



# 运筹学基础

讲者：顾乃杰 教授、黄章进 副教授

计算机科学与技术学院

2020/5/10

# 整数线性规划

## Chap. 6 Integer Linear Programming



# 主要内容

- 6.1 整数线性规划问题的提出
- 6.2 分支定界解法
- 6.3 割平面解法
- 6.4 0-1型整数线性规划
- **6.5 指派问题**
- **6.6 使用计算机工具求解本章问题**

## 6.5 指派问题

- 某单位需完成 $n$ 项任务，恰好有 $n$ 个人可承担这些任务，由于每人的专长不同，各人完成任务不同（或所费时间），效率不同。于是产生应指派哪个人去完成哪项任务，使完成 $n$ 项任务的总效率最高（或所需总时间最小）的问题。这类问题称为**指派问题（Assignment Problem）**。
- 例7 有一份中文说明书，需译成英、日、德、俄四种文字，分别记作E、J、G、R。现有甲乙丙丁四人，他们将说明书翻译成不同语种的所需时间如表中所示。问应如何指派使所需总时间最少？

人员 \ 任务	E	J	G	R
甲	2	15	13	4
乙	10	4	14	15
丙	9	14	16	13
丁	7	8	11	9

## 6.5 指派问题

- 上例可推广为：有  $n$  项加工任务到  $n$  台机床上的指派、 $n$  条航线由  $n$  艘船来航行……等问题。对应每个指派问题，类似上表那样的数据表，称为 **效率矩阵** 或 **系数矩阵**，其元素  $c_{ij} > 0$  ( $i, j = 1, 2, \dots, n$ ) 表示指派第  $i$  人去完成第  $j$  项任务时的效率（或时间、成本等）。解题时需引入 **0-1 变量**  $x_{ij}$ ，令：

$$x_{ij} = \begin{cases} 1 & \text{当指派第 } i \text{ 人去完成第 } j \text{ 项任务} \\ 0 & \text{当不指派第 } i \text{ 人去完成第 } j \text{ 项任务} \end{cases}$$

- 问题要求 **极小化** 时的数学模型为：

$$\min z = \sum_i \sum_j c_{ij} x_{ij}$$

$$\sum_i x_{ij} = 1, \quad j = 1, 2, \dots, n$$

表明第  $j$  项任务只能由 1 人完成

$$\sum_j x_{ij} = 1, \quad i = 1, 2, \dots, n$$

表明第  $i$  人只能完成 1 项任务

$$x_{ij} = 1 \text{ 或 } 0$$

## 6.5 指派问题

- 满足上述约束条件的可行解 $x_{ij}$ 可写成表格或矩阵形式，称为解矩阵。
- 解矩阵 $(x_{ij})$ 中各行各列的元素之和都是1。
- 例7的一个可行解矩阵是：

$$x_{ij} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- 该解矩阵不是最优解。
- 指派问题是0—1规划的特例，也是运输问题的特例；  
即： $n=m$ ,  $a_j = b_i = 1$  的特殊情形。
- 可用整数规划，0-1规划或运输问题的解法去求解，但这就如同用单纯形法求解运输问题一样是不合算的。
- 利用指派问题的特点可设计更简便高效的解法。

## 6.5 指派问题

- 指派问题的最优解有这样性质：若从系数矩阵 $(c_{i,j})$ 的一行(列)各元素中分别减去该行(列)的最小元素，得到新矩阵 $(b_{i,j})$ 矩阵,那么以 $(b_{i,j})$ 为系数矩阵求得的最优解和用原系数矩阵求得的最优解相同。
- 利用这个性质，可使原系数矩阵变换为含有很多0元素的新系数矩阵，而最优解保持不变；在系数矩阵 $(b_{i,j})$ 中，位于不同行不同列的0元素，简称为独立的0元素。
- 若能在系数矩阵 $(b_{i,j})$ 中找出n个独立的0元素，则令解矩阵 $(x_{i,j})$ 中对应这n个独立0元素的元素取值为1，其他元素取值为0。
- 将其代入目标函数中得到  $z_b=0$ ，它一定是最小，这就是以 $(b_{i,j})$ 为系数矩阵的指派问题的最优解，也就得到了原问题的最优解。

