

第三十二章 作业计划

排序的理论与方法是编制车间作业计划的基础。本章将介绍排序问题的基本概念，重点介绍流水作业排序问题。

§1 流水车间调度问题

1.1 排序问题的基本概念

1.1.1 名词术语

在生产管理中，常用到“编制作业计划”（Scheduling）、“排序”（Sequencing）、“派工”（Dispatching）、“控制”（Controlling）和“赶工”（Expediting）这些名词。

一般说来，编制作业计划与排序不是同义语。排序只是确定工件在机器上的加工顺序。而编制作业计划，则不仅包括确定工件的加工顺序，而且还包括确定机器加工每个工件的开始时间和完成时间。因此，只有作业计划才能指导每个工人的生产活动。由于编制作业计划的主要问题是确定各台机器上工件的加工顺序，而且，在通常情况下都是按最早可能开(完)工时间来编排作业计划的。因此，当工件的加工顺序确定之后，作业计划也就确定了。所以，人们常常不加区别地使用“排序”与“编制作业计划”这两个术语。在本章里，讲排序的时候就暗指相应的作业计划是最早时间作业计划。只有在需要的时候，才将这两个术语区别使用。

“派工”是按作业计划的要求，将具体生产任务安排到具体的机床上加工，属于我们经常说的“调度”范围。“赶工”是在实际进度已落后于计划进度时采取的行动，也属于“调度”范围。“调度”是实行控制所采取的行动。“编制作业计划”是加工制造发生之前的活动；“调度”是在加工制造发生之后的活动。调度的依据是作业计划。比如，火车时刻表是一种作业计划，火车发生晚点，就要进行调度。调度是一种现场指挥。

描述排序问题的名词术语来自加工制造行业。为了和惯用的名词术语保持一致，本书仍使用“机器”、“工件”、“工序”和“加工时间”等术语来描述各种不同的排序问题。但要注意，它们已不限于本来的含义。这里所说的“机器”，可以是工厂里的各种机床，也可以是维修工人；可以是轮船要停靠的码头，也可以是电子计算机中央处理单元、存储器 and 输入、输出单元。一句话，表示“服务者”；工件则代表“服务对象”。工件可以是单个零件，也可以是一批相同的零件。

假定有 n 个工件要经过 m 台机器加工。“加工路线”是工件加工的工艺过程决定的，它是工件加工在技术上的约束。比如，某工件要经过车、铣、占、磨的路线加工，我们可以用 M_1, M_2, M_3, M_4 来表示。一般地，可用 M_1, M_2, \dots, M_m 来表示加工路线。

“加工顺序”则表示每台机器加工 n 个工件的先后顺序，是排序要解决的问题。

1.1.2 假设条件与符号说明

为了便于分析研究，建立数学模型，有必要对排序问题提出一些假设条件。

- ① 一个工件不能同时在几台不同的机器上加工。
- ② 工件在加工过程中采取平行移动方式，即当上一道工序完工后，立即送下道工序加工。
- ③ 不允许中断。当一个工件一旦开始加工，必须一直进行到完工，不得中途停止插入其它工件。
- ④ 每道工序只在一台机器上完成。
- ⑤ 工件数、机器数和加工时间已知，加工时间与加工顺序无关。
- ⑥ 每台机器一次只能加工一个工件。

在下面的讨论中，如不作特别说明，都是遵循以上假设条件的。

下面对有关符号进行说明。

J_i : 工件 i , $i = 1, 2, \dots, n$;

M_j : 机器 j , $j = 1, 2, \dots, m$;

t_{ij} : J_i 在 M_j 上的加工时间, J_i 的总加工时间为 $T_i = \sum_{j=1}^m t_{ij}$;

h_{ij} : J_i 在 M_j 上的等待时间, J_i 的总等待时间为 $H_i = \sum_{j=1}^m h_{ij}$;

r_i : J_i 的到达时间, 指 J_i 从外部进入车间, 可以开始加工的最早时间;

d_i : J_i 的完工期限;

c_{ij} : J_i 在 M_j 上的完工时间;

C_i : J_i 的完工时间, $C_i = r_i + \sum_{j=1}^m (h_{ij} + t_{ij}) = r_i + H_i + T_i$;

C_{\max} : 最长完工时间, $C_{\max} = \max_{1 \leq i \leq n} \{C_i\}$;

F_i : J_i 的流程时间, 即工件在车间的实际停留时间, $F_i = C_i - r_i = H_i + T_i$;

F_{\max} : 最长流程时间, $F_{\max} = \max_{1 \leq i \leq n} \{F_i\}$;

