云南大学数学与统计学院
《算法图论实验》上机实践报告

课程名称：算法图论实验	**年级：**2015 级	**上机实践成绩：**
指导教师：李建平	**姓名：**	**专业：**
上机实践名称：有向图的弧连通度	**学号：**20151910042	**上机实践日期：**2018-11-21
上机实践编号：6	**组号：**	

一、 实验目的

1. 了解图的点连通度和弧连通度的定义；
2. 了解 Menger 定理的描述以及证明。

二、 实验内容

1. 写出求图的弧连通度的算法；
2. 用 C 语言实现上述算法。

三、 实验平台

Windows 10 Pro 1803；

MacOS Mojave。

四、 算法设计

4.1 预备知识

为了刻画图的连通程度，引入图的连通度的概念。设连通图G不是完全图，如果$V_1 \subset V$，使得$G-V_1$是不连通的，则称V_1是图G的**点截集**（vertex-cut）。如果$|V_1|=k$，也称之为k-点截集。给定图G的两个顶点u和v，假如点截集V_1使得顶点u和v彼此不可到达，那么称点截集V_1是图G中**分离**u,v**的点截集**（vertex-cut separating u and v）。

定义图G的点连通度（vertex-connectivity），记为$\kappa(G)$，

$$\kappa(G)\begin{cases}\mathbf{min}\{|V_1||V_1 \text{ is } \mathbf{vertex\ cut}\}, & \text{if } G\neq K_n\\ n-1, & \text{if } G=K_n\end{cases}$$

对于非负整数k，若$\kappa(G)\geq k$，则称G是k-点连通图（k-vertex-connected Graph），简称k-连通图。

类似地可以定义边截集。给定连通图G，如果$E_1 \subset E$，使得$G-E_1$是不连通的，则称E_1是图G的**边截集**（edge-cut）。如果$|E_1|=k$，也称之为k-边截集。类似地定义**分离**u,v**的边截集**（edge-cut separating u and v）。定义图G的边连通度（edge-connectivity），记为$\lambda(G)$，

$$\lambda(G)\begin{cases}\mathbf{min}\{|E_1||E_1 \text{ is } \mathbf{edge\ cut}\}, & \text{if } G\neq K_1\\ 0, & \text{if } G=K_1\end{cases}$$

设$G=(V,\ E)$是一个图，X和Y是V的任意两个非空真子集。给定一个链，如果它的两个端点分别属于X和Y，而且链上的其他点都不属于$X\cup Y$，则称这个链是$(X,\ Y)$-**链**（$(X,\ Y)$-chain）。特别地，若$v\in X\cap Y$，则v本身就是一个$(X,\ Y)$-链。若$Z\subset V$，且任何一个$(X,\ Y)$-链都与Z相交，则称Z是一个$(X,\ Y)$-**分离**

集（$(X,\ Y)$-separator），X与Y也是$(X,\ Y)$-分离集。

Menger 定理 X和Y是V的任意两个子集，则G中存在$k\ (k \in \mathbb{Z}^+)$个点不交的$(X,\ Y)$-链，当且仅当每个$(X,\ Y)$-分离集至少包含k个点。

Menger 推论 1 设$x,\ y \in V(G)$，$xy \notin E$，则G中存在$k\ (k \in \mathbb{Z}^+)$个点不交的$(x,\ y)$-链，当且仅当G中的每个分离$x,\ y$的点截集至少包含k个点。

Menger 推论 2 设$x,\ y \in V(G)$，$xy \notin E$，则G中存在$k\ (k \in \mathbb{Z}^+)$个边不交的$(x,\ y)$-链，当且仅当G中的每个分离$x,\ y$的边截集至少包含k个边。

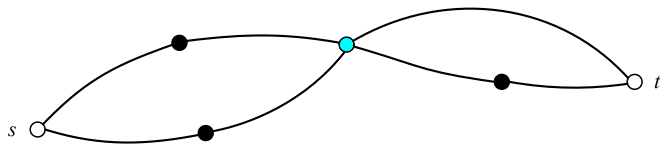
（**该推论的证明比较有启发性**，可以利用图G的线图$L(G)$来证明，线图必然满足 Menger 定理，所以根据这个观察结果，直接证明本推论。）

关于点的 Menger 型极大极小定理 设$x,\ y$是G的两个不相邻的点，则内点不交的$(x,\ y)$-链的最大个数等于局部点连通度$\kappa(x,\ y)$。

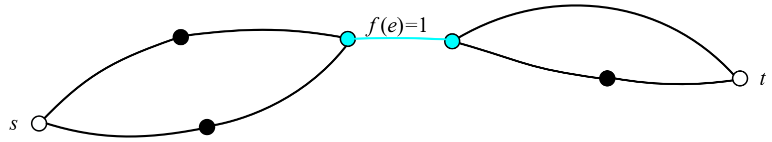
关于边的 Menger 型极大极小定理 设$x,\ y$是G的两个不相邻的点，则内边不交的$(x,\ y)$-链的最大个数等于局部点边通度$\lambda(x,\ y)$。

4.2 算法解析

有了 Menger 定理的铺垫，就可以很顺利地写出求图的点连通度的算法，其核心思想是网络流算法。因为有点不交这个限制条件，所以不能单纯地讲所有边的容量设置为 1。这直接导致了网络流算法中出现了节点的流量限制，但是 Edmonds-Karp 算法中并没有对于节点进行限制，所以需要对原图进行一定的变换。$(s,\ t)$-链集合如图(*a*)所示，对于其中的蓝色节点v来说，$d_G(v)$一定是偶数，可以将之拆分为两个节点，中间加一个边，且该边的容量设置为1，这样就可以解除节点容量限制，直接在原图中进行网络最大流算法。(*a*)图是原先的，(*b*)图是经过变换之后的图。这样就可以解决点连通度的求解问题。对于边连通度问题而言要简单一点，直接在原图上采取网络流算法即可，通过遍历所有的点对，找出最小值即可。



(*a*) 原图



(*b*) 变换后的图

4.3 求有向图的弧连通度的算法

假设已经有 Edmonds-Karp 算法，可以用来求解有向图D的顶点s与t之间的最大网络流f，且记这个形式为$f = \mathrm{MAXFLOW}(D;\ s,\ t)$，这个算法是下面这个算法的子算法。

Algorithm	求有图的弧连通度，记此算法为AC
Input	连通的有向图$D = (V,\ A)$

Output 图D的一个弧连通度C，记$C = \mathrm{AC}(D)$

Begin

Step 1 // 初始化一个空的列表

$L = \emptyset$

Step 2 // 初始化弧的容量

for each edge $e \in A$:

$e.\mathbf{capacity} = 1$

Step 3 // 寻找任意有序点对之间的最大流量

for each vertex $v_1 \in A$:

for each vertex $v_2 \in A$ **and** $v_2 \neq v_1$:

$L.\mathrm{append}(\mathrm{MAXFLOW}(D;\ v_1,\ v_2))$

Step 4 $\mathrm{C} = \mathrm{MIN}(L)$

output C

End

五、 程序代码

六、 参考文献

[1] **林锐**. 高质量 C++/C 编程指南 [M]. 1.0 ed., 2001.

[2]