# 匈牙利算法

- 匈牙利算法是一种能在多项式时间内求解指派问题的组合优化算法，由美国数学家哈罗德·库恩(**Harold Kuhn**)于**1955**年提出该算法。
  - 之所以被称作匈牙利算法，是因为算法很大程度上基于两位匈牙利数学家**Dénes König**和**Jenő Egerváry**的一个关于矩阵中**0**元素的定理：**系数矩阵中独立0元素的最多个数等于能覆盖所有0元素的最少直线数。**
- 詹姆士·芒克勒斯(**James Munkres**)在**1957**年重新审视了这个方法，证明发现该方法是严格 **polynomial** 的，此后该算法被称为 **Kuhn-Munkres** 算法。
- 原始算法的时间复杂度为  **$O(n^4)$** ，之后 **Edmonds** 和 **Karp**，以及 **Tomizawa** 独立地发现可以修改算法达到  **$O(n^3)$**  时间复杂度。
- **Ford** 和 **Fulkerson** 将该方法推广到了一般运输问题。
- **2006** 年发现卡尔·雅可比在**19** 世纪就解决了指派问题，其解法在他逝世后于**1890** 年以拉丁文发表。



- 匈牙利算法的两个基本定理：

- 【定理1】 假设问题求最小值， $m$ 个人恰好做 $m$ 项工作，第 $i$ 个人做第 $j$ 项工作的效率为 $c_{ij}$ ，效率矩阵为 $[c_{ij}]$ 。如果从指派问题效率矩阵 $[c_{ij}]$ 的每一行元素中分别减去（或加上）一个常数 $u_i$ （被称为该行的位势），从每一列分别减去（或加上）一个常数 $v_j$ （称为该列的位势），得到一个新的效率矩阵 $[b_{ij}]$ ，其中 $b_{ij}=c_{ij}-u_i-v_j$ ，则 $[b_{ij}]$ 的最优解等价于 $[c_{ij}]$ 的最优解。这里 $c_{ij}$ 、 $b_{ij}$ 均非负。
- 【定理2】 若矩阵 $A$ 的元素可分成“0”与非“0”两部分，则覆盖“0”元素的最少直线数等于位于不同行不同列的“0”元素（称为独立0元素）的最大个数。—— D. Konig

- 定理1的证明:

$$\begin{aligned}
 z' &= \sum_{\substack{i=1 \\ i \neq k}}^n \sum_{j=1}^n c_{ij} x_{ij} + \sum_{j=1}^n (c_{kj} \pm s) x_{kj} \\
 &= \sum_{\substack{i=1 \\ i \neq k}}^n \sum_{j=1}^n c_{ij} x_{ij} + \sum_{j=1}^n c_{kj} x_{kj} + (\pm s) \sum_{j=1}^n x_{kj} \\
 &= \sum_{\substack{i=1 \\ i \neq k}}^n \sum_{j=1}^n c_{ij} x_{ij} + \sum_{j=1}^n c_{kj} x_{kj} + (\pm s) \\
 &= z + (\pm s)
 \end{aligned}$$

# 匈牙利算法

- 定理1告诉我们如何将效率矩阵中的元素转换为有零元素，定理2告诉我们效率矩阵中有多少个独立的“0”元素。
- 若从系数矩阵 $(c_{ij})$ 的一行（列）各元素中分别减去该行（列）的最小元素，得到新矩阵 $(b_{ij})$ ，那么以 $(b_{ij})$ 为系数矩阵求得的最优解和用原系数矩阵求得的最优解相同。因此，可使原系数矩阵变换为含有很多0元素的新系数矩阵，而最优解保持不变。
- 若能在 $(b_{ij})$ 中找出 $n$ 个独立的0元素；则令解矩阵 $(x_{ij})$ 中对应这 $n$ 个独立的0元素的元素取值为1，其他元素取值为0。将其代入目标函数中得到了 $z_b=0$ ，它一定是最小。这就是以为 $(b_{ij})$ 系数矩阵的指派问题的最优解，即原问题的最优解。
- 如果最少直线数等于 $n$ ，则存在 $n$ 个独立的“0”元素，令这些0元素对应的 $x_{ij}$ 等于1，其余变量等于0，得到最优解。