L_i : 工件的延迟时间, $L_i = C_i - d_i$; 当 $L_i > 0$ (正延迟), 说明 J_i 的实际完工时间超过了完工期限; 当 $L_i < 0$ (负延迟), 说明 J_i 提前完工; 当 $L_i = 0$ (零延迟), J_i 按期完工。

L_{\max} : 最长延迟时间, $L_{\max} = \max_{1 \leq i \leq n} \{L_i\}$ 。

1.1.3 排序问题的分类和表示法

排序问题有不同的分类方法。最常用的分类方法是按机器、工件和目标函数的特征分类。按机器的种类和数量不同, 可以分成单台机器的排序问题和多台机器的排序问题。对于多台机器的排序问题, 按工件加工路线的特征, 可以分成单件作业 (Job-shop) 排序问题和流水作业 (Flow-shop) 排序问题。工件的加工路线不同, 是单件作业排序问题的基本特征; 而所有工件的加工路线完全相同, 则是流水作业排序问题的基本特征。

按工件到达车间的情况不同, 可以分成静态的排序问题和动态的排序问题。当进行排序时, 所有工件都已到达, 可以一次对它们进行排序, 这是静态的排序问题; 若工件是陆续到达, 要随时安排它们的加工顺序, 这是动态的排序问题。

按目标函数的性质不同, 也可划分不同的排序问题。譬如, 同是单台机器的排序, 目标是使平均流程时间最短和目标是使误期完工工件数最少, 实质上是两种不同的排序问题。按目标函数的情况, 还可以划分为单目标排序问题与多目标排序问题。以往研究的排序问题, 大都属于单目标排序问题, 而对多目标排序问题则很少研究。

另外, 按参数的性质, 可以划分为确定型排序问题与随机型排序问题。所谓确定型排序问题, 指加工时间和其它有关参数是已知确定的量; 而随机型排序问题的加工时间和有关参数为随机变量。这两种排序问题的解法本质上不同。

由机器、工件和目标函数的不同特征以及其它因素上的差别, 构成了多种多样的排序问题。对于本节要讨论的排序问题, 我们将用 Conway 等人提出的方法来表示。这个

方法只用 4 个参数就可以表示大多数不同的排序问题。4 参数表示法为：

$$n/m/A/B$$

其中， n —工件数；

m —机器数；

A —车间类型，在 A 的位置若标以“ F ”，则代表流水作业排序问题。若标以“ P ”，则表示流水作业排列排序问题。若标以“ G ”，则表示一般单件作业排序问题。当 $m=1$ ，则 A 处为空白。因为对于单台机器的排序问题来说，无所谓加工路线问题，当然也就谈不上是流水作业还是单件作业的问题了。

B —目标函数，通常是使其值最小。

有了这 4 个符号，就可以简明地表示不同的排序问题。例如， $n/3/P/C_{\max}$ 表示 n 个工件经 3 台机器加工的流水作业排列排序问题，目标函数是使最长完工时间 C_{\max} 最短。

1.2 流水作业排序问题

流水作业排序问题的基本特征是每个工件的加工路线都一致。在流水生产线上制造不同的零件，遇到的就是流水作业排序问题。我们说加工路线一致，是指工件的流向一致，并不要求每个工件必须经过加工路线上每台机器加工。如果某些工件不经某些机器加工，则设相应的加工时间为零。

一般说来，对于流水作业排序问题，工件在不同机器上的加工顺序不尽一致。但本节要讨论的是一种特殊情况，即所有工件在各台机器上的加工顺序都相同的情况。这就是排列排序问题。流水作业排列排序问题常被称作“同顺序”排序问题。对于一般情形，排列排序问题的最优解不一定是相应的流水作业排序问题的最优解，但一般是比较好的解；对于仅有 2 台和 3 台机器的特殊情况，可以证明，排列排序问题下的最优解一定是相应流水作业排序问题的最优解。

本节只讨论排列排序问题。

1.2.1 最长流程时间 F_{\max} 的计算

本节所讨论的是 $n/m/P/F_{\max}$ 问题，目标函数是使最长流程时间最短。最长流程时间又称作加工周期，它是从第一个工件在第一台机器开始加工时算起，到最后一个工件在最后一台机器上完成加工时为止所经过的时间。由于假设所有工件的到达时间都为零（ $r_i = 0, i = 1, 2, \dots, n$ ），所以 F_{\max} 等于排在末位加工的工件在车间的停留时间，也等于一批工件的最长完工时间 C_{\max} 。

