最大流算法的应用

- 带需求的流通
- 运输问题
- 二部图的最大匹配
- 赋权二部图的匹配
- 图像分割

带需求的流通

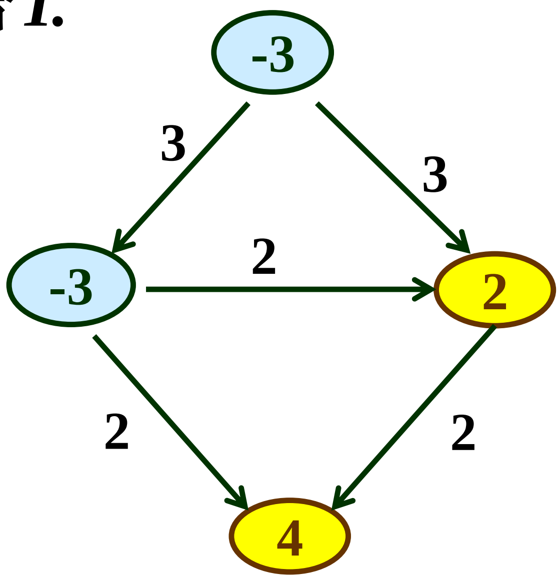
给定带需求的流通图 $N=<V,E,c,S,T>$, $\forall v\in V$ 存在需求 d_v，所有发点集合 S，所有收点集合 T.

•收点：$d_v > 0$，表示 v 对流有 d_v 的需求

•发点：$d_v < 0$，表示 v 有 $-d_v$ 的供给

•结点：$d_v = 0$，v 不是发点和收点

所有容量和需求都是整数



带需求的流通

问是否存在可行流通？即存在函数 $f: E\to \mathbf{R}^*$，满足

(1) 容量条件：$\forall e\in E$，$0\le f(e)\le c_e$

(2) 需求条件：$\forall v\in V$，$f^{\text{in}}(v) - f^{\text{out}}(v) = d_v$

必要条件

命题1 如果存在一个带需求$\{d_v\}$的可行流通，那么 $\sum_v d_v = 0$

证 假设存在可行流通 f. 那么

$$\sum_v d_v = \sum_v (f^{\text{in}}(v) - f^{\text{out}}(v))$$

对于每条边 $e=\langle u,v\rangle$，值 $f(e)$恰好在 f^{in}和 f^{out}中各出现1次. 两项抵消；总和是0.

