月 (133)
 2016年5月 (86)

QQ空间

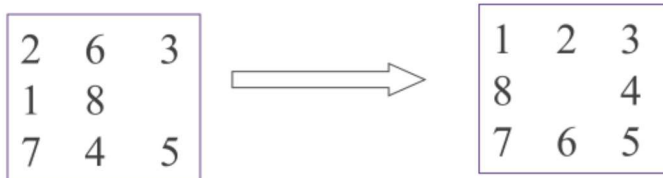
积分与排名

积分 - 124087
 排名 - 3103

阅读排行榜

1. 匈牙利算法 (二分图) (16456)
 2. NOI 1.7编程基础之字符串(35题) (4083)
 3. 最短路径—Dijkstra算法(2716)
 4. 3118 高精度练习之除法(2312)

h计算举例



2,6,1,8,4,5都不在位，因此 $h(n) = 6$

A*算法

- A算法中的估价函数若选取不当，则可能找不到解，或找到的解也不是最优解。因此，需要对估价函数做一些限制，使得算法**确保找到最优解**（步数，即状态转移次数最少的解）。A*算法即为对估价函数做了特定限制，且确保找到最优解的A算法。

A*算法

- $f^*(n) = g^*(n) + h^*(n)$

$f^*(n)$ ：从初始节点S0出发，经过节点n到达目标节点的最小步数（真实值）。

$g^*(n)$ ：从S0出发，到达n的最少步数（真实值）

$h^*(n)$ ：从n出发，到达目标节点的最少步数（真实值）

估价函数 $f(n)$ 则是 $f^*(n)$ 的估计值。

A*算法

- $f(n) = g(n) + h(n)$, 且满足以下限制:

$g(n)$ 是从 s_0 到 n 的真实步数 (未必是最优的), 因此:

$$g(n) > 0 \text{ 且 } g(n) \geq g^*(n)$$

$h(n)$ 是从 n 到目标的估计步数。估计总是过于乐观的, 即

$$h(n) \leq h^*(n)$$

且 $h(n)$ 相容, 则A算法转变为A*算法。A*正确性证明略。

A*算法

$h(n)$ 的相容:

如果 h 函数对任意状态 s_1 和 s_2 还满足:

$$h(s_1) \leq h(s_2) + c(s_1, s_2)$$

$c(s_1, s_2)$ 是 s_1 转移到 s_2 的步数, 则称 h 是相容的。

h 相容能确保随着一步步往前走, f 递增, 这样A*能更高效到最优解。

h 相容 \Rightarrow

$$g(s_1) + h(s_1) \leq g(s_1) + h(s_2) + c(s_1, s_2) = g(s_2) + h(s_2)$$

\Rightarrow

$$f(s_1) \leq f(s_2) \text{ 即 } f \text{ 是递增的。}$$

A*算法

A*算法的搜索效率很大程度上取决于估价函数 $h(n)$ 。一般说来, 满足 $h(n) \leq h^*(n)$ 的前提下, $h(n)$ 的值越大越好。

八数码问题:

