



**实验报告**

**题目：作业一**

**学院：信息科学与技术学院**

**专业：计算机技术**

**年级：2018级研究生**

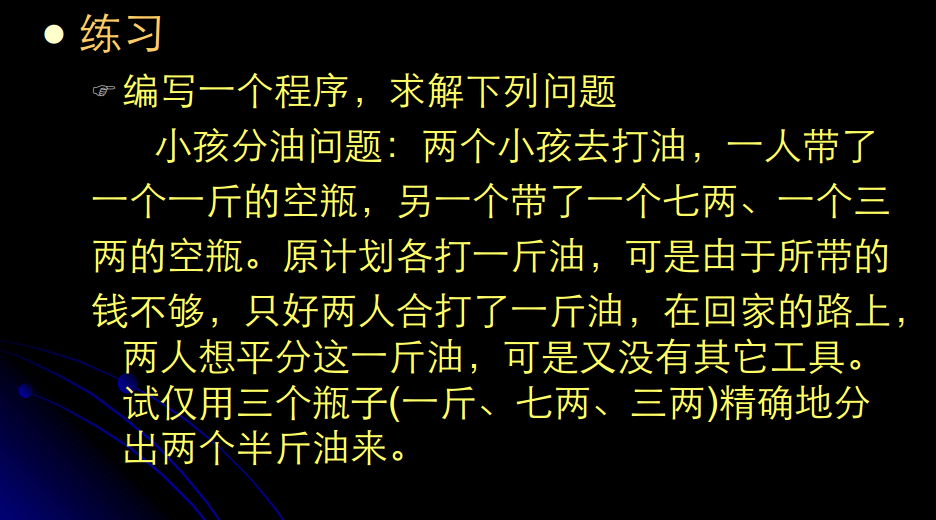
**课程：计算智能\_作业1**

**姓名：王晨**

**学号：23020181154239**

**备注：**

**1.问题描述**



**2.算法设计**

油瓶中有的变化规则：S，R分别表示七两，三两的油瓶。

|  |  |  |
| --- | --- | --- |
| 规则号 | 规则 | 规则解释 |
| 1 | (S,R) and S<7 -> (7,R) | 七两油瓶不满时装满 |
| 2 | (S,R) and R<3 -> (S,3) | 三两油瓶不满时装满 |
| 3 | (S,R) and S>0 -> (0,R) | 七两油瓶不空时倒空 |
| 4 | (S,R) and R>0 -> (S,0) | 三两油瓶不空时倒空 |
| 5 | (S,R) and S>0 and S+R<=3 -> (0,S+R) | 七两瓶中的油全部倒入三两瓶中 |
| 6 | (S,R) and R>0 and S+R<=7 -> (S+R,0) | 三两瓶中的油全部倒入七两瓶中 |
| 7 | (S,R) and S<7 and S+R>=7 -> (7,S+R-7) | 用三两瓶中的油装满七两瓶 |
| 8 | (S,R) and R<3 and S+R>=3 -> (S+R-3,3) | 用七两瓶中的油装满三两瓶 |

**3.数据结构**

Oil类具有(x,y,z)属性，存放（一斤、七两、三两）瓶里的油。

Node类，具有Oil和n属性，存放三个瓶子里的油状态，以及瓶子存放油的父节点在数组中的位置。

队列oilQueue存放分油问题的广度优先搜索树中的节点。

数组arr存放分油问题的广度优先搜索树中不重复的节点。

数组result存放分油问题的结果。

问题的起始状态<10,0,0>,问题的目标状态<5,5,0>

**4.核心伪代码**

void BFS(Node\* r)

{

queue<Node \*>oilQueue;

Oil \*On;

int cnt = 0;

int t;

int res = 0;

Node arr[50];

oilQueue.push(r);

while (!oilQueue.empty())

{

On = oilQueue.front()->o;

if (!isVisited(oilQueue.front(), cnt, arr))

{

arr[cnt].o = oilQueue.front()->o;

arr[cnt].n = oilQueue.front()->n;

cnt++;

t = cnt - 1;

}

oilQueue.pop();

if (On->y < 7)

{

Oil \*newOil = new Oil();