根据命题1，有

$$\sum_{v:d_v>0} d_v = \sum_{v:d_v<0} -d_v$$

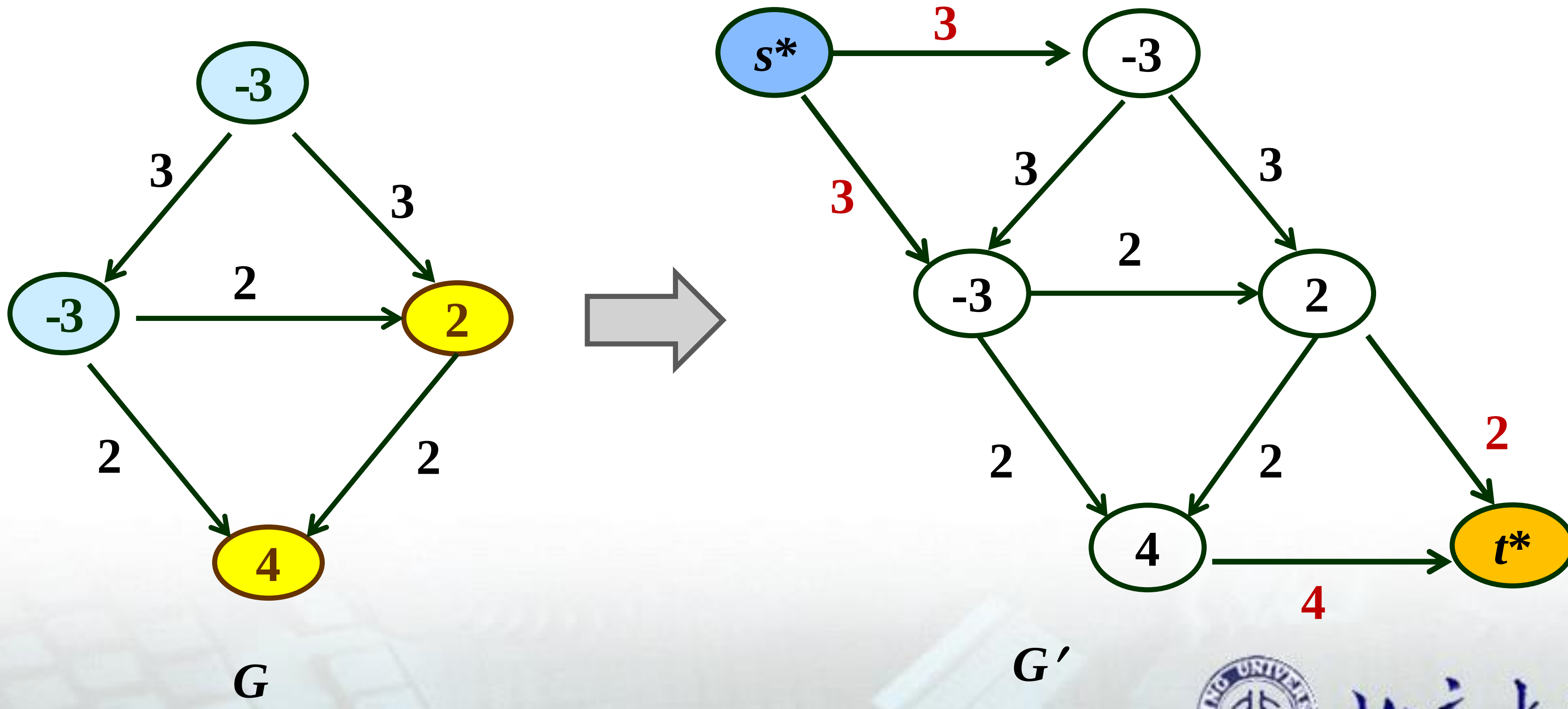
将该和记作D.

用最大流建模

加超结点 s^* 和 t^*，构造单发点单收点的容量网络 G'

从 s^*到每个 $v \in S$ 加边 $e = \langle s^*, v \rangle$，令 $c_e = -d_v$

从每个 $v \in T$ 加边 $e = \langle v, t^* \rangle$，令 $c_e = d_v$



算法与拓展

命题2 G'没有大于D的 s^*-t^* 流，因为 $c(\{s^*\},V\cup\{t^*\})=D$, 其中

$$D=\sum_{v:d_v>0} d_v=\sum_{v:d_v<0} -d_v$$

定理 G 中存在一个带需求$\{d_v\}$的可行流通，当且仅当G'的最大 s^*-t^* 流有值 D.

算法

1. 将流通图 G 转换为对应的容量网络 G'
2. 对 G'运行最大流算法找到 s^*-t^*最大流 f
3. 如果 $v(f)=D$，则 G 存在带需求$\{d_v\}$的可行流通
   否则在该需求下不存在可行流通

拓展 带需求和下界的可行流通. 每条边增加下界 l_e , $0\le l_e\le c_e$. 修改容量条件，对每个$e\in E$，$l_e\le f(e)\le c_e$. 需求条件不变.

运输问题

运输问题 (**Hitchcock**问题)

有m个产地$A_1, A_2, \ldots, A_m$和 n个销地$B_1, B_2, \ldots, B_n$, A_i 的产量为 a_i, B_j的销量为 b_j. 从A_i 到 B_j 的单位运费为 w_{ij}, $1 \le i \le m$, $1 \le j \le n$.

假设产销平衡, 即

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$$

制订调运方案使总运费最少.

调运方案 $x = \{ x_{ij} \}$: $x_{ij} \ge 0$, $1 \le i \le m$, $1 \le j \le n$

$$\sum_{j=1}^{n} x_{ij} = a_i \quad (1 \le i \le m), \qquad \sum_{i=1}^{m} x_{ij} = b_j \quad (1 \le j \le n)$$

x的总费用 $w(x) = \sum_{i=1}^{m} \sum_{j=1}^{n} w_{ij} x_{ij}$

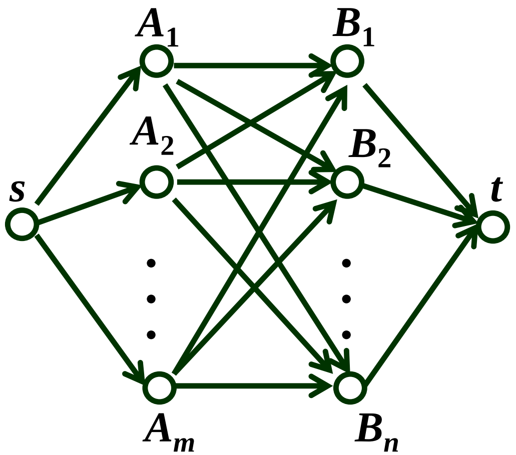
运输问题是求使 $w(x)$ 最小的调运方案 x.