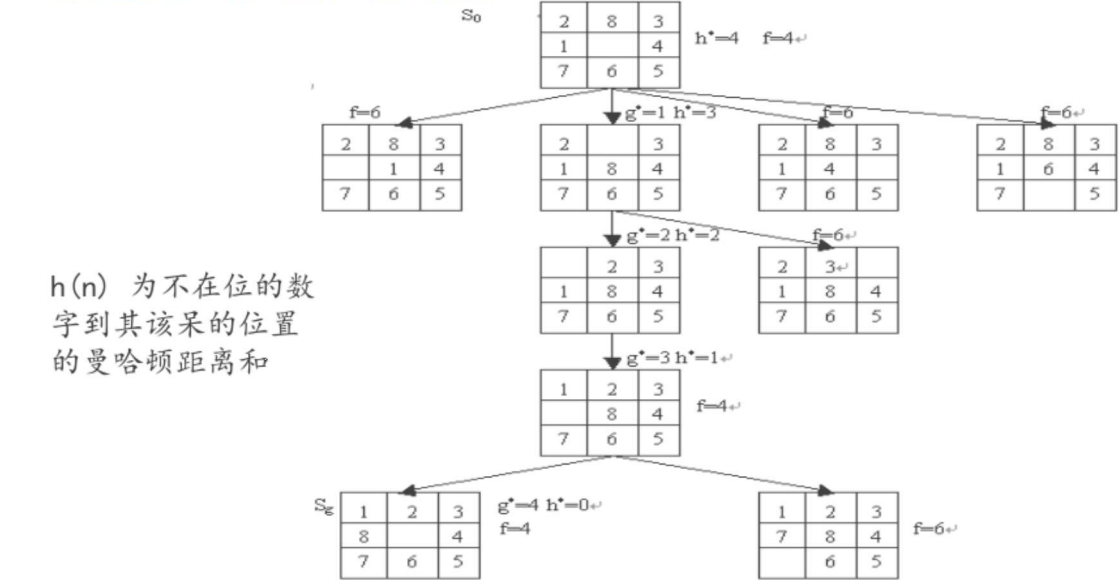
方案一. $h(n)$ 为不在位的数字个数

方案二. $h(n)$ 为不在位的数字到其该呆的位置的曼哈顿距离

后者优于前者

代码采用方案2

A*算法解决八数码问题



代码实现细节

首先定义一个dis[10][10]数组，记录偏移距离，我们把地图中的点按行标则dis[i][j]表示从第i个点到第j个点至少移动的步数。
举个例子：
点的编号： 1 2 3
 4 5 6
 7 8 9
那么显然dis[1][2]=1,dis[2][9]=3,dis[3][4]=3
这个dis数组可通过预处理完成，应该是这个样子：

```
0 1 2 1 2 3 2 3 4
1 0 1 2 1 2 3 2 3
2 1 0 3 2 1 4 3 2
1 2 3 0 1 2 1 2 3
2 1 2 1 0 1 2 1 2
3 2 1 2 1 0 3 2 1
2 3 4 1 2 3 0 1 2
3 2 3 2 1 2 1 0 1
4 3 2 3 2 1 2 1 0
```

接着定义一个goal[10]数组，goal[i]表示数字i在目标状态中的位置
也可以通过预处理完成，
int goal[10]={5,1,2,3,6,9,8,7,4};
然后定义一个now[10]数组，now[i]表示数字i在当前状态中的位置，
这个需要动态维护。
那么估价函数的值当然就是数字0-8 的偏移量之和了。

```
#include<cstdio>
#include<cstring>
#include<iostream>
using namespace std;
const int N=10;
const int dx[]={0,0,1,-1};
const int dy[]={1,-1,0,0};
int flag,now[N],goal[N];
int dis[N][N],a[N][N],mp[N][N];
inline int calcx(int x){return (x-1)/3+1;}
inline int calcy(int x){return x%3?x%3:3;}
inline int abs(int x){return x>0?x:-x;}
inline int h(){int t=0;for(int i=1;i<=9;i++) t+=dis[now[i]][goal[i]];return t;}
inline int check(){for(int i=0;i<9;i++) if(now[i]!=goal[i]) return 0;return 1;}
```



```

void dfs(int depth,int x,int y,int lim){
    if(depth+h()>lim) return ;
    if(check()){flag=1;return ;}
    for(int i=0,nx,ny;i<4;i++){
        nx=x+dx[i];
        ny=y+dy[i];
        if(flag) return ;
        if(nx>0&&nx<=3&&ny>0&&ny<=3){
            swap(a[x][y],a[nx][ny]);swap(now[a[x][y]],now[a[nx][ny]]);
            dfs(depth+1,nx,ny,lim);
            swap(a[x][y],a[nx][ny]);swap(now[a[x][y]],now[a[nx][ny]]);
        }
    }
}

void pre(){
    for(int i=1;i<=9;i++)
        for(int j=i+1;j<=9;j++)
            dis[i][j]=dis[j][i]=calcx(j)-calcx(i)+abs(calcy(j)-calcy(i));
}

int main(){
    pre();
    goal[0]=5;goal[1]=1;goal[2]=2;goal[3]=3;goal[4]=6;goal[5]=9;goal[6]=8;goal[7]=7;goal[8]=4;
    int sx,sy;
    for(int i=1,x,y,z;i<=9;i++){
        scanf("%d",&z);
        x=calcx(i);y=calcy(i);
        mp[x][y]=z;now[z]=i;
        if(!z) sx=x,sy=y;
    }
    for(int i=0;;i++){
        memcpy(a,mp,sizeof mp);
        dfs(0,sx,sy,i);
        if(flag){printf("%d\n",i);break;}
    }
    return 0;
}

```

内心独白：
——将来的我，一定会感谢现在的自己！



版权声明：

本篇随笔版权归作者 [shenben\(www.cnblogs.com/shenben\)](http://www.cnblogs.com/shenben/) 所有，转载请保留

好文要顶

关注我

收藏该文



神犇(shenben)

关注 - 7

粉丝 - 146

+加关注

« 上一篇: codevs 5967 [SDOI2017]相关分析

» 下一篇: POJ2286 The Rotation Game[IDA*迭代加深搜索]