171491208 魏家富

**人工智能实验：四皇后问题**

1. 问题描述：

一个4×4国际象棋盘，依次放入四个皇后，条件：每行、每列及对角线上只允许出现一枚棋子。设：DATA=L（表）    x∈L    x ∈﹛i j﹜    1≤ i, j ≤4  
其中：i j 表示棋子所在行列

2.问题代码：

#include <stdio.h>  
#define N 4  
char board[N][N];  
int t;  
int col[N];                 int safetyPlace(int x,int y)   {  
int i,j;  
for(i=0;i<x;i++)  
{  
j=col[i];  
if(x==i||y==j)  
return 0;

return 0;       
}  
return 1;  
}  
void get\_position(int i)  {  
int w,j;  
char a[1]={3};  
if(i==N)               {  
for (w=0;w<N;w++)  
{  
for (j=0;j<N;j++)  
{  
if(board[w][j]==001)  
printf("%c  ",board[w][j]);  
else  
{  
printf("%c",a[0]);  
printf("%c ",board[w][j]);  
}  
}  
printf("\n");  
}  
printf("\n");  
printf("--------------\n");  
t++;  
}  
else  
{  int u;  
for (u=0;u<N;u++)  
{    if (safetyPlace(i,u)==1)  
{  
col[i]=u;

board[i][u]=001;

get\_position(i+1);

col[i]=0;

board[i][u]=0;      } }  
}  
}  
3.问题分析：

通过对四皇后问题的编程学习，让我对搜索策略更深层次的理解，尤其能比较熟练掌握回溯策略——首先将规则给出一个固定的排序，在搜索时，对当前状态（搜索开始时，当前状态是初始状态）依次检测每一条规则，在当前状态未使用过的规则中找到第一条可应用规则，应用于当前状态，得到的新状态重新设置为当前状态，并重复以上搜索。如果当前状态无规则可用，或者所有规则已经被试探过仍未找到问题的解，则将当前状态的前一个状态（即直接生成该状态的状态）设置为当前状态。重复以上搜索，直到找到问题的解，或者试探了所有可能后仍找不到问题的解为止

4. 分析BFS

（1）完整性：如果有限且目标存在，BFS一定能找到：若目标不存在且为无限大，则BFS不会找到目标。

（2）最优性 ：改进BFS使其得到最优解：找到一个目标结点时，不是立即离开，而记录下目标结点的路径和值，如果有多个目标结点就加以比较，留下较优的结点，把所有可能的路径都搜索完之后，才输出记录的最优路径。

（3）时间复杂度：假设搜索树的状态空间中每个状态都有b个孩子，搜索树的根结点生

第一层的b个结点，每个子结点又生成b个子结点，于是第二层有了C:\Users\dell\AppData\Local\Temp\ksohtml\wpsDFAB.tmp.jpg个结点，以此类推，可以得到1+b+C:\Users\dell\AppData\Local\Temp\ksohtml\wpsDFAC.tmp.jpg+…..+C:\Users\dell\AppData\Local\Temp\ksohtml\wpsDFAD.tmp.jpg=O（C:\Users\dell\AppData\Local\Temp\ksohtml\wpsDFAE.tmp.jpg）个结点总数，其中d为深度，O（C:\Users\dell\AppData\Local\Temp\ksohtml\wpsDFBF.tmp.jpg）也就是BFS的时间复杂度。

（4）空间复杂度：最好的情况下，每层只包含一个结点，故空间复杂度为O（1），最坏情况下将会储存（C:\Users\dell\AppData\Local\Temp\ksohtml\wpsDFC0.tmp.jpg-1）个结点和一个无用的树，这时空间复杂度为O（C:\Users\dell\AppData\Local\Temp\ksohtml\wpsDFC1.tmp.jpg）。