6.4 赋权匹配的有效算法－1

一、引言

权：给定一个图$G=(V,E)$，$\forall [v_i,v_j]\in E$，给定一个数$w_{ij}\ge 0$，称w_{ij}为边$[v_i,v_j]$的权。

最大权匹配：求G的一个匹配，使得匹配中的权之和最大。

在最大权匹配中，可以假设：

（1） G是完全图；

（2） G有偶数个点；

（3） 对于二部图，是完全的，且两组节点相等；

（4） 最优解是一个完美匹配；

（5） 转化为最小权匹配问题：$c_{ij}=W-w_{ij}$，其中$W=\max\{w_{ij}\}$。

二、二部图的赋权匹配问题

二部图的赋权匹配问题，又称为指派问题。

(i) 用最小费用流算法求解二部图的赋权匹配问题

——直接将二部图的匹配问题化成最小费用流问题，然后用最小费用流的原始对偶算法（圈算法或迭加算法）求解，从而得到二部图的匹配问题的有效算法。

将二部图的匹配问题归结为简单网络上的最大流问题。给定一个二部图$B=(V,U,E)$，定义一个单位容量的网络$N(B)=(s,t,W,A)$，其中s,t是新增的两个特殊节点，$W=\{s,t\}\cup V\cup U$，且A由下述三种类型的弧组成：