# 指派问题例子

— 以下用例7说明指派问题的匈牙利解法。

— 解：第一步：使指派问题的系数矩阵经变换，在各行各列中都出现0元素。

$$(c_{ij}) = \begin{bmatrix} 2 & 15 & 13 & 4 \\ 10 & 4 & 14 & 15 \\ 9 & 14 & 16 & 13 \\ 7 & 8 & 11 & 9 \end{bmatrix} \xrightarrow[\text{最小元素}]{\text{减去每行}} \begin{bmatrix} 0 & 13 & 11 & 2 \\ 6 & 0 & 10 & 11 \\ 0 & 5 & 7 & 4 \\ 0 & 1 & 4 & 2 \end{bmatrix} \xrightarrow[\text{最小元素}]{\text{减去每列}} \begin{bmatrix} 0 & 13 & 7 & 0 \\ 6 & 0 & 6 & 9 \\ 0 & 5 & 3 & 2 \\ 0 & 1 & 0 & 0 \end{bmatrix} = (b_{ij})$$

— 第二步：进行试指派，以寻求最优解。

- (1) 从只有一个0元素的行（列）开始，给这个0元素加圈，记作⊙。这表示对这行所代表的人，只有一种任务可以指派。然后划去⊙所在行（列）的其他0元素，记作∅，表示这列所代表的任务已经完成，不必再考虑其他人；
- (2) 给只有一个0元素的列（行）的0元素加圈，然后划去所在行的0元素；
- (3) 反复进行(1)和(2)，直到所有的0元素都被画圈或者划掉。

# 指派问题例子

- (4) 若仍没有画圈的0元素，且同行（列）的0元素至少有两个（表示对这人可以两项任务中指派其一），则可用不同的方案试探。从剩余0元素最少的行（列）开始，比较这行各0元素所在列中0元素的数目，选择最少的画圈（表示选择性多的要礼让选择性少的），然后划掉同行同列的0元素，反复进行；
- (5) 若◎元素的数目m等于矩阵的阶数n，那么指派问题的最优解已得到；若m<n则转入第三步。

$$\begin{bmatrix} 0 & 13 & 7 & 0 \\ 6 & 0 & 6 & 9 \\ 0 & 5 & 3 & 2 \\ 0 & 1 & 0 & 0 \end{bmatrix} \xrightarrow{\text{(a) 给 } b_{22}, b_{31} \text{ 加圈, 划掉所在列的0元素}} \begin{bmatrix} \phi & 13 & 7 & 0 \\ 6 & \odot & 6 & 9 \\ \odot & 5 & 3 & 2 \\ \phi & 1 & 0 & 0 \end{bmatrix}$$

(b) 给  $b_{43}$  加圈，划掉所在行的0元素

$$\begin{bmatrix} \phi & 13 & 7 & 0 \\ 6 & \odot & 6 & 9 \\ \odot & 5 & 3 & 2 \\ \phi & 1 & \odot & \phi \end{bmatrix} \xrightarrow{\text{(c) 最后给 } b_{14} \text{ 加圈}} \begin{bmatrix} \phi & 13 & 7 & \odot \\ 6 & \odot & 6 & 9 \\ \odot & 5 & 3 & 2 \\ \phi & 1 & \odot & \phi \end{bmatrix}$$

m=n=4，所以最优解为：

$$(x_{ij}) = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$



# 指派问题例子

— 第三步：作最少的直线覆盖所有的0元素，以确定该系数矩阵中能找到最多的独立0元素数。按下列步骤进行：

- (1)对没有 $\odot$ 的行打 $\checkmark$ 号；
  - (2)对已打 $\checkmark$ 号的行中所有含 $\Phi$ 元素的列打 $\checkmark$ 号；
  - (3)再对打有 $\checkmark$ 号的列中含 $\odot$ 元素的行打 $\checkmark$ 号；
  - (4)重复(2)、(3)，直到得不出新的打 $\checkmark$ 号的行、列为止；
  - (5)对没有打 $\checkmark$ 号的行画一横线，有打 $\checkmark$ 号的列画一纵线，这就得到覆盖所有0元素的最少直线数。
- 令直线数为 $l$ ：  
若 $l < n$ ，说明必须再变换当前的系数矩阵才能找到 $n$ 个独立的0元素，转第四步；  
若 $l = n$ ，而 $m < n$ ，应回到第二步(4)，另行试探。

# 指派问题例子

— 例8 求下表所示效率矩阵的指派问题的最小解。

人员 \ 任务	A	B	C	D	E
甲	12	7	9	7	9
乙	8	9	6	6	6
丙	7	17	12	14	9
丁	15	14	6	6	10
戊	4	10	7	10	9