建模:最小费用流

添加发点 s 和收点 t, 作容量-费用网络$N=<V,E,c,w,s,t>$, 其中

$$V = \{ s, t, A_1, A_2, \ldots, A_m, B_1, B_2, \ldots , B_n \}$$

$$E = \{< s, A_i > \mid 1\le i\le m\}\cup\{< B_j, t > \mid 1\le j\le n\}$$
$$\cup\{< A_i, B_j > \mid 1\le i\le m, 1\le j\le n\}$$



$$c(s, A_i) = a_i,\quad w(s, A_i) = 0,\quad 1\le i\le m$$
$$c(B_j, t) = b_j,\quad w(B_j, t) = 0,\quad 1\le j\le n$$
$$c(A_i, B_j) = +\infty,\quad w(A_i, B_j) = w_{ij},\quad 1\le i\le m, 1\le j\le n$$

流量 $v_0 = \sum_{i=1}^{m} a_i$ 的可行流恰好对应运输问题的调运方案.

总运费最小的调运方案就是流量v_0的最小费用流

二部图匹配

定义 设简单二部图$G=<A,B,E>$, $M\subseteq E$, 如果M中任意两条边都不相邻, 则称M是G的匹配. 边数最多的匹配称作最大匹配. 当$|A|=|B|=n$时, 边数为n的匹配称作完美匹配.

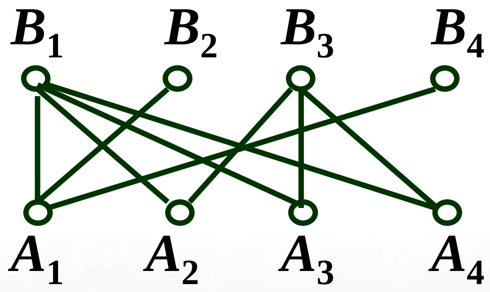
例5 有4名新入学的硕士生和4位硕导, A_1申请B_1,B_2或B_4作导师, A_2,A_3和A_4都申请B_1或B_3作导师. 每名硕士生有一位导师, 每位硕导只收一名新生. 如何分配才能尽可能满足学生要求.

作二部图$G=<A,B,E>$, 其中

$A=\{ A_i \mid 1\leq i \leq 4 \}$

$B=\{ B_j \mid 1\leq j \leq 4 \}$

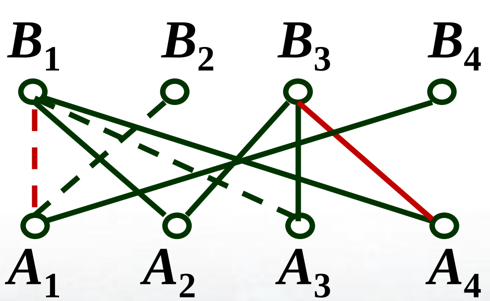
$E=\{ (A_i,B_j) \mid A_i$ 申请 B_j 作导师, $1\leq i \leq 4, 1\leq j \leq 4 \}$



增广交错路径

定义 设 M 是二部图 G 的匹配.

- 称 M中的边为匹配边
- 不属于 M 的边为非匹配边
- 与匹配边关联的顶点为饱和点
- 不与匹配边关联的顶点为非饱和点
- G 中由匹配边和非匹配边交替构成的路径称为交错路径
- 起点和终点都是非饱和点的交错路径称为增广交错路径