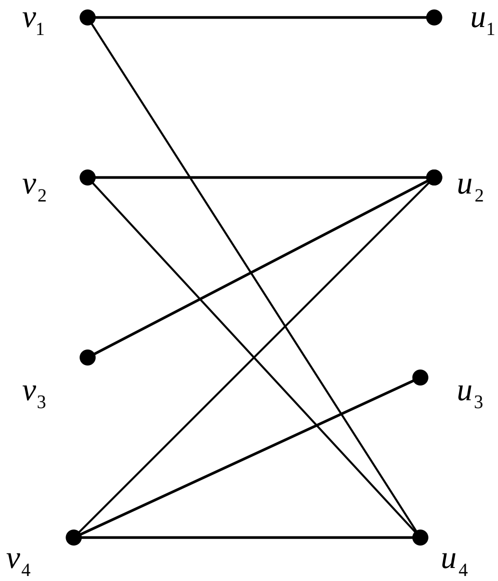
（1）$\forall v\in V,(s,v)$弧；

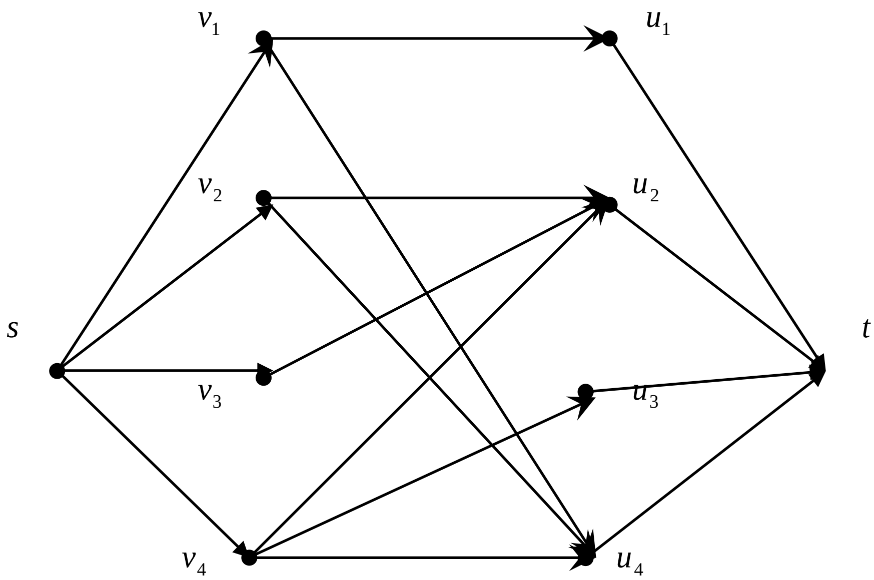
（2）$\forall u\in U,(u,t)$弧；

（3）对$\forall v\in V,u\in U,(v,u)\in E$，弧$(v,u)$。

其中（1）和（2）两类弧的容量为1，费用为0。第（3）类弧的容量为1，费用为c_{ij}。

如：





(ii) 二部图的赋权匹配问题的匈牙利算法

匈牙利算法 1955 年由 Kuhn 利用匈牙利数学家 D.Konig（康尼格）的一个定理构造了这个算法，被称之为匈牙利算法。

(iii) 直接用原始－对偶算法求解二部图的赋权匹配问题

二部图的赋权匹配问题是 Hitchcock 问题的一个特殊情形，可以用 Hitchcock 问题的原始－对偶算法求解。

1. 二部图的赋权匹配问题的数学规划模型

记 $n=|V|=|U|$， $B=(V,U,E)$ 为完全二部图。设：

$$x_{ij}=\begin{cases}1\text{：}\quad \text{边}[v_i,u_j]\text{是匹配边}\\ 0:\text{边}[v_i,u_j]\text{不是匹配边}\end{cases}$$

显然，表示一个完美匹配的 x_{ij} 的值必须满足：

$$\sum_{j=1}^{n} x_{ij}=1,\ i=1,2,\cdots,n$$
$$\sum_{i=1}^{n} x_{ij}=1,\ i=1,2,\cdots,n$$
$$x_{ij}\ge 0$$

数学规划模型为：

$$\begin{aligned}&\min\sum_{i,j}c_{ij}x_{ij}\\ &\sum_{j=1}^{n} x_{ij}=1, i=1,2,\cdots,n \qquad (1)\\ &\sum_{i=1}^{n} x_{ij}=1, i=1,2,\cdots,n \qquad (2)\\ &x_{ij}\ge 0\end{aligned}\qquad \text{(BLP)}$$

注：

（1） 上述模型中没有限制 x_{ij} 的值必须取 0 或 1。因此，这个规划很可能有分数最优解，而分数最优解将不对应任何可行的匹配。

（2） 分数最优解决不可能是该规划的基可行解。

（3） 用单纯形类方法求解该问题，得到的解必然是整数解，从而它对应一个匹配。

（4） 总存在一个最优解，它对应一个完美匹配。

2. 原始－对偶算法

（1）一般线性规划的原始－对偶算法

$$\begin{aligned}&\min z=c^T x\\ &\text{s.t.}\, Ax=b\ge 0 \quad \text{(LP)}\\ &\qquad x\ge 0\end{aligned}$$

$$\begin{aligned}&\max \varpi=b^T y\\ &\text{s.t.}\quad A^T y\le c \qquad \text{(D)}\\ &\qquad y\text{无限制}\end{aligned}$$

y 是(D)的一个可行解

$$J=\{j: A_j^T y=c_j\}$$

$$\begin{aligned}&\min \xi=\sum_{i=1}^{m} x_i^a\\ &\text{s.t.}\ \sum_{j\in J} a_{ij}x_j+x_i^a=b_i, i=1,2,\cdots,m\\ &\qquad x_j\ge 0,\quad j\in J \qquad\qquad \text{(RP)}\\ &\qquad x_j=0,\quad j\notin J\\ &\qquad x_i^a\ge 0,\ i=1,2,\cdots,m\end{aligned}$$

$$\Longrightarrow$$

$$\begin{aligned}&\max \varpi=b^T y\\ &\text{s.t.}\ A_j^T y\le 0,\quad j\in J \qquad \text{(DRP)}\\ &\qquad y_i\le 1,\ i=1,2,\cdots,m\\ &\qquad y_i\text{无限制}\end{aligned}$$

用单纯形类方法求解(RP)， 若：

- (RP)的最优值 $\xi_{\text{opt}}=0$，则得到(LP)和(D)的最优解，算法终止；
- (RP)的最优值 $\xi_{\text{opt}}>0$，设 $\bar{y}$ 是(DRP)的最优解，若 $A_j^T\bar{y}\le 0, j\notin J$，则(D)无上界，从而(LP)不可行，算法终止；
- 否则，取 $\theta\le\theta_1=\min\limits_{\substack{A_j^T\bar{y}>0,\\ j\notin J}}\left\{\dfrac{c_j-A_j^T y}{A_j^T\bar{y}}\right\}$， $y:=y+\theta\bar{y}$