— 解：

第一步变换系数矩阵：

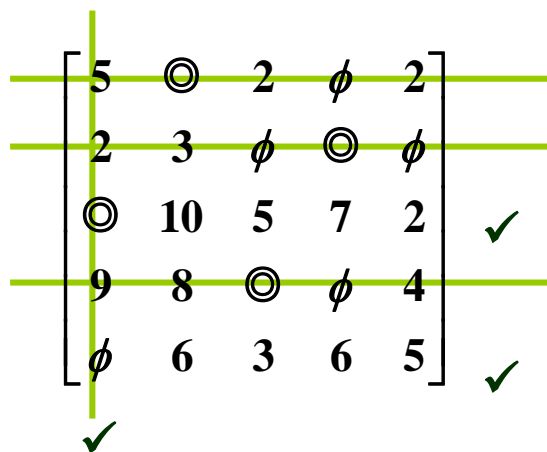
$$\begin{bmatrix} 12 & 7 & 9 & 7 & 9 \\ 8 & 9 & 6 & 6 & 6 \\ 7 & 17 & 12 & 14 & 9 \\ 15 & 14 & 6 & 6 & 10 \\ 4 & 10 & 7 & 10 & 9 \end{bmatrix} \rightarrow \begin{bmatrix} 5 & 0 & 2 & 0 & 2 \\ 2 & 3 & 0 & 0 & 0 \\ 0 & 10 & 5 & 7 & 2 \\ 9 & 8 & 0 & 0 & 4 \\ 0 & 6 & 3 & 6 & 5 \end{bmatrix}$$

# 指派问题例子

- 第二步进行试指派，得到：

$$\begin{bmatrix} 5 & \odot & 2 & \phi & 2 \\ 2 & 3 & \phi & \odot & \phi \\ \odot & 10 & 5 & 7 & 2 \\ 9 & 8 & \odot & \phi & 4 \\ \phi & 6 & 3 & 6 & 5 \end{bmatrix}$$

- 可知 $m=4, n=5$ ，所以转入第三步；
- 第三步做最少直线覆盖可得：



The matrix is shown with a vertical green line through the first column and four horizontal green lines through the first, second, fourth, and fifth rows. The third row is not covered by any line. Checkmarks are placed at the bottom of the first column, to the right of the third row, and to the right of the fifth row.

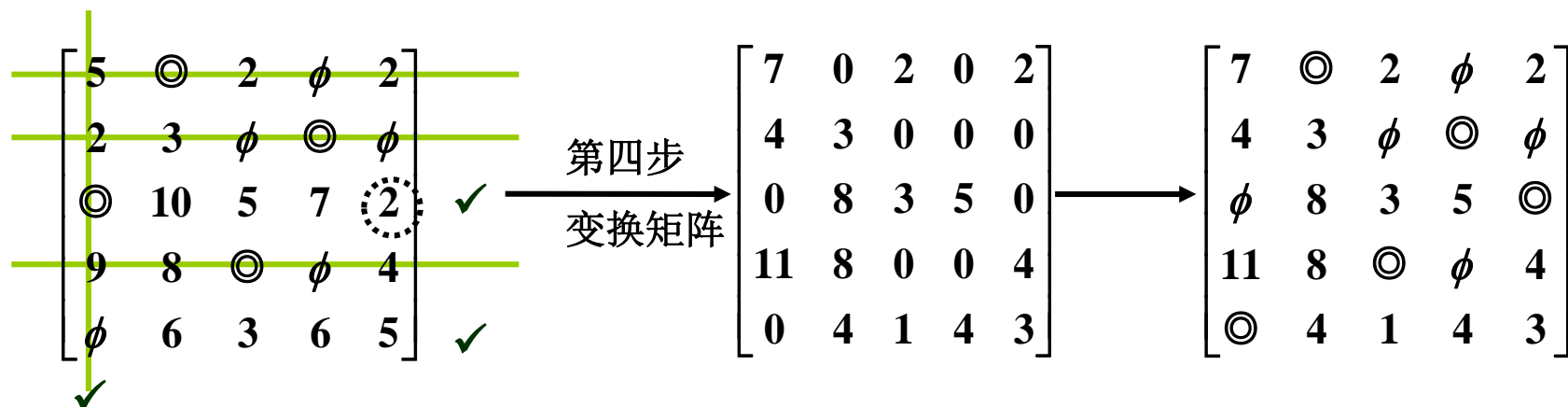
$$\begin{bmatrix} 5 & \odot & 2 & \phi & 2 \\ 2 & 3 & \phi & \odot & \phi \\ \odot & 10 & 5 & 7 & 2 \\ 9 & 8 & \odot & \phi & 4 \\ \phi & 6 & 3 & 6 & 5 \end{bmatrix}$$



