数据结构与算法导论

实验报告

实验题目：实验三 图 多叉路口交通灯的问题

姓 名：\_\_\_\_\_\_\_\_\_\_\_\_孙浩峰\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

学 号：\_\_\_\_\_\_\_\_\_\_\_2019210588\_\_\_\_\_\_\_\_\_\_\_\_\_\_

日 期：\_\_\_\_\_\_\_\_\_\_\_\_2020.5.31\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

自我评分：\_\_\_\_\_\_\_\_\_【 A+ 】\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

自我评分说明：A+，A，B+，B，B-，C，D，分别对应分数95、90、85、80、75、70、60

诚信声明

本人郑重承诺：本实验程序和实验报告均是本人独立学习和工作所获得的成果。尽我所知，实验报告中除特别标注的地方外，不包含其他同学已经发表或撰写过的成果；实验程序中对代码工作的任何帮助者所作的贡献均做了明确的说明，并表达了谢意。

如有抄袭，本人原因承担因此而造成的任何后果。

特此声明。

签名：\_\_\_\_\_孙浩峰 \_\_\_\_\_\_\_

日期：\_\_\_2020.5.31\_\_\_\_\_\_\_\_\_

程序引用说明

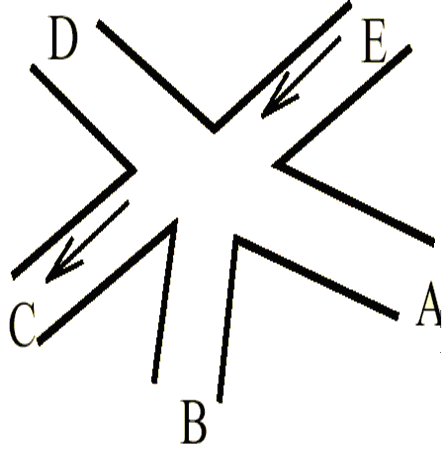
|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 引用项 | 来源 | 雷同代码行数 |
| 1 | 判别函数 | 课本 | 3 |
| 2 |  |  |  |
| 3 |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| 小计 | | |  |

总代码行数\_\_\_\_\_\_\_\_\_\_100\_\_\_\_\_\_\_\_\_\_; 引用占比\_\_\_3%\_\_\_\_\_\_\_\_\_\_\_\_\_

1、实验简介

问题：多叉路口交通灯的问题。

假如一个如下图所示的五叉路口，其中C、E是箭头所示的单行道，如何设置路口的交通灯，使得车辆之间既不相互碰撞，又使交通流量最大？

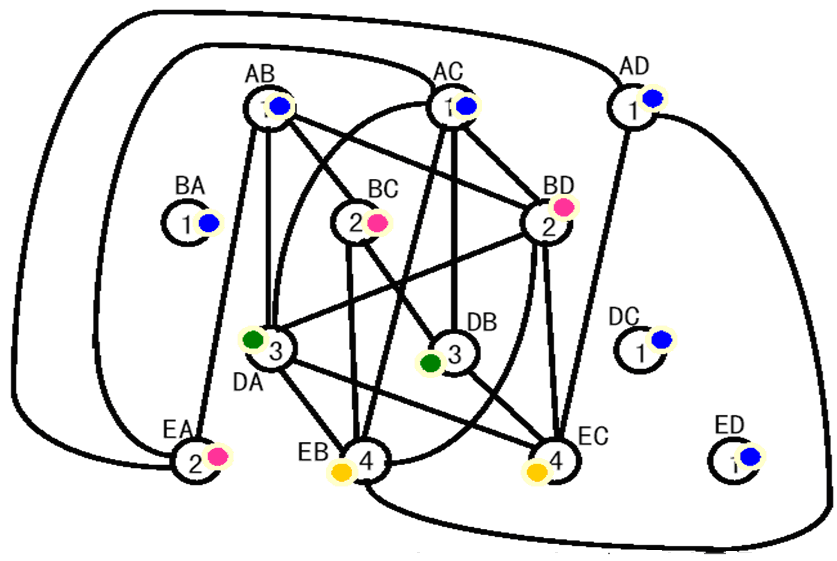


在这副图中有AB,AC,AD,BA,BC,BD,DA,DB,DC,EA,EB,EC,ED十三种行驶路线，例如以A为入口有D，C，B 三种出口，B，D也是三种，E有四种 共4+3\*3=13种

第一个路线AB，当路线AB是通路时，可以发现路线BC、BD、DA、EA均与AB有交叉，因此将AB的圆圈与其他四个圆圈用线连接

后面的依次类推

于是问题就可以转换成染色问题进行求解



2、程序框架

TU::TU(int b[][N], int m)//构造函数 1. 定义图中的顶点 2. //定义图定点连接 3. //按照顶点度的大小给顶点排序，为的是选择度大的顶点

bool OK(int a[N][N], int k, int x[])//判断当前顶点的颜色是否可用

void Co(int b[N][N], int x[], int n, Node No[])//染色函数

int main()//主函数

3、关键代码实现

3.1 函数名1 判断函数

bool OK(int a[N][N], int k, int x[])//判断当前顶点的颜色是否可用

{

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

if ((a[k][i]) && (x[k] == x[i]))

return false;

return true;

}

通过遍历所有结点来确定改结点的颜色与其相连的结点的颜色是否相同，相同则使用下一个颜色

3.2 函数名2 染色函数

void Co(int b[N][N], int x[], int n, Node No[])//染色函数

{

x[No[0].col] = 1;

for (int k = 0; k < N; k++)

{

x[No[k].col] = 1;

while (!OK(b, No[k].col, x))

x[No[k].col] = x[No[k].col] + 1;

}

}

对结点进行染色，调用判断函数进行颜色的辨别

3.3 函数名 图的构造函数

TU::TU(int b[][N], int m)//构造函数

{

n = m;//赋值顶点个数

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

for (int j = 0; j < N; j++)

a[i][j] = b[i][j];//a[][]=b[][]结点连接

char l[N] = { 'a','b','c','d','e','f','g','h','i','j','k','l','m' };

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

{

S[i].name = l[i];//S[i].name=l[i]道路的名字

S[i].col = i;

S[i].url = 0;//度赋初值

for (int j = 0; j < N; j++)

S[i].url = S[i].url + b[i][j];//计算顶点的度

}

//按照顶点度的大小给顶点排序，为的是选择度大的顶点，经过查询资料显示，这种方法在大部分情况下可以得到最优解

for (int i = 1; i < N; i++)//起泡排序

for (int j = 1; j < N - i; j++)

if (S[j].url < S[j + 1].url)

{

Node tmp = S[j];

S[j] = S[j + 1];

S[j + 1] = tmp;

}

}

对于染色函数，可以得到一种有效的染色方案，但是我们并不知道这种方案是否是最有效的，即是否使用了最少种类的颜色，因此需要给出相应的解决方法，如果在构造图的过程中我们就按照结点度的大小进行从大到小的排序，然后进行度从大到小的染色，大量的实践证明，这种方法在大部分的情况下可以得到一个最优解

3.4

4、不足

尽管在构造图的过程中我们就按照结点度的大小进行从大到小的排序，然后进行度从大到小的染色，大量的实践证明，这种方法在大部分的情况下可以得到一个最优解，但是也会有一定的意外情况出现的结果不是最优解，

5、心得体会

图这一部分，内容相对较难，理论知识相对较难，但是代码实现与前几部分相差不大，在本次实验中，练习了图的构造，路口交通灯设置需要思考的问题，问题迁移的模式-转换成地图染色问题。