（2）Hitchcock 问题的原始－对偶算法——$\alpha\beta$ 算法

$$\min \sum_{i,j} c_{ij} f_{ij}$$

$$\sum_{j=1}^{n} f_{ij} = a_i, i = 1,2,\cdots,m \qquad (1)$$

(LP)

$$\sum_{i=1}^{m} f_{ij} = b_j, j = 1,2,\cdots,n \qquad (2)$$

$$f_{ij} \ge 0$$

$$\max \varpi = \sum_{i=1}^{m} a_i \alpha_i + \sum_{j=1}^{n} b_j \beta_j$$

$$\alpha_i + \beta_j \le c_{ij}$$

(D)

α_i, β_j 无限制

$$i = 12,\cdots,m; \ j = 1,2,\cdots,n$$

(α,β) 是(D)的一个可行解

允许弧集合 $IJ = \{(i,j): \alpha_i + \beta_j = c_{ij}\}$

$$\max \sum_{(i,j)\in IJ}\sum f_{ij}$$

$$\sum_{j=1}^{n} f_{ij} \le a_i, i = 1,2,\cdots,m$$

$$\sum_{i=1}^{m} f_{ij} \le b_j, j = 1,2,\cdots,n \quad \text{(RP')}$$

$$f_{ij} \ge 0, (i,j) \in IJ$$

$$f_{ij} = 0, (i,j) \notin IJ$$

$$\max \varpi = \sum_{i=1}^{m} a_i \alpha_i + \sum_{j=1}^{n} b_j \beta_j$$

s.t. $\alpha_i + \beta_j \le 0, \quad (i,j) \in IJ$

$\alpha_i \le 1, \ \beta_j \le 1$ (DRP)

$i = 1,2,\cdots,m$

α_i, β_j 无限制

$(\bar{\alpha},\bar{\beta})$ 是(DRP)

调整对偶可行解 $(\alpha,\beta) := (\alpha,\beta) + \theta(\bar{\alpha},\bar{\beta})$

（3）指派问题的原始－对偶算法——$\alpha\beta$ 算法

$$
\begin{aligned}
&\min \sum_{i,j} c_{ij} x_{ij} \\
&\sum_{j=1}^{n} x_{ij} = 1, i = 1,2,\cdots,n \qquad (1) \\
&\sum_{i=1}^{n} x_{ij} = 1, j = 1,2,\cdots,n \qquad (2) \\
&x_{ij} \ge 0
\end{aligned}
\qquad \text{(LP)}
$$

$$
\begin{aligned}
&\max \varpi = \sum_{i=1}^{n} \alpha_i + \sum_{j=1}^{n} \beta_j \\
&\alpha_i + \beta_j \le c_{ij} \\
&\alpha_i, \beta_j \text{无限制} \\
&i = 12,\cdots,n; j = 1,2,\cdots,n
\end{aligned}
\qquad \text{(D)}
$$

(α, β) 是(D)的一个可行解

允许弧集合 $IJ = \{(i, j) : \alpha_i + \beta_j = c_{ij}\}$

$$
\begin{aligned}
&\max \sum_{(i,j)\in IJ}\sum x_{ij} \\
&\sum_{j=1}^{n} x_{ij} \le 1, i = 1,2,\cdots,n \\
&\sum_{i=1}^{n} x_{ij} \le 1, j = 1,2,\cdots,n \\
&x_{ij} \ge 0, (i, j) \in IJ \\
&x_{ij} = 0, (i, j) \notin IJ
\end{aligned}
\qquad \text{(RP')}
$$

$$
\begin{aligned}
&\max \varpi = \sum_{i=1}^{m} \alpha_i + \sum_{j=1}^{n} \beta_j \\
&\text{s.t. } \alpha_i + \beta_j \le 0, \quad (i, j) \in IJ \\
&\qquad \alpha_i \le 1,\ \beta_j \le 1 \\
&\qquad i = 1,2,\cdots,m \\
&\qquad \alpha_i, \beta_j \text{无限制}
\end{aligned}
\qquad \text{(DRP)}
$$