设 n 个工件的加工顺序为 $S = \{s_1, s_2, \dots, s_n\}$ ，其中 s_i 为排第 i 位加工的工件的代号。以 $c_{s_i, k}$ 表示工件 s_i 在机器 M_k 上的完工时间， $t_{s_i, k}$ 表工件 s_i 在 M_k 上的加工时间， $i = 1, 2, \dots, n$ ； $k = 1, 2, \dots, m$ ，则 $c_{s_i, k}$ 可按以下公式计算：

$$\begin{aligned} c_{s_1, 1} &= t_{s_1, 1} \\ c_{s_1, k} &= c_{s_1, k-1} + t_{s_1, k}, \quad k = 2, \dots, m \end{aligned} \quad (1)$$

$$c_{s_i, 1} = c_{s_{i-1}, 1} + t_{s_i, 1}, \quad i = 2, \dots, n \quad (2)$$

$$c_{s_i, k} = \max\{c_{s_{i-1}, k}, c_{s_i, k-1}\} + t_{s_i, k}, \quad i = 2, \dots, n, \quad k = 2, \dots, m \quad (3)$$

当 $r_i = 0, i = 1, 2, \dots, n$ 时，最大流程时间为

$$F_{\max} = c_{s_n m} \quad (4)$$

当由式 (3) 得出 $c_{s_n m}$ 时, F_{\max} 就求得了。

在熟悉以上计算公式之后, 可直接在加工时间矩阵上从左到右计算完工时间。下面以一例说明。

例 1 有一个 $6/4/P/F_{\max}$ 问题, 其加工时间如表 1 所示。当按顺序 $S = (6, 1, 5, 2, 4, 3)$ 加工时, 求 F_{\max} 。

表 1 加工时间矩阵

i	1	2	3	4	5	6
t_{i1}	4	2	3	1	4	2
t_{i2}	4	5	6	7	4	5
t_{i3}	5	8	7	5	5	5
t_{i4}	4	2	4	3	3	1

解: 按顺序 $S = (6, 1, 5, 2, 4, 3)$ 列出加工时间矩阵, 如表 2 所示。按式 (1) ~ (3) 进行递推, 将每个工件的完工时间标在其加工时间的后面。对于第一行第一列, 只需把加工时间的数值作为完工时间标在加工时间的后面。对于第一行的其它元素, 使用迭代公式 (1), 只需从左到右依次将前一列后面的数字加上计算列的加工时间, 将结果填在计算列加工时间的后面。对于从第二行到第 m 行, 第一列的算法相同, 使用迭代公式 (2), 只要把上一行后面的数字和本行的加工时间相加, 将结果填在本行加工时间的后面; 从第 2 列到第 n 列, 使用迭代公式 (3), 则要从本行前一列后面数字和本列上一行后面数字中取大者, 再和本列加工时间相加, 将结果填在本列加工时间的后面。这样计算下去, 最后一行最后一列的后面数字, 即为 $c_{s_n m}$, 也是 F_{\max} 。计算结果如表 2 所示。

本例 $F_{\max} = 46$ 。

表 2 顺序 S 下的加工时间矩阵

i	6	1	5	2	4	3
t_{i1}	2, 2	4, 6	4, 10	2, 12	1, 13	3, 16
t_{i2}	5, 7	4, 11	4, 15	5, 20	7, 27	6, 33
t_{i3}	5, 12	5, 17	5, 22	8, 30	5, 35	7, 42
t_{i4}	1, 13	4, 21	3, 25	2, 32	3, 38	4, 46

计算的 Matlab 程序如下:

```
clc, clear
t=[4    2    3    1    4    2
   4    5    6    7    4    5
   5    8    7    5    5    5
   4    2    4    3    3    1]';
s=[6 1 5 2 4 3];
st=t(s, :); %提出指定顺序的时间矩阵
[n, m]=size(st);
c(1, :)=cumsum(st(1, :)); %计算 c 的第一行
c(2:n, 1)=c(1, 1)+cumsum(st(2:n, 1)); %计算 c 的第一列的除第 1 个元素外其它元素
```

```

for i=2:n
    for k=2:m
        c(i,k)=max(c(i-1,k),c(i,k-1))+st(i,k);
    end
end
ds=c' %显示计算结果
Fmax=ds(end) %显示最后一个元素

```

1.2.2 $n/2/F/F_{\max}$ 问题的最优算法

对于以最大流程时间为目标的两台机器流水车间问题称为 Johnson 问题,其最优调度由著名的 Johnson 规则确定。Johnson 规则的基本思想为后来 m 台机器问题的启发式算法提供了基础。

假定有 n 个工件,在第一台机器和第二台机器上的加工时间分别为 t_{i1} , $i=1,\dots,n$ 和 t_{i2} , $i=1,\dots,n$, 最优调度由如下规则给出:

Johnson 规则: 如果 $\min\{t_{i1}, t_{j2}\} \leq \min\{t_{i2}, t_{j1}\}$, 则最优调度中工件 i 排在工件 j 之前。

