**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**： **最短路的插点问题** | | |
| **实验编号**：No.3 | **实验日期**：**2016.10.8** | **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**

使用c语言解决最短路的插点问题；

**二、实验内容**

给定一个连通赋权图G=(V,E;w,c;s,t)及正整数d，这里w,c: ；s和t是两个固定的顶点，在图G中寻找一条从s到t的最短路，并向最短路的一些特殊边上插入一些顶点，不妨设得到的新路为（称为P的加细路），使得最短路中任意边的权重都不超过常数d，目标是在满足上述条件情况下，求出插入顶点后所产生的费用达到最小。这里用insert(e)表示插入到边e中新点的个数，于是插点费用；

**三、使用环境**

平台：Microsoft Visual C++ 6.0

语言：C语言

**四、算法介绍**

Algorithm 最短路的插点问题

Input 连通赋权图G=(V,E;w,c;s,t)及正整数d;

Output 最短路，每条边上插点数目，；

Begin

Step 1: 由修正的反圈法求出G中从s到t的所有最短路构成的子图并构 造辅助有向图D；

Step 2: 对于D中的每条弧e，构造新的权重，其中 ；

Step 3: 在D中再利用反圈算法求出s到t的最短路;

Step 4: 输出最短路，每条边上插点数目，;

End.

**五、调试过程**

1．程序代码

**HSCSP.c**

#include <stdio.h>

#include <stdlib.h>

#define MAXN 999999999

#define MAXV 100

int w[MAXV][MAXV],c[MAXV][MAXV],n,VS,VT,d;

int X[MAXV],n,LAM[MAXV],VS,L[MAXV][MAXV],wD[MAXV][MAXV],insert[MAXV][MAXV],LD[MAXV],SP[MAXV];

//L[MAXV][MAXV]在G中记录，L[i][j]表示边(i,j)有可能在最终形成的最短路上

//LD[MAXV]在D中记录，LD[i]表示i的前驱节点

//wD[MAXV][MAXV]为D的边权值

void input() {

int i,j;

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

printf("请输入G的顶点个数n=");

scanf("%d",&n);

printf("请输入G的w矩阵:\n");

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

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

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

if (w[i][j]==0) w[i][j]=MAXN;//没有边相连，将权值设为足够大

}

printf("请输入G的c矩阵:\n");

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

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

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

printf("起点Vs=");

scanf("%d",&VS);

printf("终点Vt=");

scanf("%d",&VT);

printf("d=");

scanf("%d",&d);

}

void shortestPathInitial() {

int i;

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

X[i]=0;

LAM[i]=MAXN;

}

X[VS]=1;

LAM[VS]=0;

}

/\*φ(X)是否为空? 是:返回0;否:返回1\*/

int PHi(int M[MAXV][MAXV]) {

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);

}

void primAllSP(int M[MAXV][MAXV]) {

int i,j,min;

int now[MAXV];//表示某顶点是否在这一轮反圈中选边时被加入X集合;

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

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];

if (min!=MAXN) {

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

if (X[i])

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

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

LAM[j]=min;

X[j]=1;

now[j]=1;

//X[0]++;

L[j][i]=1;//表示边(i,j)有可能在最终形成的最短路上

}

}

}

int buildD(int v) {

int i;

if (v==VS) return 0;

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

if (L[v][i]) {

buildD(i);

insert[i][v]=(w[i][v]+d-1)/d-1;//对正整数而言,a/b向上取整只需 (a+b-1)/b

wD[i][v]=insert[i][v]\*c[i][v];

}

return 0;

}

void prim(int M[MAXV][MAXV]) {

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]++;

LD[y]=x;

LAM[y]=min;

}

}

int getSP(int v) {

int i;

if (v==VS) {

SP[0]=1;

SP[1]=VS;

return 0;

}

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

if (LD[v]==i) {

getSP(i);

SP[0]++;

SP[SP[0]]=v;

return 0;

}

}

void main() {

int i,j,total;

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

input();

shortestPathInitial();

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

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

L[i][j]=0;

while (!X[VT] && PHi(w)) primAllSP(w);

buildD(VT);

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

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

if (wD[i][j]==0) wD[i][j]=MAXN;

shortestPathInitial();

while (!X[VT] && PHi(wD)) prim(wD);

getSP(VT);

printf("最短路为:");

for (i=1;i<=SP[0];i++) printf("%d ",SP[i]);

printf("\n最短路的每条边上的插点数目为:\n");

total=0;

for (i=1;i<=SP[0]-1;i++) {

printf("%d %d:%d\n",SP[i],SP[i+1],insert[SP[i]][SP[i+1]]);

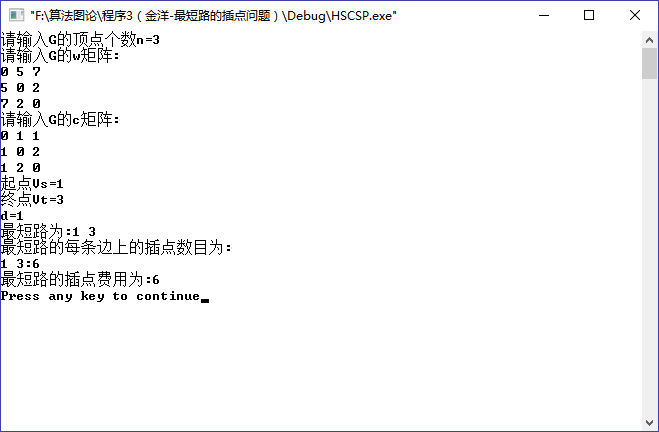
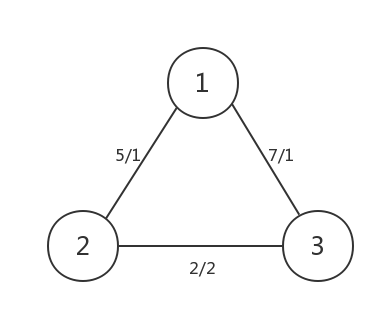
total+=wD[SP[i]][SP[i+1]];

}

printf("最短路的插点费用为:%d\n",total);

}

1. 运行窗口



**六、总结**

1.学会使用c语言解决最短路的插点问题；

2.路的插点问题是NP-完备的，而最短路的插点问题可以用多项式算法解决；

**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**