**云南大学数学系《算法图论实验》课程上机实验报告**

|  |  |  |
| --- | --- | --- |
| **课程名称**：算法图论实验 | **学期：**2016-2017学年第一学期 | **成绩**： |
| **指导教师**：李建平 | **学生姓名**：金洋 | **学生学号**：20131910023 |
| **实验名称**： **染色** | | |
| **实验编号**：No.6 | **实验日期**：**2016.12.15** | **实验学时**： **1** |
| **学院：** 数学与统计学院 | **专业： 信息与计算科学** | **年级**： **2013** |

**一、实验目的**

1、使用c语言实现二部图的边染色；

2、使用c语言实现点染色，使得染色数最多△+1；

**二、实验内容**

1、给定二部图G=(S,T;E)，求其边染色方案；

2、给定任意图G=(V,E),求其一种点染色方案，使得最多需要△(G)+1种颜色；

**三、使用环境**

平台：Microsoft Visual C++ 6.0

语言：C语言

1. **算法介绍**

1、二部图的边染色

Algorithm 二部图边染色

Input 二部图G=(S,T;E);

Output G的边染色方案;

Begin

Step 1: 令，i=1；

Step 2: repeat

1. 找中的最大匹配；
2. 令；
3. 将中的边染为同一色；
4. i=i+1;

until 是空图；

Step 3:输出每条边的染色号；

End.

1. 点染色

Algorithm 点染色

Input G=(V,E);

Output G的点染色方案（至多用△(G)+1种颜色）;

Begin

Step 1: 令，在中找最小度点，记为；

令，在中找最小度点，记为；

…

令，在中找最小度点，记为；（为 空图）；

Step 2: For i=n downto 1 do

If 在中,与,…,中有边相连 then

使用一种与,…,都不相同的新色染色；

Else 使用一种与其邻点都不相同的旧色染色；

Step 3:输出每点的染色号；

End.

**五、调试过程**

1．程序代码

①二部图的边染色

**EdgeColoring.c**

#include <stdio.h>

#include <string.h>

#include <windows.h>

#define MAXN 999999999

#define MAXV 100

int E;

void input(int S[MAXV],int T[MAXV],int C[MAXV][MAXV]) {

int i,a,b;

//freopen("EdgeColoring.in", "r", stdin);

printf("请输入S集合的顶点个数|S|=");

scanf("%d",&S[0]);

printf("请输入S集合的顶点:");

for (i=1;i<=S[0];i++) scanf("%d",&S[i]);

printf("请输入T集合的顶点个数|T|=");

scanf("%d",&T[0]);

printf("请输入T集合的顶点:");

for (i=1;i<=T[0];i++) scanf("%d",&T[i]);

printf("请输入二部图G的边数:\n");

scanf("%d",&E);

printf("请依次输入二部图G的%d条边:\n",E);

for (i=1;i<=E;i++) {

scanf("%d %d",&a,&b);

C[a][b]=1;

}

}

void buildN(int n,int S[MAXV],int T[MAXV],int C[MAXV][MAXV]) {

int i;

for (i=1;i<=S[0];i++)

C[n+1][S[i]]=1;//点n+1为起点，起点与S集合都相连，容量为1;

for (i=1;i<=T[0];i++)

C[T[i]][n+2]=1;//点n+2为终点，T集合与终点都相连，容量为1;

}

/\*构造剩余网络\*/

void buildNf(int n,int C[MAXV][MAXV],int Cf[MAXV][MAXV],int F[MAXV][MAXV]) {

int i,j;

for (i=1;i<=n;i++)

for (j=1;j<=n;j++)

if (C[i][j]) {

Cf[i][j]=C[i][j]-F[i][j];

Cf[j][i]=F[i][j];

}

for (i=1;i<=n;i++)

for (j=1;j<=n;j++)

if (Cf[i][j]==0)