M与 P

M={ (B_1,A_1), (B_3,A_4) }

饱和点：B_1, B_3, A_1, A_4

非饱和点：A_2, A_3, B_2, B_4

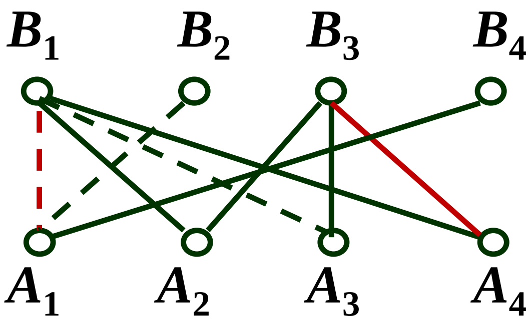
增广交错路径 P：$A_3B_1A_1B_2$

最大匹配的条件

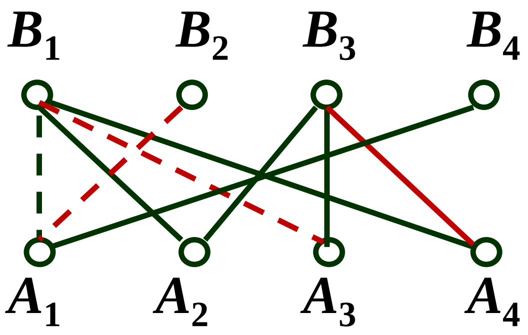
引理14 设 M 是二部图 G 的一个匹配, P是一条关于M 的增广交错路径, 则

$$M' = M \oplus E(P)$$

是一个匹配 且 $|M'| = |M| + 1$.



M与 $E(P)$

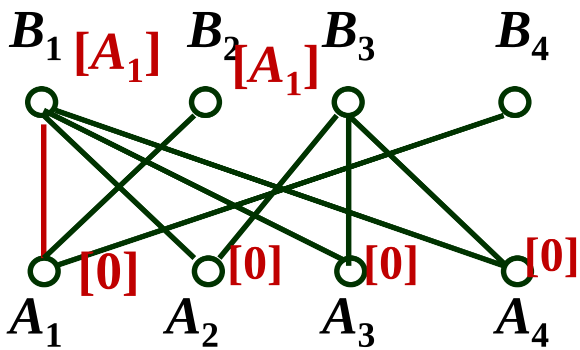


M'

定理8 二部图的匹配是最大匹配当且仅当不存在关于它的增广交错路径.

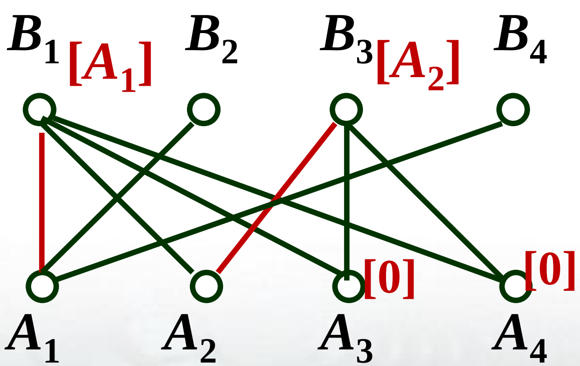
匈牙利算法

匈牙利算法　从一个初始匹配 M 开始, 每次找一条增广交错路径 P, 令 $M \leftarrow M \oplus E(P)$, 直到不存在增广交错路径为止.