# 指派问题例子

- 可见  $l=4 < n$ ，所以应继续对矩阵进行变换，转入第四步；
- 第四步：对矩阵变换的目的是增加0元素。为此在没有被直线覆盖的部分中找出最小元素，然后在打✓号行的各元素中都减去这最小元素，而在打✓号列的各元素都加上这最小元素，以保证原来0元素不变。这样得到的新系数矩阵（它的最优解和原问题相同），若得到n个独立0元素，则已得到最优解，否则回到第三步重复进行。
- 上例6.9中，继续进行第四步，有：

$$\begin{array}{c}
 \begin{bmatrix} 5 & \odot & 2 & \phi & 2 \\ 2 & 3 & \phi & \odot & \phi \\ \odot & 10 & 5 & 7 & 2 \\ 9 & 8 & \odot & \phi & 4 \\ \phi & 6 & 3 & 6 & 5 \end{bmatrix}
 \end{array}
 \xrightarrow[\text{变换矩阵}]{\text{第四步}}
 \begin{bmatrix} 7 & 0 & 2 & 0 & 2 \\ 4 & 3 & 0 & 0 & 0 \\ 0 & 8 & 3 & 5 & 0 \\ 11 & 8 & 0 & 0 & 4 \\ 0 & 4 & 1 & 4 & 3 \end{bmatrix}
 \longrightarrow
 \begin{bmatrix} 7 & \odot & 2 & \phi & 2 \\ 4 & 3 & \phi & \odot & \phi \\ \phi & 8 & 3 & 5 & \odot \\ 11 & 8 & \odot & \phi & 4 \\ \odot & 4 & 1 & 4 & 3 \end{bmatrix}$$



# 指派问题例子

- 可见具有**n=5**个独立**0**元素，即得到最优解：

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

- 最优指派方案为：甲—**B**，乙—**D**，丙—**E**，丁—**C**，戊—**A**
- 本例还有另一最优指派方案：甲—**B**，乙—**C**，丙—**E**，丁—**D**，戊—**A**
- 当指派问题的系数矩阵，经过变换得到了同行和同列中都有两个或两个以上**0**元素时，这时可以任选一行（列）中的某个**0**元素，再划去同行（列）的其他**0**元素，这时会出现**多重解**（如上例）。

# 指派问题的变异

- 指派问题的变异
  - 对于极大化问题:

$$\max z = \sum_i \sum_j c_{ij} x_{ij}$$

可令:  $b_{ij} = M - c_{ij}$ , 其中  $M$  是足够大的常数 (如选  $c_{ij}$  中最大元素为  $M$  即可), 这时系数矩阵可变为  $B = (b_{ij})$ , 这时  $b_{ij} \geq 0$ , 符合匈牙利法的条件。

目标函数经变换后, 即解  $\min z' = \sum_i \sum_j b_{ij} x_{ij}$

所得的最小解就是原问题的最大解。

因为:  $\sum_i \sum_j b_{ij} x_{ij} = \sum_i \sum_j (M - c_{ij}) x_{ij} = \sum_i \sum_j M x_{ij} - \sum_i \sum_j c_{ij} x_{ij} = nM - \sum_i \sum_j c_{ij} x_{ij}$

$nM$  为常数, 所以当  $\sum_i \sum_j b_{ij} x_{ij}$  取最小时,  $\sum_i \sum_j c_{ij} x_{ij}$  便最大。

# 指派问题的变异

## — 人数与任务数不均等问题

- 设分配问题中人数为 $m$ ，工作数为 $n$ ，当 $m > n$ 时，虚拟 $m-n$ 项工作，对应的效率为零；当 $m < n$ 时，虚拟 $n-m$ 个人，对应的效率为零。通过上述步骤化为人数与任务数相等的平衡问题后再求解。

