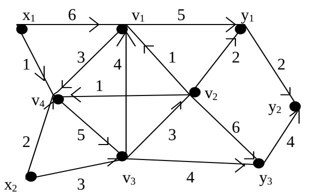
第九章 网络

9.1 流

- 一个网络（Network）N是由一有向图（称为**基础有向图**（underlying digraph）；及二特定的不相交顶点子集X和Y；以及弧集A上定义的非负整数值函数c(·)（**容量函数**）。
- X的每个顶点称为**发点**（source）；
- Y的每个顶点称为**收点**（sink）；
- I = V \ (X ∪ Y)中每个顶点称为**中间顶点**（intermediate vertex）；
- 每弧a上c(a)的值称为a的**容量**（capacity）。

例



流

- 定义在A(D)上的整数值函数f(·)若满足以下条件，就称为网络N上的**流**（flow）：
  ① $0 \le f(a) \le c(a)$，$\forall a \in A(D)$，称为**容量约束**（capacity constraint）。
  ② $f^+(v) = f^-(v)$，$\forall v \in I$，称为**守恒条件**（conservation condition）。
- 流的存在性，零流

记号

- 设f(.)为网络N上的任一流，对任一顶点子集S，令：K=(S,$\overline{S}$):
- 记：

$$f^+(S) = f(S,\overline{S}) = \sum_{uv\in(S,\overline{S})} f(u,v);$$

$$f^-(S) = f(\overline{S},S) = \sum_{wz\in(\overline{S},S)} f(w,z);$$

$$f^+(v) = \sum_{w\in N^+(v)} f(v,w) = f(\{v\},V\setminus\{v\});$$

$$f^-(v) = \sum_{w\in N^-(v)} f(w,v) = f(V\setminus\{v\},\{v\});$$

$$f^+(S) - f^-(S): \text{流出}S\text{的合成流量；}$$

$$valf = f^+(X) - f^-(X): f\text{的流值}$$

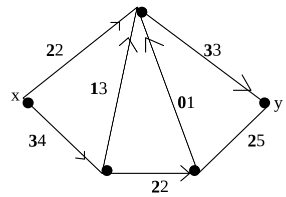
最大流

- 流 f为**最大流**（maximum flow）⇔ 不存在流f‘，使val f' > val$f$ 。
- 求最大流时考虑简化模型：
  - 网络N中不存在有向圈C，使得C中每条弧中的流量大于0；
  - 网络N中不存在(x_i,x_j)-有向路P，使得P中每条弧中的流量大于0；
  - 只需讨论单发点，单收点的网络

割

- 设网络N只有单发点x及单收点y, $S \subset V(D)$, 称$(S, \bar{S})$为N中的割(cut)⇔ $x \in S$, $y \in \bar{S}$。
- 并称 $capK = c(S,\bar{S}) = \sum_{\alpha \in K} c(\alpha)$ 为割 K的容量。
- 称为K**最小割**(minimum cut) ⇔ 不存在割K' 使 cap K' < cap K。

例图



引理9.1.2

- **引理9.1.2** 对N中任一流 f 及任一割$(S, \bar{S})$, 有
$$valf = f^{+}(S) - f^{-}(S)$$
- 证明：注意到
$$f^{+}(v) - f^{-}(v) = \begin{cases} valf, v = x; \\ 0, v \in S \setminus \{x\} \end{cases}$$
因此有:
$$valf = \sum_{v \in S} (f^{+}(v) - f^{-}(v)) = f^{+}(S) - f^{-}(S)$$

定理9.1.3

- **称弧a为： f-零的 ⇔ f(a) = 0** ；
  f-正的 ⇔ f(a) > 0 ；
  f-不饱和的 ⇔ f(a) < c(a) ；
  f-饱和的 ⇔ f(a) = c(a) 。

定理9.1.3 对N中任一流 f及任一割K =$(S, \bar{S})$ 有 valf ≤ cap K；
又，上式中等号成立 ⇔ K =$(S, \bar{S})$中每弧为f-饱和的， 且$(\bar{S},S)$ 中每弧为 f-零的 。

定理9.1.3证明

- 证明：由容量约束知
  $f_{+}(S) \le$ cap K (1)
  $f_{-}(S) \ge 0$ (2)
  再由引理9.1.2知定理的第一个结论成立。又，
(1) 中等号成立 ⇔ （S，$\bar{S}$）中每弧为f-饱和的；
(2) 中等号成立 ⇔ （$\bar{S}$，S）中每弧为f-零的；
  从而定理的第二个结论也成立。 #

推论

- 显然，若 f* 为最大流，K为最小割，则
  valf * ≤ cap K。
- **推论9.1.4** 设流f 及割K , 使valf = cap K ， 则f为最大流，K为最小割。