增广交错路径：A_1B_1

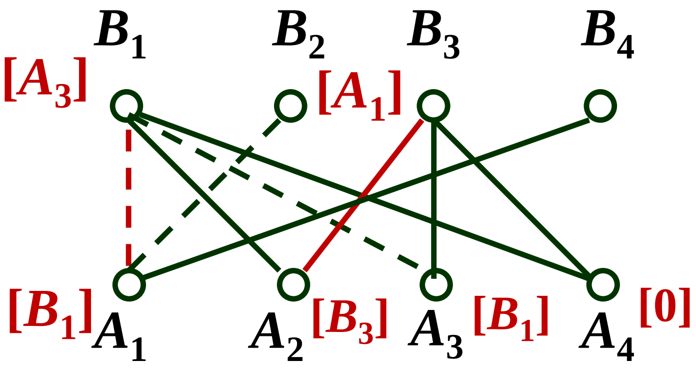
$M=\{(A_1,B_1)\}$



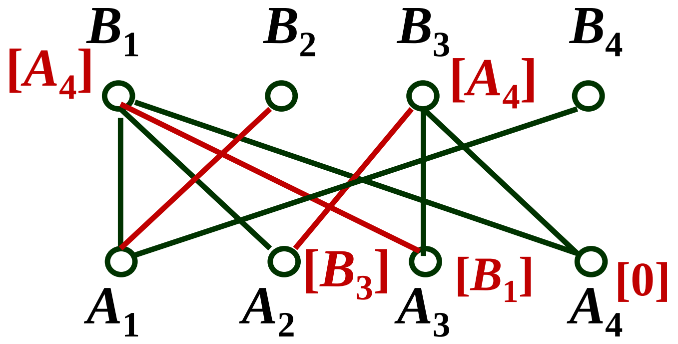
增广交错路径：A_2B_3

$M=\{(A_1,B_1),(A_2,B_3)\}$

匈牙利算法（续）



增广交错路径：$B_2A_1B_1A_3$



匹配 $M\oplus E(P)$
$=\{(A_1,B_2),(A_2,B_3),(A_3,B_1)\}$

算法时间：$O(\min\{|A|,|B|\}\cdot|E|)$
阶段数：$\min\{|A|,|B|\}+1$, 每个阶段检查时间： $O(|E|)$

二部图最大匹配与最大流

最大匹配转化成最大流

设 $G=<A,B,E>$，作容量网络 $N_G = <V, E', c, s, t>$, 其中 $c \equiv 1$,

$V=\{ s, t \}\cup A\cup B$

$E'=\{< s, A_i > \mid A_i\in A\}\cup\{<B_j, t > \mid B_j\in B\}\cup\{<A_i, B_j> \mid (A_i,B_j)\in E\}$

G 的匹配 $\Leftrightarrow$ N_G 上的 0-1可行流

前向边是非饱和边, 后向边是饱和边, 增广链$\Leftrightarrow$增广交错路径.

匈牙利算法是FF算法的应用.

算法设计思想:

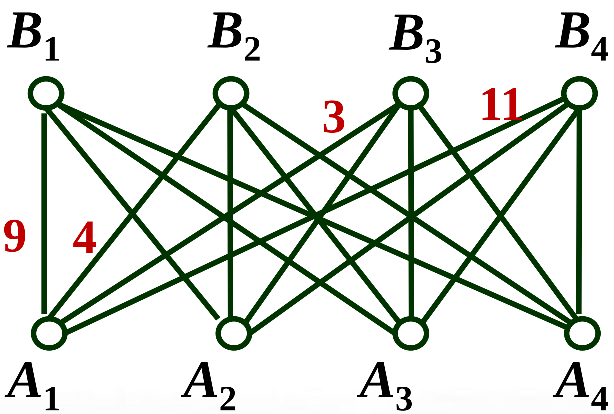
1. 构造对应容量网络 N_G
2. 对 N_G 应用 Dinic 算法求最大流 f
3. 再把 f 转化为G的匹配 M

时间: $O(n^{1/2}m)$

赋权二部图完美匹配

指派问题 给定赋权完全二部图$G=\langle A,B,E,w\rangle$, 其中$|A|=|B|=n$, $w: E\to R$, 求 G 的权最小的完美匹配 M.

例6 有4项任务由4个人完成, 每人完成一项. 预计每个人完成每一项任务的时间如下表所示. 如何分配才能是完成任务的总时间最少?



	任务			
	B_1	B_2	B_3	B_4
人员 A_1	9	4	3	11
人员 A_2	8	5	8	4
人员 A_3	3	8	3	1
人员 A_4	10	6	9	6

使用匈牙利算法，时间 $O(n^3)$

指派问题与线性规划

指派问题可表成 线性规划(P)

$$x_{ij}=\begin{cases}1, & (A_i,B_j)\in M,\\ 0, & \text{否则},\end{cases}$$

$$\min \quad \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}x_{ij}$$

$$s.t. \quad \sum_{j=1}^{n} x_{ij}=1, \qquad 1\le i\le n$$

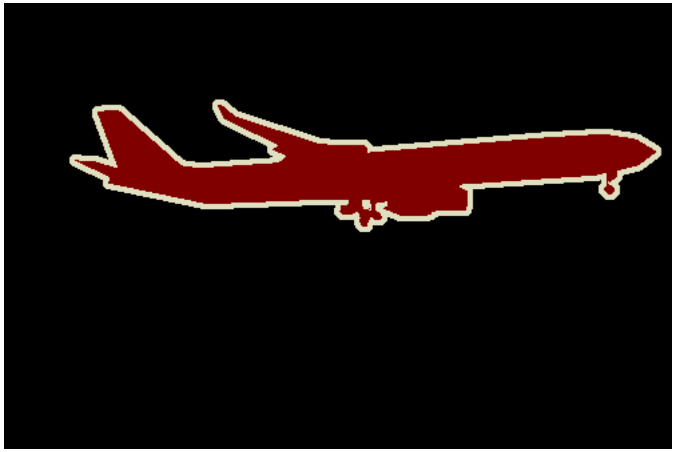
$$\sum_{i=1}^{n} x_{ij}=1, \qquad 1\le j\le n$$