最优顺序可以直接利用 Johnson 规则通过两个工序之间的检查来构造。Johnson 算法可以描述为:

步骤 1: 令 $U = \{i | t_{i1} \leq t_{i2}\}$, $V = \{i | t_{i1} > t_{i2}\}$;

步骤 2: 对 U 中的工件按 t_{i1} 非减顺序调度;

步骤 3: 对 V 中的工件按 t_{i2} 非增顺序调度;

步骤 4: 有序集 U 放到 V 之前构成工件的最优加工顺序。

例 2 求表 3 所示的 $8/2/F/F_{\max}$ 问题的最优解。

表 3 8 工件问题的实例

i	1	2	3	4	5	6	7	8
t_{i1}	5	2	1	7	6	3	7	5
t_{i2}	2	6	2	5	6	7	2	1

本例的解构造如下:

步骤 1: 工件集 $U = \{2,3,5,6\}$ 和 $V = \{1,4,7,8\}$;

步骤 2: U 中的工件调度为

工件 i : 3 2 6 5
 t_{i1} : 1 2 3 6

步骤 3: V 中的工件调度为

工件 i : 4 7 1 8
 t_{i1} : 5 2 2 1

步骤 4: 最优顺序为 (3,2,6,5,4,7,1,8)。

计算的 Matlab 程序如下:

```

clc,clear
t=[5    2    1    7    6    3    7    5

```

```

2    6    2    5    6    7    2    1]
n=size(t,2);
U=find(t(1,:)<=t(2,:)); %提出工件集 U
V=find(t(1,:)>t(2,:)); %提出工件集 V
tU=t(1,U);tV=t(2,V); %提出工件集 U, V 对应的时间
[stU, ind1]=sort(tU); %对时间按照从小到大排序
[stV, ind2]=sort(tV,'descend'); %对时间按照从大到小排序
UV=[U(ind1) V(ind2)] %计算最优顺序
st=t(:,UV) %最优顺序下的加工时间表
ti1=cumsum(st(1,:)); %求机器 1 各工件的加工结束时间
ti2(1)=ti1(1)+st(2,1); %求机器 2 的第一个工件的加工结束时间
for j=2:n
    ti2(j)=max(ti1(j),ti2(j-1))+st(2,j);
end
solu=[ti1;ti2] %显示机器 1 和机器 2 加工的各工件的结束时间

```

当从应用 Johnson 算法求得的最优顺序中任意去掉一些工件时,余下的工件仍构成最优顺序。同时,我们还要指出,Johnson 法则只是一个充分条件,不是必要条件。不符合这个法则的加工顺序,也可能是最优顺序。

1.2.3 一般 $n/m/P/F_{\max}$ 问题的启发式算法

对于 3 台机器的流水车间排序问题,只有几种特殊类型的问题找到了有效算法。例如若存在一个 $n/3/P/F_{\max}$ 问题,且 $t_{i1} \geq t_{i2}$ 或 $t_{i3} \geq t_{i2}$ ($i=1,2,\dots,n$),则可采用 Johnson 法排序。

求解步骤为:

- (1) 先找出 $t_{i1} \geq t_{i2}$ 或 $t_{i3} \geq t_{i2}$ ($i=1,2,\dots,n$) 关系;
- (2) 将 3 台设备变换成 2 台假想设备 M_A 和 M_B , 并令

$$t_{iA} = t_{i1} + t_{i2}, \quad t_{iB} = t_{i2} + t_{i3}$$

- (3) 依据 t_{iA} 和 t_{iB} , 采用 Johnson 法则进行作业排序;
- (4) 最大流程时间仍然按照三台设备计算。

多个工件在三台设备上的作业排序。若存在一个 $n/3/P/F_{\max}$ 问题,但不满足 $t_{i1} \geq t_{i2}$ 或 $t_{i3} \geq t_{i2}$ ($i=1,2,\dots,n$),则不可采用 Johnson 法排序。

对于一般的流水车间排列排序问题,可以用分支定界法。用分支定界法可以保证得到一般 $n/m/P/F_{\max}$ 问题的最优解。但对于实际生产中规模较大的问题,计算量相当大,以至连电子计算机也无法求解。同时,还需考虑经济性。如果为了求最优解付出的代价超过了这个最优解所带来的好处,也是不值得的。

为了解决生产实际中的排序问题,人们提出了各种启发式算法。启发式算法以小的计算量得到足够好的结果,因而十分实用。下面介绍求一般 $n/m/P/F_{\max}$ 问题近似最优解 (Near optimal solution) 的启发式算法。

(一) Palmer 启发式算法

Palmer 提出了基于工件的加工时间按斜度顺序指标 (slope order index) 排列工件的启发式算法,其基本思想是给每个工件赋优先权数。按机器的顺序,加工时间趋于