## — 不可接受的配置问题

- 当某人不能完成某项工作时，令对应的效率为一个大大 $M$ 即可。



## 6.6 使用计算机工具求解本章问题

---

- 6.6.1 使用编程语言
- 6.6.2 使用Excel
- 6.6.3 使用Matlab

- 使用编程语言：（将算法用编程语言实现）
- 指派问题：
- 输入：效率矩阵（人数=任务）
- 输出：最优方案
- **Java**为例：

```
decrease(ma); // 每行减去最小值
decrease2(ma); // 每列减去最小值
int[][] best = new int[n][n];
// 初始化最优解
while (true) {
    if (findbest(ma, best)) {
        // 查找矩阵最优值，能找到则返回true
        break; // 计算结束
    }
    transformmatrix(ma);
    // 增加0值的个数
}
```

指派问题的Excel实现,例7（与运输问题类似）

人员 \ 任务	E	J	G	R			
甲	2	15	13	4			
乙	10	4	14	15		总时间	
丙	9	14	16	13		0	
丁	7	8	11	9			
人员 \ 任务	E	J	G	R	实际人员		最多一人
甲	0	0	0	0	0 =		1
乙	0	0	0	0	0 =		1
丙	0	0	0	0	0 =		1
丁	0	0	0	0	0 =		1
实际任务	0	0	0	0			
	=	=	=	=			
最多一个任务	1	1	1	1			



# 使用Excel

规划求解规则设定：  
总时间=SUMPRODUCT  
(C4:F7,C10:F13)

The image shows the 'Solver Parameters' dialog box in Microsoft Excel. The 'Set Objective' field is set to '总时间' (Total Time). The 'To' section has 'Min' (Minimum) selected. The 'By Changing Variable Cells' field is set to '人员安排' (Personnel Arrangement). The 'Subject to the Constraints' list contains three constraints: '人员安排 = binary', '实际人员 = 最多一人', and '实际任务 = 最多一个任务'. The 'Make Unconstrained Variables Non-Negative' checkbox is checked. The 'Select a Solving Method' dropdown is set to 'Simplex LP'. The 'Solving Method' section provides instructions on selecting the appropriate engine for different problem types. At the bottom, there are buttons for 'Help', 'Solve', and 'Close'.





# 使用Excel

规划求解结果：

人员 \ 任务	E	J	G	R
甲	0	0	0	1
乙	0	1	0	0
丙	1	0	0	0
丁	0	0	1	0

- MATLAB**

- 例7 有一份中文说明书，需译成英、日、德、俄四种文字，分别记作**E**、**J**、**G**、**R**。现有甲乙丙丁四人，他们将说明书翻译成不同语种的所需时间如表中所示。问应如何指派使所需总时间最少？

人员 \ 任务	<b>E</b>	<b>J</b>	<b>G</b>	<b>R</b>
甲	<b>2</b>	<b>15</b>	<b>13</b>	<b>4</b>
乙	<b>10</b>	<b>4</b>	<b>14</b>	<b>15</b>
丙	<b>9</b>	<b>14</b>	<b>16</b>	<b>13</b>
丁	<b>7</b>	<b>8</b>	<b>11</b>	<b>9</b>



## 指派问题的MATLAB实现

```
%适用于任意n阶系数矩阵
clear all;
C=[2 15 13 4;10 4 14 15;9 14 16 13;7 8 11 9];%效率矩阵C
n=size(C,1);%计算C的行列数n
C=C(:);%计算目标函数系数，将矩阵C按列排成一个列向量即可。
A=[];B=[];%没有不等式约束
Ae=zeros(2*n,n^2);%计算等约束的系数矩阵a
for i=1:n
    for j=(i-1)*n+1:n*i
        Ae(i,j)=1;
    end
    for k=i:n:n^2
        Ae(n+i,k)=1;
    end
end
```



## 指派问题的MATLAB实现

```
end
Be=ones(2*n,1);%等式约束右端项b
Xm=zeros(n^2,1);%决策变量下界Xm
XM=ones(n^2,1);%决策变量上界XM
[x,z]=linprog(C,A,B,Ae,Be,Xm,XM);%使用linprog求解
x=reshape(x,n,n);%将列向量x按列排成一个n阶方阵
disp('最优解矩阵为:');%输出指派方案和最优值
Assignment=round(x)%使用round进行四舍五入取整
disp('最优解为:');
z
```

28.0000
---------



本章完  
*The end*