$$x_{ij}\ge 0, \qquad 1\le i\le n, 1\le j\le n$$

北京大學

最大流应用：图像分割

- 图像分割：将一幅图片的前景与背景分离
  对每个像素进行“前景/背景”的标记



图片 分割后

对图像分割的建模

- 设V是基本图像中的像素集合. 用E表示所有相邻像素对的集合，构成无向图 $G=<V,E>$.
- 像素 i 属于前景的可能性：a_i，属于背景的可能性：b_i
- 如果 $a_i > b_i$，像素 i 标为前景，否则标为背景. 标记为前景的像素构成集合A，标记为背景的像素构成集合B.
- 如果像素 i 的邻居都标为“背景”，那么倾向于将 i 标为背景，使标记更“光滑”. 对于每对邻居像素(i, j)，定义分离罚分 $p_{ij} > 0$ 来惩罚 i 和 j 标记不同.
- 图像分割问题：将像素点集划分为 A 和B，使得划分的权值 $q(A,B)$最大化（即得到最优标记）. 其中

$$q(A,B)=\sum_{i\in A}a_i+\sum_{j\in B}b_j-\sum_{\substack{(i,j)\in E\\ |A\cap\{i,j\}|=1}}p_{ij}$$

最优标记与最小割

- 两个问题的相似性

  都涉及图的点割集(A,B)

- 区别：

  图像最优标记的图是无向图，流网络最小割是有发点 s、收点 t 的有向图

  最优标记是最大化问题，最小割是最小化问题

  最优划分问题的结点有参数 a_i 和 b_i，最小割问题边有参数

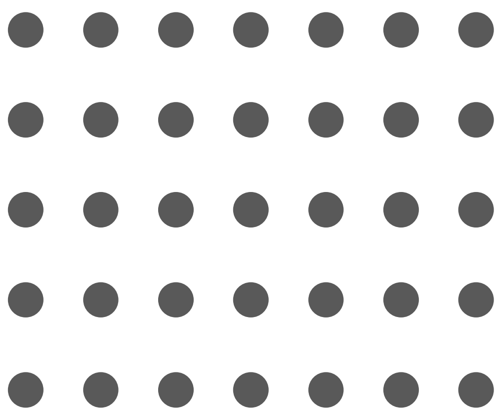
- 思路：

  将最优标记的邻居图转化为一个容量网络

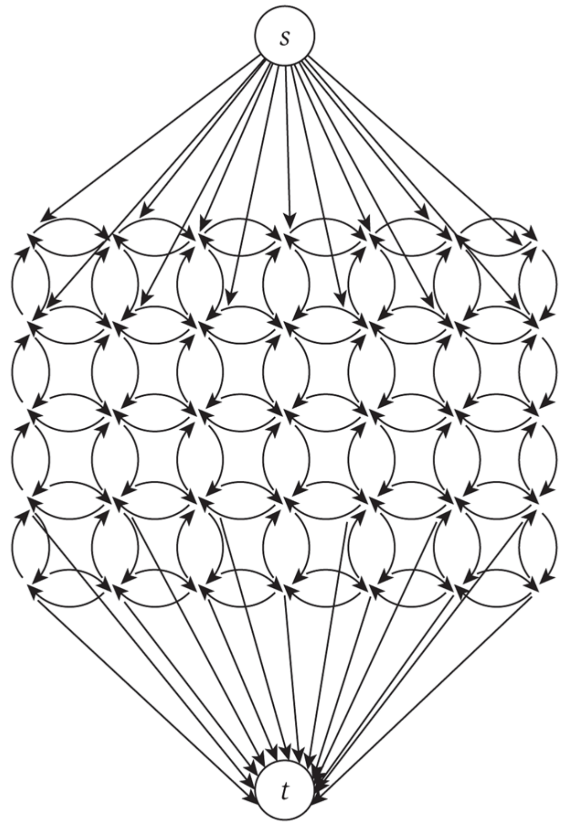
转化为容量网络G'

- 设$G=<V,E>$是图像最优标记问题的无向图，将每条边$(u,v)\in E$ 转化成一对有向边$<u,v>$和$<v,u>$，构成有向图
- 超源点 s (前景), 超汇点 t (背景)

  加边 $<s,v>$ 和 $<v,t>$, $\forall v\in V$.