增加的工件得到较大的优先数；与此相反，按机器的顺序，加工时间趋于减少的工件得到较小的优先数。工件 i 的斜度指标（slope index） λ_i 按下式计算：

$$\lambda_i = \sum_{j=1}^m (2j - m - 1)t_{ij}, \quad i = 1, 2, \dots, n \quad (5)$$

按照各工件 λ_i 不增的顺序排列工件，可得出令人满意的顺序。

例3 有一个 $4/3/F/F_{\max}$ 问题，其加工时间如表5所示，用Palmer法求解。

表4 加工时间矩阵

i	1	2	3	4
t_{i1}	1	2	6	3
t_{i2}	8	4	2	9
t_{i3}	4	5	8	2

解 计算 λ_i 。对于本例，式（5）变成

$$\lambda_i = \sum_{j=1}^3 (2j - 4)t_{ij}, \quad i = 1, 2, 3, 4 \quad (6)$$

$$\lambda_i = -2t_{i1} + 2t_{i3}, \quad i = 1, 2, 3, 4$$

于是

$$\lambda_1 = 6, \quad \lambda_2 = 6, \quad \lambda_3 = 4, \quad \lambda_4 = -1$$

按 λ_i 不增的顺序排列工件，得到加工顺序（1，2，3，4）和（2，1，3，4），恰好这两个顺序都是最优顺序。如不是这样，则从中挑选较优者。在最优顺序下， $F_{\max} = 28$ 。

（二）Gupta启发式算法

Gupta提出了另一种类似Palmer方法的启发式算法。他考虑了对于三台机器问题的Johnson规则的最优性这一很有趣的问题，并用不同的方法定义斜度指标。按照他的定义，工件 i 的斜度指标 λ_i 定义为

$$\lambda_i = \frac{e_i}{\min_{1 \leq j \leq m-1} \{t_{ij} + t_{i,j+1}\}}, \quad i = 1, 2, \dots, n \quad (7)$$

其中

$$e_i = \begin{cases} 1, & t_{i1} < t_{im} \\ -1, & t_{i1} \geq t_{im} \end{cases}$$

然后，根据斜度指标（7）排列工件的加工顺序。

（三）关键工件法

其步骤如下：

（1）计算每个工件的总加工时间 $T_i = \sum_{j=1}^m t_{ij}$ ，找出加工时间最长的工件 J ，将其作为关键工件。

（2）对于余下的工件，若 $t_{i1} \leq t_{im}$ ，则按 t_{i1} 不减的顺序排成一个序列 s_u ；若 $t_{i1} > t_{im}$ ，

则按 t_{im} 不增的顺序排列成一个序列 s_v 。

(3) 顺序 (s_u, J, s_v) 即为所求顺序。

下面用关键工件法求例 3 的近似解。求 T_i , $i=1,2,3,4$, 如表 5 所示。总加工时间最长的为 3 号工件; $t_{i1} \leq t_{i3}$ 的工件为 1 和 2, 按 t_{i1} 不减的顺序排成 $s_u = (1,2)$; $t_{i1} > t_{i3}$ 的工件为 4 号工件, $s_v = (4)$, 这样得到的加工顺序为 $(1,2,3,4)$, 对本例为最优顺序。

表 5 用关键工件法求解

i	1	2	3	4
t_{i1}	1	2	6	3
t_{i2}	8	4	2	9
t_{i3}	4	5	8	2
T_i	13	11	16	14

(四) CDS 启发式算法

Campbell, Dudek 和 Smith 三人提出了一个启发式算法, 简称 CDS 法。他们把 Johnson 算法用于一般的 $n/m/P/F_{\max}$ 问题, 得到 $m-1$ 个加工顺序, 取其中优者。

算法首先把 m 台机器系统地组成两组, 产生 $m-1$ 个两台机器问题的集合, 然后利用 Johnson 的两台机器算法得到 $m-1$ 个加工顺序, 最后选择其中最好的一个作为近似最优解。第一阶段, 考虑由机器 1 和 m 形成的两台机器问题; 第二阶段, 考虑模拟的两台机器问题: 模拟机器 1 由机器组 $\{1, 2\}$ 组成, 模拟机器 2 由机器组 $\{m, m-1\}$ 组成; \dots ; 第 k 阶段, 考虑模拟的两台机器问题: 模拟机器 1 由机器组 $\{1, 2, \dots, k\}$ 组成, 模拟机器 2 由机器组 $\{m, \dots, m+1-k\}$ 组成; \dots 。

具体做法是, 对加工时间

$$\sum_{j=1}^k t_{ij} \text{ 和 } \sum_{j=1}^k t_{i, m+1-j}, \quad k=1, 2, \dots, m-1$$