Node \*newNode = new Node();

newOil->y = 7;

newOil->x = 10 - 7 - On->z;

newOil->z = On->z;

newNode->o = newOil;

newNode->n = t;

if (!isVisited(newNode, cnt, arr))

oilQueue.push(newNode);

}

if (On->z < 3)

{

Oil \*newOil = new Oil();

Node \*newNode = new Node();

newOil->z = 3;

newOil->x = 10 - 3 - On->y;

newOil->y = On->y;

newNode->o = newOil;

newNode->n = t;

if (!isVisited(newNode, cnt, arr))

oilQueue.push(newNode);

}

if (On->y>0)

{

Oil \*newOil = new Oil();

Node \*newNode = new Node();

newOil->y = 0;

newOil->x = 10 - On->z;

newOil->z = On->z;

newNode->o = newOil;

newNode->n = t;

if (!isVisited(newNode, cnt, arr))

oilQueue.push(newNode);

}

if (On->z >0)

{

Oil \*newOil = new Oil();

Node \*newNode = new Node();

newOil->z = 0;

newOil->x = 10 - On->y;

newOil->y = On->y;

newNode->o = newOil;

newNode->n = t;

if (!isVisited(newNode, cnt, arr))

oilQueue.push(newNode);

}

if (On->y>0 && On->y+On->z <=3)

{

Oil \*newOil = new Oil();

Node \*newNode = new Node();

newOil->y = 0;

newOil->z = On->y + On->z;

newOil->x = 10 - On->y - On->z;

newNode->o = newOil;

newNode->n = t;

if (!isVisited(newNode, cnt, arr))

oilQueue.push(newNode);

}

if (On->z>0 && On->y + On->z <= 7)

{

Oil \*newOil = new Oil();

Node \*newNode = new Node();

newOil->z = 0;

newOil->y = On->y + On->z;

newOil->x = 10 - On->y - On->z;

newNode->o = newOil;

newNode->n = t;

if (!isVisited(newNode, cnt, arr))

oilQueue.push(newNode);

}

if (On->y<7 && On->y + On->z >=7)

{

Oil \*newOil = new Oil();

Node \*newNode = new Node();

newOil->y = 7;

newOil->z = On->y + On->z - 7;

newOil->x = 10 - On->y - On->z;

newNode->o = newOil;

newNode->n = t;

if (!isVisited(newNode, cnt, arr))

oilQueue.push(newNode);

}

if (On->z<3 && On->y + On->z >= 3)

{

Oil \*newOil = new Oil();

Node \*newNode = new Node();

newOil->z = 3;

newOil->y = On->y + On->z - 3;

newOil->x = 10 - On->y - On->z;

newNode->o = newOil;

newNode->n = t;

if (!isVisited(newNode, cnt, arr))

oilQueue.push(newNode);

}

}

for (int i = 0; i < cnt; i++)

{

if (arr[i].o->x == 5)

{

res = i;

break;

}

}

Node result[20];

int a = 0;

while (arr[res].n != -1)

{

result[a] = arr[res];

res = arr[res].n;

a++;

}

if (arr[res].n == -1)

{

result[a] = arr[res];

res = arr[res].n;

}

for (int i = a; i >= 0; i--)

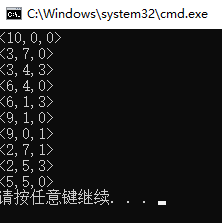
{

cout << "<" << result[i].o->x << "," << result[i].o->y << "," << result[i].o->z << ">" << endl;

}

}

**5.代码运行及测试**



**6.结论**

A.本实验采用的是盲目搜索算法中的广度优先搜索，因为是盲目搜索，使得实验的空间复杂度较大，但是采用重复的节点不再存储，使得在一定程度上降低了实验的空间复杂度。

B.因为实验只有起始状态以及终止状态，使得实验过程中在存储节点的时候都必须检测是否达到终止状态。对于队列中每个节点的存储也需要列出所有规则进行条件判断。

C.这个实验只是针对油瓶是（一斤、七两、三两）的问题进行解决，灵活度并不高，假设油瓶容量为其它情况，需要重新再次编写代码。