$(\overline{\alpha}, \overline{\beta})$ 是(DRP)

调整对偶可行解 $(\alpha, \beta) := (\alpha, \beta) + \theta(\overline{\alpha}, \overline{\beta})$

求解(RP’)和(DRP)的方法：

(RP’)是一个最大流问题的数学规划模型。虚拟一个超级发点 s 和超级收点 t，当 $(i, j) \in IJ$ 时，自 i 到 j 有一条弧，其容量为∞，它保证仅当弧 (i, j) 是允许弧时，x_{ij} 才可以大于 0。

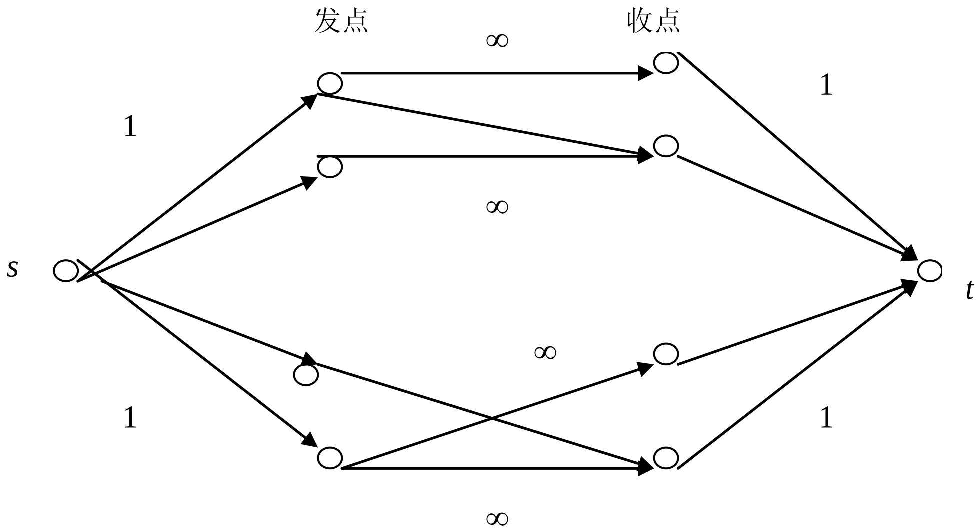


图 1 . (RP’)对应的最大流问题

显然，(RP’)对应的最大流问题，实际上是由当前的允许弧集合 $IJ=\{(i,j):\alpha_i+\beta_j=c_{ij}\}$ 构成的二部图的基数匹配，可以用增广路的方法求解上述二部图的最大基数匹配问题。

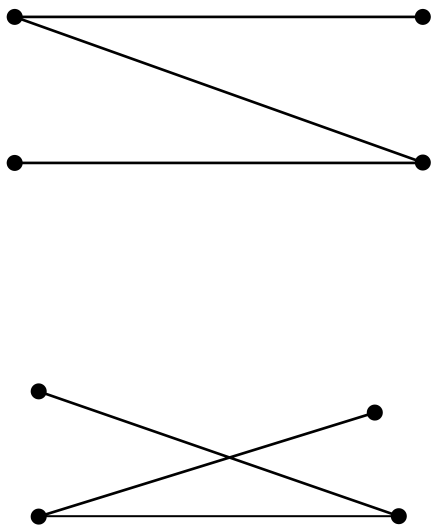


图 2. (RP’)对应的由允许边集合构成的二部图的最大基数匹配问题

- 当搜索增广路失败时，说明当前的允许边集合中不含有增广路，得到(RP’)和(DRP)的最优解，调整对偶可行解 $(\alpha,\beta):=(\alpha,\beta)+\theta(\overline{\alpha},\overline{\beta})$，使得新的边变为允许边，得到新的(RP’)，继续求解(RP’)。