用 Johnson 算法求 $m-1$ 次加工顺序, 取其中最好的结果。

现在我们对例 3 用 CDS 法求解。

首先, 求 $\sum_{j=1}^k t_{ij}$ 和 $\sum_{j=1}^k t_{i, m+1-j}$, $k=1, 2$, 结果如表 6 所示。

表 6 用 CDS 法求解

i		1	2	3	4
$k=1$	t_{i1}	1	2	6	3
	t_{i3}	4	5	8	2
$k=2$	$t_{i1} + t_{i2}$	9	6	8	12
	$t_{i2} + t_{i3}$	12	9	10	11

当 $k=1$ 时, 按 Johnson 算法得到加工顺序 $(1, 2, 3, 4)$, 相应的 $F_{\max} = 28$; 当 $k=2$ 时, 得到加工顺序 $(2, 3, 1, 4)$, 相应的 $F_{\max} = 29$ 。所以, 取顺序 $(1, 2, 3, 4)$ 。

我们已经知道，这就是最优顺序。

计算的 Matlab 程序如下（所有代码放在一个文件中）：

```
function example
clc
T=[1    2    6    3
   8    4    2    9
   4    5    8    2]';
[n,m]=size(T); %n 为工件的个数，m 为机器的台数
txm=[]; %存放所有 m-1 个两台虚拟机器问题的加工时间矩阵
Fmax=[]; %存放所有 m-1 个两台虚拟机器问题的流程时间
TUV=[]; %存放 m-1 个加工顺序
for k=1:m-1
    for i=1:n
        tx(i,1)=sum(T(i,1:k));
        tx(i,2)=sum(T(i,m+1-k:m));
    end
    [fmax,uv]=myjohnson(tx,T,n,m); %调用下面的 Johnson 算法函数
    Fmax=[Fmax,fmax];
    txm=[txm,tx];
    TUV=[TUV,uv];
end
txm,Fmax,TUV
optim_Fmax=min(Fmax) %近似最优流程时间
ind=find(Fmax==optim_Fmax);
optim_seq=TUV(:,ind) %近似最优加工顺序
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [f,UV]=myjohnson(tx,T,n,m);
U=find(tx(:,1)<=tx(:,2)); %提出工件集 U
V=find(tx(:,1)>tx(:,2)); %提出工件集 V
tU=tx(U,1);tV=tx(V,2); %提出工件集 U, V 对应的时间
[stU,ind1]=sort(tU); %对时间按照从小到大排序
[stV,ind2]=sort(tV,'descend'); %对时间按照从大到小排序
UV=[U(ind1);V(ind2)]; %计算最优顺序
st=T(UV,:); %最优顺序下的加工时间表
c(1,:)=cumsum(st(1,:)); %求工件 1 的加工结束时间
c(2:n,1)=c(1,1)+cumsum(st(2:n,1)); %求机器 1 的加工结束时间
for i=2:n
    for k=2:m
        c(i,k)=max(c(i-1,k),c(i,k-1))+st(i,k);
    end
end
end
f=c(end);
```

§ 2 作业车间调度问题

机器调度问题来源于不同的领域，如柔性制造系统，生产计划，计算机设计，后勤

及通信等，这些问题的共同特性是没有任何一个有效的算法能在多项式时间内求出其最优解。古典的作业车间调度问题是最著名的机器调度问题之一。问题可以描述为：给定一个工件的几何和一个机器的集合，每个工件包括多道工序，每道工序需要在一台给定的机器上非间断地加工某一段时间；每台机器一次最多只能加工一道工序；调度就是把工序分配给机器某个时间段。问题的目标是找到最小时间长度的调度。

作业车间调度问题（JSP）是最困难的组合最优化问题之一。由于其本身的难处理的特性，一些启发式算法成为有吸引力的备选方法。多数传统的启发式算法用优先权规则。所谓优先权规则是一个从未排序的工序特定子集中选用工序的规则。近年来，概率的局域搜索方法的发展引起了人们用局域搜索方法求解作业车间调度的兴趣，如模拟退火（SA），禁忌搜索（TS），遗传算法（GA）。

2.1 古典作业车间模型

古典作业车间调度问题可以表述为：有 m 台不同的机器和 n 个不同的工件，每个工件包含一个由多道工序组成的工序集合，工件的工序顺序是预先给定的。每道工序用它所要求的机器和固定的加工时间来表示。此外对工件和机器有一些约束，例如

- (1) 一个工件不能两次访问同一机器；
- (2) 不同工件的工序之间没有先后约束；
- (3) 工序一旦进行不能中断；
- (4) 每台机器一次只能加工一个工件；
- (5) 下达时间和交货期都不是给定的。