图像像点阵列



对应容量网络G'(部分边)

G的最大化与G'的最小化

- 令 $$Q=\sum_i (a_i+b_i)$$
$$\Rightarrow \sum_{i\in A} a_i+\sum_{j\in B} b_j = Q-\sum_{i\in A} b_i-\sum_{j\in B} a_j$$

- $q(A,B)$的最大化转变为 $q'(A,B)$ 的最小化

$$q(A,B) = Q-q'(A,B)$$

$$q(A,B)=Q-\sum_{i\in A} b_i-\sum_{j\in B} a_j-\sum_{\substack{(i,j)\in E\\ |A\cap\{i,j\}|=1}} p_{ij}$$

$$q'(A,B)=\sum_{i\in A} b_i+\sum_{j\in B} a_j+\sum_{\substack{(i,j)\in E\\ |A\cap\{i,j\}|=1}} p_{ij}$$

容量分配

- 边容量

  对边 $e = <s,i>$，令$c_e = a_i$；对边 $e = <i,t>$，令 $c_e = b_i$.

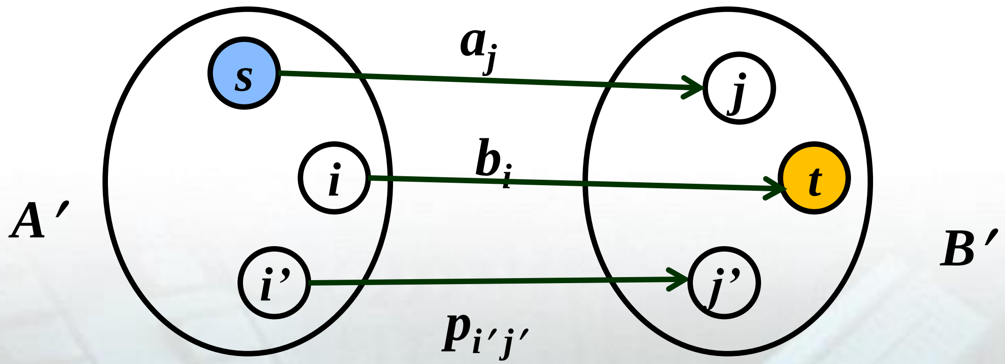
  对邻居边 $e = <j,i>$ 和 $e' = <i,j>$，令 $c_e = c_{e'} = p_{ij}$.

- 穿过割 (A',B') 的边分成三类：

  边 $<s,j>$，$j \in B'$：为割的容量贡献 a_j.

  边 $<i,t>$，$i \in A'$：为割的容量贡献 b_i.

  边 $<i',j'>$, $i' \in A'$, $j' \in B'$：为割的容量贡献 $p_{i'j'}$



割的容量

所有边$< s,j >$，$j\in B'$，对 割容量贡献$\sum\limits_{j\in B} a_i$

所有边$< i,t >$，$i\in A'$，对割容量贡献$\sum\limits_{i\in A} b_i$

所有边$< i,j>$，$i\in A'$，$j\in B'$；对割容量贡献$\sum\limits_{i\in A,j\in B} p_{ij}$

$$
\begin{aligned}
&c(A',B') \\
&= \sum_{i\in A} b_i + \sum_{j\in B} a_j + \sum_{\substack{(i,j)\in E \\ |A\cap\{i,j\}|=1}} p_{ij} \\
&= q'(A,B)
\end{aligned}
$$

图像分割算法

主要步骤

- 建立像素邻居图 G
- 将 G 转化成容量网络 G'
- 求解 G'的最小割 (A',B')
- 删除 s 和 t 可以得到划分 (A, B)
- 将 A 中像素标记为前景，B 中像素标记为背景

时间复杂度$O(n^3)$