- 一旦找到一条增广路，则利用它增广匹配，并且以新的匹配重新开始搜索增广路，直到得到(RP’)的最优解为止。

用搜索算法求解(RP’)对应的由允许边集合构成的二部图的最大基数匹配问题，搜索算法结束时，得到最大匹配，同时得到对应于(DRP)的最优解$(\overline{\alpha},\overline{\beta})$。

$$\theta_1=\min_{\substack{i\in I^*\\ j\notin J^*}}\left(\frac{c_{ij}-\alpha_i-\beta_j}{2}\right)$$

其中I^*和J^*表示得到匹配时：

$I^*=\{i:i$为已标号的V中的点$\}$，$J^*=\{j:j$为已标号的U中的点$\}$

所以新的对偶可行解$(\alpha,\beta):=(\alpha,\beta)+\theta_1(\overline{\alpha},\overline{\beta})$，其中

$$\alpha_i:=\begin{cases}\alpha_i+\theta_1, i\in I^*\\ \alpha_i-\theta_1, i\notin I^*\end{cases},\quad \beta_j:=\begin{cases}\beta_j-\theta_1, j\in J^*\\ \beta_j+\theta_1, j\notin J^*\end{cases}$$

在一个阶段（对偶可行解的一次调整）中：

- 在允许边构成的二部图中，交替搜索增广路（求解 RP’）;
- 调整对偶变量，使允许边集合不断发生变化，构造新的 RP’对应的允许边构成的二部图。

求θ_1的计算量：$O(n^2)$。

为了降低计算复杂度，保存和不断修改两个数组：

$$slack[u_j]=\min_{i\in I^*}\{c_{ij}-\alpha_i-\beta_j\}\quad (j=1,2,\cdots,n)$$

$$nhbor[u_j]=\arg\min_{i\in I^*}\{c_{ij}-\alpha_i-\beta_j\}\quad (j=1,2,\cdots,n)$$

$slack[u_j]$表示对一切标号点v_i，$c_{ij}-\alpha_i-\beta_j$的最小值；

$nhbor[u_j]$表示使$slack[u_j]$达到最小的v_i。

所以： $slack[u_j]=0\Rightarrow[nhbor[u_j],u_j]$**是一条允许边，**

或者： $slack[u_j]>0\Rightarrow[nhbor[u_j],u_j]$**不是一条允许边**

所以：

$$\theta_1=\min_{\substack{i\in I^*\\ j\notin J^*}}\left(\frac{c_{ij}-\alpha_i-\beta_j}{2}\right)=\frac{1}{2}\min_{slack[u_j]>0}\{slack[u_j]\}$$

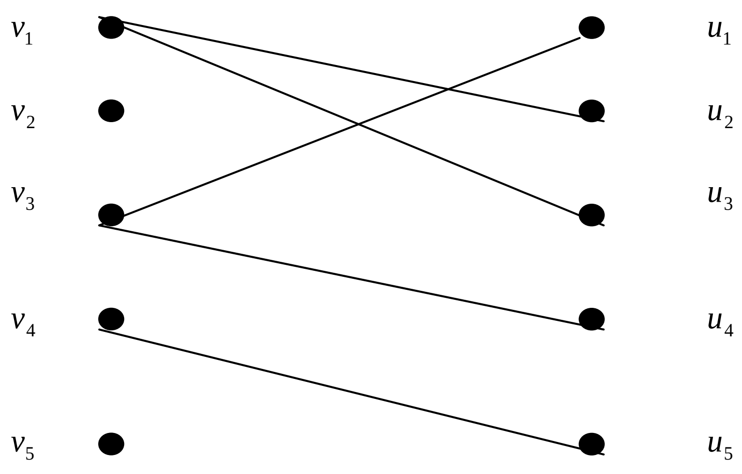
初始对偶可行解：$\alpha_i = 0, i = 1,2,\cdots,n, \quad \beta_j = \min\{c_{ij} : 1 \le i \le n\}$