问题是确定机器上工序的顺序使最大流程时间，即完成所有工件的时间，达到最小。

2.1.1 整数规划模型

古典作业车间调度问题需要考虑两类约束：

- (1) 给定工件的工序前后约束；
- (2) 给定机器的工序非堵塞约束。

一般地，令 c_{jk} 表示工件 j 在机器 k 上的完工时间， t_{jk} 表示工件 j 在机器 k 上的加工时间。

首先考虑给定作业车间调度的工序前后约束。

对于工件 i ，如果在机器 h 上的加工先于机器 k ，有如下约束：

$$c_{ik} - t_{ik} \geq c_{ih} \quad (8)$$

另一方面，如果在机器 k 上的加工首先出现，有约束

$$c_{ih} - t_{ih} \geq c_{ik} \quad (9)$$

由于约束(8)和(9)中有且仅有一个成立，因此称它们为分离性约束(Disjunctive Constraints)。定义如下的指示系数：

$$a_{ihk} = \begin{cases} 1, & \text{对于给定的工件 } i, \text{ 如果在机器 } h \text{ 上的加工先于机器 } k \\ 0, & \text{其它} \end{cases}$$

以上约束也可以写成

$$c_{ik} - t_{ik} + M(1 - a_{ihk}) \geq c_{ih}; \quad i = 1, 2, \dots, n; \quad h, k = 1, 2, \dots, m$$

其中 M 是一个很大的数，注意上面的不等式很巧妙地吸收了相互排斥的约束。如果先在机器 h 上加工， $M(1 - a_{ihk})$ 为 0，给定的不等式正是所要的。否则， $M(1 - a_{ihk})$ 变得很大，不等式同样为真。

现在来考虑给定机器的工序非堵塞约束：

对于两个工件 i 和 j 都需要在机器 k 上加工，如果 i 先于工件 j 来到，有以下约束：

$$c_{jk} - c_{ik} \geq t_{jk}$$

另一方面, 如果工件 j 先来到, 有约束

$$c_{ik} - c_{jk} \geq t_{ik}$$

定义指示器变量 x_{ijk} 为

$$x_{ijk} = \begin{cases} 1, & \text{如果工件 } i \text{ 先于工件 } j \text{ 来到机器 } k \\ 0, & \text{其它} \end{cases}$$

以上约束可以写成

$$c_{jk} - c_{ik} + M(1 - x_{ijk}) \geq t_{jk}; \quad i, j = 1, 2, \dots, n; \quad k = 1, 2, \dots, m$$

以最大流程时间最小为目标的作业车间调度问题可以表述为

$$\begin{aligned} \min \quad & \max\{\max\{c_{ik}\}\} \\ \text{s. t.} \quad & c_{ik} - t_{ik} + M(1 - a_{ihk}) \geq c_{ih}; \quad i = 1, 2, \dots, n; \quad h, k = 1, 2, \dots, m \end{aligned} \quad (10)$$

$$c_{jk} - c_{ik} + M(1 - x_{ijk}) \geq t_{jk}; \quad i, j = 1, 2, \dots, n; \quad k = 1, 2, \dots, m \quad (11)$$

$$c_{ik} \geq 0; \quad i = 1, 2, \dots, n; \quad k = 1, 2, \dots, m \quad (12)$$

$$x_{ijk} = 0 \text{ 或 } 1; \quad i, j = 1, 2, \dots, n; \quad k = 1, 2, \dots, m \quad (13)$$

约束 (10) 保证每个工件的工序的加工顺序满足预先的要求; 约束 (11) 保证每台机器一次只能加工一个工件。

如果考虑平均流程时间问题, 目标可以写成:

$$\min \quad \sum_{i=1}^n \max\{c_{ik}\} \quad (14)$$

2.1.2 线性规划模型

令 $N = \{0, 1, 2, \dots, n\}$ 表示工序的集合, 其中 0 和 n 表示虚设的起始工序和完成工序; $M = \{1, 2, \dots, m\}$ 是机器的集合; A 是表示每个工件的工序前后关系约束的工序对集合; E_k 是机器 k 上加工的工序对集合。对每个工序 i , 加工时间 d_i 是一定的, 工序的起始时间 t_i 是优化工程中有待确定的变量。因此, 作业车间调度问题可以描述为

$$\min \quad t_n \quad (15)$$

$$\text{s. t.} \quad t_j - t_i \geq d_i, \quad (i, j) \in A \quad (16)$$

$$t_j - t_i \geq d_i \text{ 或 } t_i - t_j \geq t_j, \quad (i, j) \in E_k, \quad k \in M \quad (17)$$

$$t_i \geq 0 \quad (18)$$

约束 (16) 保证每个工件的工序顺序满足预先的要求, 约束 (17) 保证每台机器一次只能加工一个工件。问题的一个可行解称为一个调度。

2.2 传统启发式方法

作业车间调度问题是很重要的常见实际问题。由于作业车间调度问题是最困难的组合优化问题, 因此很自然地希望寻找一些近似方法能在有效时间内产生可接受的调度。作业车间调度问题的启发式方法大致分为两类:

(1) 一次通过启发式;

(2) 多次通过启发式。

一次通过启发式指通过基于优先分配规则一次固定一个工序, 简单地构造一个完全

解。有多种规则可用来从特定的子集中选择工序作为下一步的调度。这种启发式的特点是快，而且能找到不太坏的解。此外，可以重复应用一次通过启发式来建立更为复杂的多次通过启发式，以花费额外的计算费用来获得更好的调度。

常用的传统启发式方法有

- (1) 优先分配启发式方法；
- (2) 随机分配启发式方法；
- (3) 瓶颈移动启发式方法。

这些算法实现的详细过程可参见文献[52]。

习题三十二

1. (计划排序问题中的车间作业问题) 研究 n 个工件在 m 台机器上有序的加工问题，每个工件都有完工的日期 (DD, Due date)，加工的时间 (PT, Processing time) 和工件的价值 (VAL, Value if job is selected)。车间作业计划研究一个工厂生产工序的计划和安排，需要计划与合理安排各个工件在这些机器上加工的先后次序，即拟订加工工序，通过各个工件在各种机器上加工次序的合理安排，使得完成这批工件加工任务所需的总时间最省 (注：总时间即为各个零件的加工时间和加工其它零件时它们等待时间之和) 或要求整个选择加工的工件价值最大。

有一个工厂现在有 12 种工件 (编号为工件 1, 工件 2, ..., 工件 12) 需要在车床, 钻床, 铣床几种不同的设备上加工。考虑下面的工件加工的排序问题：

(1) 这 12 种工件都要求在车床上加工，车床一次只能加工一种工件，这 12 种工件加工所需时间，每个工件的完工时间和每个工件的价值如表 7 所示。

表 7 12 个加工工件的数据

工件	加工时间 (h)	完工时间(h)	工件价值
1	2.8	9	8
2	3.2	7.5	4
3	1.2	15	16
4	4	23	3
5	2.7	10	7
6	0.9	22	20
7	2.5	17	17
8	3.3	33	11
9	1.7	7	7
10	2.5	18	12
11	3.6	25	5
12	4.7	11	18

i) 不考虑工件的完工时间和工件的价值，为该工厂安排工件加工的次序，使得完成这批工件加工任务所需的总时间最省。建立数学模型并给出相应的算法。

ii) 由于工件必须在它们要求的时间内完工，按照表 7 的数据，为该工厂安排选择

加工工件的种类及加工的次序,使得整个选择加工的工件价值最大。建立数学模型并给出相应的算法。

(2) 如果这 12 种工件都要求先在车床上加工,然后再在钻床上加工(即工件在钻床加工之前必须先车床上加工过),每种机器一次只能加工一种工件,这 12 种工件加工所需时间如表 8 所示。

表 8 两台机器的加工时间

工件	车床加工时间 (h)	钻床加工时间(h)
1	2.8	4
2	3.2	1.3
3	1.2	1.8
4	4	2.2
5	2.7	3
6	0.9	4.5
7	2.5	1.7
8	3.3	2.5
9	1.7	4.5
10	2.5	2.5
11	3.6	3.8
12	4.7	1.9

为该工厂安排工件加工的次序,使得完成这批工件加工任务所需的总时间最省。建立数学模型并给出相应的算法。

(3) 如果这 12 种工件都要求先在车床上加工,然后再在钻床上加工,最后再在铣床上加工,每种机器一次只能加工一种工件,这 12 种工件加工所需时间如表 9 所示。

表 9 三台机器的加工时间

工件	车床加工时间 (h)	钻床加工时间 (h)	铣床加工时间 (h)
1	2.8	4	3
2	3.2	1.3	1
3	1.2	1.8	2.5
4	4	2.2	1.3
5	2.7	3	1.8
6	0.9	4.5	2
7	2.5	1.7	3.6
8	3.3	2.5	0.8
9	1.7	4.5	1
10	2.5	2.5	1.1
11	3.6	0.9	1.3

12	4.7	1.9	0.7
----	-----	-----	-----

为该工厂安排工件加工的次序,使得完成这批工件加工任务所需的总时间最省。建立数学模型并给出相应的算法。

(4)对于上述问题你做出的数学模型和相应的算法给出评价。并将模型推广到 n 个工件在 m 台机器上加工的一般的工件排序问题,给出你的想法和解决问题的思路。