Cf[i][j]=MAXN;//去掉Nf中容量值为0的所有弧

}

/\*最短路径初始化\*/

void shortestPathInitial(int n,int VS,int X[MAXV],int LAM[MAXV],int L[MAXV]) {

int i;

for (i=1;i<=n;i++) {

X[i]=0;

LAM[i]=MAXN;

}

LAM[VS]=0;

X[VS]=1;

X[0]=1;//X[0]记录X集合中顶点个数,初始时Vs进入X集合

for (i=1;i<=n;i++) L[i]=0;

}

/\*φ(X)是否为空? 是:返回0;否:返回1\*/

int PHi(int M[MAXV][MAXV],int X[MAXV],int n) {

int i,j;

for(i=1;i<=n;i++)

for (j=1;j<=n;j++)

if ((i!=j)&&(X[i])&&(X[j]==0)&&(M[i][j]<MAXN)) return(1);

return(0);

}

/\*最短路\*/

void prim(int M[MAXV][MAXV],int X[MAXV],int LAM[MAXV],int L[MAXV],int n) {

int i,j,min;

int x,y;

min=MAXN;

for(i=1;i<=n;i++)

if (X[i])

for(j=1;j<=n;j++)

if (!X[j] && i!=j && LAM[i]+M[i][j]<min ) {

min=LAM[i]+M[i][j];

x=i;

y=j;

}

if (min!=MAXN) {

X[y]=1;

X[0]++;

L[y]=x;

LAM[y]=min;

}

}

/\*改变N中的流量\*/

void changeFlow(int VT,int L[MAXV],int Cf[MAXV][MAXV],int F[MAXV][MAXV],int C[MAXV][MAXV]) {

int theta=MAXN;

int y=VT;

while (L[y]!=0) {

if (theta>Cf[L[y]][y] ) theta=Cf[L[y]][y];

if (theta>Cf[y][L[y]] ) theta=Cf[y][L[y]];

y=L[y];

}

y=VT;

while (L[y]!=0) {

if (F[L[y]][y]<C[L[y]][y]) F[L[y]][y]+=theta;

if (F[y][L[y]]>0) F[y][L[y]]-=theta;

y=L[y];

}

}

void main() {

int n;//顶点数

int S[MAXV],T[MAXV];

int C[MAXV][MAXV],F[MAXV][MAXV];//C,F分别记录N的容量、流量

int Cf[MAXV][MAXV];//剩余网络权值

int LAM[MAXV],L[MAXV],X[MAXV];

int VS,VT,m;

int color=0;

int i,j;

memset(C, 0, sizeof(C));//容量初始为全0

input(S,T,C);

n=S[0]+T[0];//原图顶点个数

buildN(n,S,T,C);

VS=n+1;//起点

VT=n+2;//终点

n=n+2;

m=0;

printf("二部图G的染色方案为:\n");

while (m<E) {

memset(F, 0, sizeof(F));//流量初始为全0

memset(Cf, 0, sizeof(Cf));

while (1) {

buildNf(n,C,Cf,F);

shortestPathInitial(n,VS,X,LAM,L);//最短路径初始化

while (!X[VT] && PHi(Cf,X,n)) prim(Cf,X,LAM,L,n);

if (X[VT]) changeFlow(VT,L,Cf,F,C);

else break;

//printf("Jin");

}

color++;

for (i=1;i<=S[0];i++)

for (j=1;j<=T[0];j++)

if (F[S[i]][T[j]]>0) {

printf("v%d v%d:%d\n",S[i],T[j],color);

C[S[i]][T[j]]=0;//已经染色的容量设为0，下次求匹配时不再选择

m++;

}

}

printf("G的染色数=%d",color);

getch();