允许边集合：$IJ = \{(v_i, u_j) : \alpha_i + \beta_j = c_{ij}\}$

具体算法见【参考资料 2】。

例如：用矩阵表示的指派问题：

		u_1	u_2	u_3	u_4	u_5
	β	3	2	1	1	2
	α					
v_1	0	7	2	1	9	4
v_2	0	9	6	9	5	5
v_3	0	3	8	3	1	8
v_4	0	7	9	4	2	2
v_5	0	8	4	7	4	8



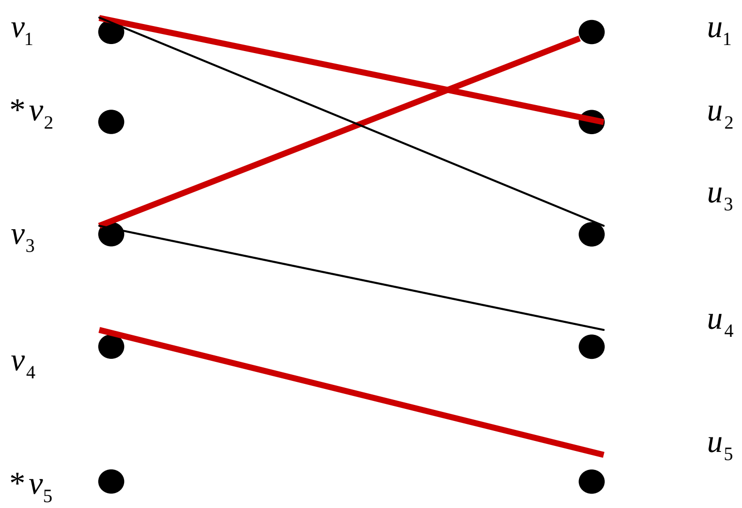
允许边集合对应的二部图

开始搜索允许边集合对应的二部图的增广路：前三次得到三条匹配边，下一次搜索如下图，$I^* = \{v_2, v_5\}$

$slack[u_j]$表示对一切标号点v_i，$c_{ij} - \alpha_i - \beta_j$的最小值；

$nhbor[u_j]$表示使$slack[u_j]$达到最小的v_i。

	α \ β	u_1: 3	u_2: 2	u_3: 1	u_4: 1	u_5: 2
v_1	0	7	2	1	9	4
v_2	0	9	6	9	5	5
v_3	0	3	8	3	1	8
v_4	0	7	9	4	2	2
v_5	0	8	4	7	4	8
slack:		8-3=5	4-2=2	7-1=6	4-1=3	5-2=3
nhbor:		5(v_5)	5(v_5)	5(v_5)	5(v_5)	2(v_2)



允许边集合对应的二部图

此时不存在增广路，从而已经得到最大匹配，需要调整对偶变量。

$$\theta_1=\frac{1}{2}\min_{slack[u_j]>0}\{slack[u_j]\}=\frac{1}{2}slack[u_2]=1$$

$$\alpha_i:=\begin{cases}\alpha_i+\theta_1, i\in I^*\\ \alpha_i-\theta_1, i\notin I^*\end{cases}\qquad \beta_j:=\begin{cases}\beta_j-\theta_1, j\in J^*\\ \beta_j+\theta_1, j\notin J^*\end{cases}$$

	α \ β	u_1: 4	u_2: 3	u_3: 2	u_4: 2	u_5: 3
v_1	-1	7	②	①	9	4
v_2	1	9	6	9	5	5
v_3	-1	③	8	3	①	8
v_4	-1	7	9	4	2	②
v_5	1	8	④	7	4	8

定理 6.12 上述求完全二部图$B=(V,U,E)$的最小权匹配算法是正确的，且其计算复杂度为$O(n^3)$。

证明： 上述算法是 Hitchcock 问题的$\alpha\beta$算法用于求解求完全二部图$B=(V,U,E)$的最小权匹配问题，必然得到该问题的线性规划模型(BLP)的最优解，且该问题的最优解是具有单位容量限制的一个最大流问题的解，所以最优解的每个分量$x_{ij}=0$或者 1，从而对应一个匹配，所以，算法是正确的。

容易验证：

所需要的阶段数为：$O(n)$

每个阶段的计算复杂度为：$O(n^2)$

从而计算复杂度为$O(n^3)$。