**人工智能第一次作业**

**1.四皇后问题**

四个皇后的所有可能摆放位置，既问题的解空间：16\*15\*14\*13=43680

若考虑行冲突，问题的解空间变为：16\*12\*8\*4=6144

若考虑列冲突，问题的解空间变为：16\*12\*8\*4=6144

若同时考虑行列冲突，问题的解空间变为：16\*9\*4\*1=576

解决代码：

#include <iostream>

#include <cmath>

using namespace std;

//输出所有皇后的坐标

void show(int \*queen, int& sum)

{

int i = 0;

while (true) {

cout << "(" << i << "," << queen[i] << ")";

if (++i != 4) {

cout << " ";

}

else {

cout << endl;

break;

}

}

++sum;

}

//检查当前列能否放置皇后

bool check(int \*queen, int n)

{

for (int i = 0; i != n; ++i) //检查横排和对角线上是否可以放置皇后

{

if (queen[i] == queen[n] || abs(queen[i] - queen[n]) == (n - i)) {

return false;

}

}

return true;

}

//回溯尝试摆放皇后的位置，n 为横坐标

void put(int\* queen, int n,int& sum)

{

for (int i = 0; i != 4; ++i) {

queen[n] = i; //将皇后摆到当前循环到的位置

if (check(queen, n)) {

if (n ==3) {

show(queen, sum); //如果全部摆好，则输出所有皇后的坐标

}

else {

put(queen, n + 1, sum); //否则继续摆放下一个皇后

}

}

}

}

int main()

{

cout << "--------------------------------------------------" << endl;

//定义棋盘数组。因为只可能每行、每列都是一个棋子，所以没必要使用二维数组，

//使用一维数组的索引和值来等价一个二维数组

int \*queen = new int[4];

for (int ix = 0; ix != 4; ++ix) queen[ix] = 0;

//记录满足要求的排列的个数

int sum = 0;

put(queen, 0, sum); //从横坐标为 0 开始依次尝试

delete[] queen;

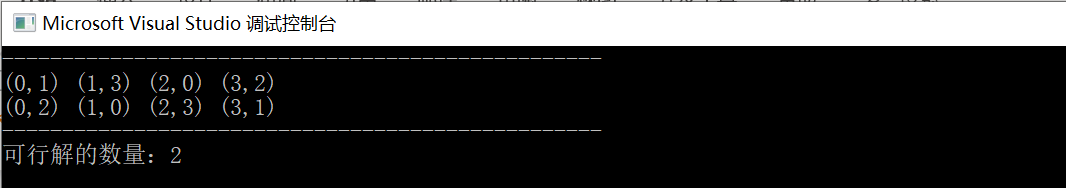
cout << "--------------------------------------------------" << endl;

cout << "可行解的数量：" << sum << endl;

return 0;

}

结果：



**2.分析深度优先搜索的四个特性**

完整性：完整性是指能否确定问题是否有解，并且能够至少输出一个解。

对于深度优先搜索来说，当树的深度是确定的时候，那么是完整的，但是如果深度是无限的那么就不是完整的，算法会陷入循环之中。

最优性：最优性是指算法能够找到一个所有解中的最优的一个，对于深度优先搜索来说，它不具备最优性，因为它的搜索方向是最开始人工指定的，只有把某一分支遍历到最深的一层，才会搜索另一个分支，在搜索到一个解之后就会停止搜索。

时间复杂度：时间复杂度是指算法的所需操作步骤数量的估计值。深度优先遍历的时间复杂度为，其中b为目标的深度，m为树的深度。

空间复杂的：空间复杂度是指算法在执行操作的时候需要储存数据的大小估计。深度优先遍历的时间复杂度是，其中b为目标的深度，m为树的深度。