}

②点染色

#include <stdio.h>

#define MAXV 100

void sort(int d[2][MAXV],int n) {

int i,j,t;

for (i=1;i<=n-1;i++)

for (j=i+1;j<=n;j++)

if (d[1][i]>d[1][j]) {

t=d[0][i];d[0][i]=d[0][j];d[0][j]=t;

t=d[1][i];d[1][i]=d[1][j];d[1][j]=t;

}

}

void main() {

int i,j,n;

int d[3][MAXV];//d[0]保存节点号，d[1]保存度

int M[MAXV][MAXV];//邻接矩阵

int color[MAXV];//记录已经使用过的颜色

int result[MAXV];//记录已经使用过的颜色

memset(d, 0, sizeof(d));

//freopen("VertexColoring.in", "r", stdin);

printf("请输入连通图G的顶点个数n=");

scanf("%d",&n);

printf("请输入连通图G的邻接矩阵:\n");

for (i=1;i<=n;i++) {

d[0][i]=i;

for (j=1;j<=n;j++) {

scanf("%d",&M[i][j]);

d[1][i]+=M[i][j];//同时计算度

}

}

sort(d,n);

result[d[0][n]]=1;//最后一点颜色为1号

for (i=n-1;i>=1;i--) {

memset(color, 0, sizeof(color));

for (j=n;j>=i+1;j--)

if (M[d[0][i]][d[0][j]]) color[result[d[0][j]]]++;

for (j=1;j<=n-1;j++)

if (color[j]==0) break;

result[d[0][i]]=j;

}

printf("染色方案为:\n");

for (i=1;i<=n;i++)

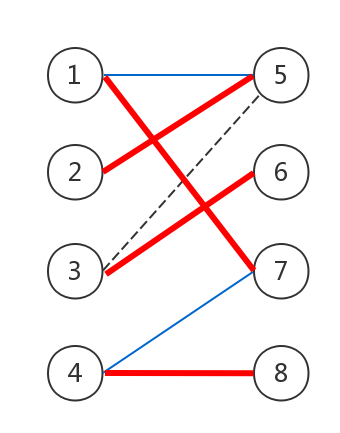
printf("v%d:%d\n",i,result[i]);

getchar();

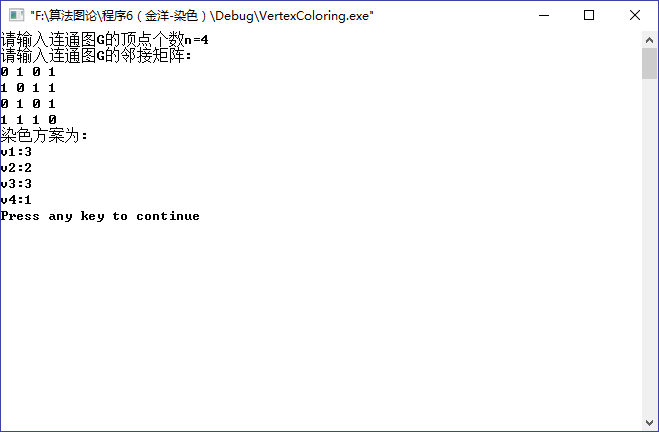
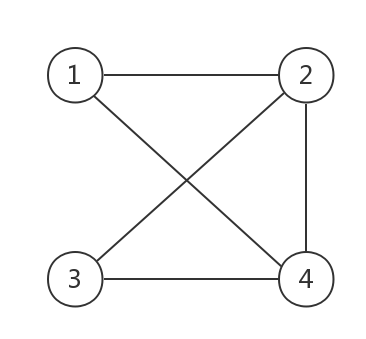
}

1. 运行窗口

①二部图的边染色



②点染色



**六、总结**

1、学会使用c语言实现二部图的边染色；

2、学会使用c语言实现点染色，使得染色数最多△+1；

**七、参考文献**

[1] 谭浩强，c程序设计（第三版），清华大学出版社，2005年7月；

[2] 《运筹学》教材编写组，运筹学（第四版），清华大学出版社，2012年9月；

**八、教师评语**