9.2 最大流最小割定理

- 设f为网络N中的流，P为N中一条路，令

$$\iota(a)=\begin{cases} c(a)-f(a) & a\text{为}P\text{的顺向弧} \\ f(a) & a\text{为}P\text{的反向弧} \end{cases}$$

$$l(P)=\min_{a\in A(P)} l(a)$$

称路P为**f-饱和的** ⇔ l(P) = 0 ；

f-不饱和的 ⇔ l(P) > 0 ；

f-可增路（f-incrementing path）P ⇔ P是以发点x为起点，以收点y为终点的 f-不饱和路 。

修改流

- 若N中有一f-可增路P，则，易见，f 不是最大流。这时，可沿P输送一附加流 而得一新流 f′:

$$f'(a)=\begin{cases} f(a)+\iota(P) & a\text{为}P\text{的顺向弧} \\ f(a)-\iota(P) & a\text{为}P\text{的反向弧} \\ f(a) & \text{其它} \end{cases}$$

容易验证， f′也是一个流，而且流值为valf +l(P)

并称 f′为**基于P的修改流**（revised flow based on P）

定理9.2.1

- **定理9.2.1** N中的流f为最大流⇔ N不含f-可增路 。
- 证明：⇒ 反证，若N中含一f-可增路P，则f不是最大流，因为基于P的修改流f′的值更大。

⇐： 令 S = { v | ∃ f-不饱和（x，v）-路}，则显然有 x∈S， y ∈ $\bar{S}$ ； ∴K = (S, $\bar{S}$) 为割。

注意到，这时(S, $\bar{S}$)的任一弧 a = (u,v)一定是f-饱和的。否则，由于u ∈ S，存在f-不饱和(x,u) -路Q 。因此 Q可通过a延伸为f-不饱和(x,v) -路，从而 v ∈ S，矛盾。

类似地，($\bar{S}$,S)中任一弧 一定是f-零的。

因此，由定理9.1.3知，val*f* = cap K 。从而由推论9.1.4知，f为最大流，同时K为最小割。 #

最大流最小割定理

- 定理9.2.2(Ford & Fulkerson 1956)

在任一网络中，最大流的值等于最小割的容量。

求最大流的算法

- **原理** ：1. 以一已知流f（例如，零流）作为开始。
  2. 系统搜索f-可增路P。若P不存在，停止（f-为最大流）。
  3. 若P存在，求出基于P的修改流f′，令 f ← f′，并转到1。
- **系统搜索f-可增路P标号法** （通过标号‘生长’ f-不饱和树T）开始，标x以l(x) = ∞ 。（此后，在T的生长过程中，T中每个顶点 将标 以 l(v) = ι (Pv) ，其中Pv是T中唯一的(x, v)- 路。）

(1) 若 a =(u,v)为f-不饱和弧，且u已标号， 而v未曾，则标v以

l(v) = min{ l(u), c (a) - f (a) } 。

(2) 若 a =(v,u)为f-正的，且u已标号，而v未曾，则标v以

l(v) = min{ l(u), f(a) }

上述标号过程一直进行到：或者y已标号（“breakthrough”，找到了f-可增路）；或者所有已标号顶点都已扫描，但无顶点可再标号（f为最大流）。

标号程序 (labelling method,Ford & Fulkerson,1957)

以已给流f（例如，零流）作为开始。

① 标x以 l(x) = ∞。扫描 （scan） x 。

② 对正在扫描的（已标号）顶点u，

(i) 检查每条以u为尾的弧 a = (u,v)。如果 a 为f-不饱和的，且顶点v未标号，则标v以 l(v) = min{ l(u), c (a) - f (a) } 。

(ii) 检查每条以u为头的弧 a = (v,u) 。如果 a 为f-正的，且顶点v未标号，则标v以 l(v) = min{ l(u), f(a) } 。

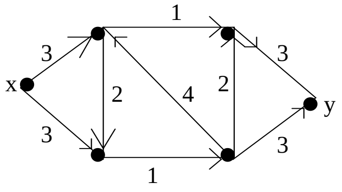
③ 若y已标号（ ‘break through’，找到了一条f-可增路），则转到④ ；否则选一未曾扫描的已标号顶点进行扫描，并回到②。如果已标号顶点都已扫描过，停止（得最大流。且由已标号顶点集S，得最小割（S， S））。

④ 找一f-可增路P，令

$$\tilde{f}(a)=\begin{cases} f(a)+l(y) & \text{当}a\text{为}P\text{上的顺向弧} \\ f(a)-l(y) & \text{当}a\text{为}P\text{上的逆向弧，} \\ f(a) & \text{其它} \end{cases} \qquad \tilde{f}\to f$$

去掉全部标号，并回到① 。

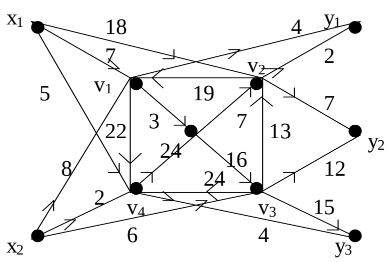
例图



例图

- P159

例图



标号程序算法分析

- 上述算法还不是‘好’算法，例如上图中的网络，其最大流的值为2m。但若标号程序以零流开始，且反复地选取xuvy及xvuy为 f-可增路，则总共要进行2m+1次标号程序，因此其计算量为输入长（= O(log2m))的指数函数。
- Edmonds & Karp (1970) 证明，若在上述标号程序中采用first labelled first scan （即广度优先）法则，则，可使该算法成为好算法（复杂性为O(νε2) ）