组合优化理论

第8章 作业调度问题

主讲教师：陈安龙

第8章 作业调度问题

§1 单机调度问题

§2 平行机调度问题

§3 车间作业调度问题

作业调度问题

Example 1 机械加工

一个机械加工车间要加工一批机器零件，每一个零件都具有相同的工序，即按相同的顺序在几个不同的机床上加工，但每个零件在每个机床上的加工时间可能不同．如何按排加工顺序才能以最短的时间加工完所有的零件．

这是一个流水线调度问题．

Example 2 进程调度

在计算机多道程序操作系统中，并发执行多个进程，任何时刻CPU只能执行一个进程，进程的到达时间是不同的，怎样调度这些进程才能使CPU的利用率最高或进程的平均周转时间最短？

事先不知道每个进程的到达时间和执行时间——在线调度

事先知道随机到达时间和执行时间的分布、数学期望、方差，目标是极小化平均周转时间的数学期望——随机调度

Example 3 机场调度

在一个飞机场，有几十个登机口，每天有几百架飞机降落和起飞，登机口的种类和大小是不同的，而班机的机型和大小也是不同的.

飞机按时刻表降落和起飞，当飞机占有登机口时，旅客上下飞机，飞机要接受加油、维护和装卸行李等服务．由于天气和机场的原因，飞机不能起飞，登机时间推迟.

调度人员如何制订一个登机口的分配方案，使机场的利用率最高或晚点起飞的飞机最少.

登机口——机器， 飞机——零件， 机场的规定——约束条件

用一台或一台以上的机器加工两个或两个以上的零件（任务）时，确定加工顺序使效率最高。

—— 调度（*Scheduling*调度）问题

由于效率的度量方法的不同、引进不同的约束条件和机器的数量、类型等，使之得到不少的调度模型，也使调度问题有了更多的应用.

由于应用范围逐渐扩大，新的问题不断出现，因而从事这一领域研究的人与日俱增，其内容也越来越丰富，应用也越来越广泛.

确定性调度 (*Deterministic Scheduling*)

所有数据在进行决策前都是已知的

随机性调度 （*Stochastic Scheduling*）

有的数据在进行决策前是未知的，是随机变量，但它们的分布是已知的

在线调度 （*On-line Scheduling*）

半在线调度 (*Semi- On-line Scheduling*)

离线调度 （*Off-line Scheduling*）

用 $C=(C_1, C_2, \ldots, C_n)$ 表示任务的完工时间，极小化的目标函数总是完工时间 C_j 的函数.

常见的目标函数（效率的度量方法）

（1） 时间表长

时间表长（*schedule length，makespan*）定义为

$$C_{\max}=\max_j\{C_j\}$$

它等于最后一个被加工完任务的完工时间，小的时间表长意味着处理机的利用率高.

（2） 平均加权流时间和加权总完工时间

平均加权流时间（*mean weighted flow time*）是

$$F = \sum_{j=1}^{n} w_j F_j \Big/ \sum_{j=1}^{n} w_j .$$

任务的到达时间

其中 $F_j = C_j - r_j$ 是任务 T_j 的流（周转）时间，

它等于任务在系统中**等待时间和加工时间的和.**

对平均加权流时间进行变形，可得极小化 F 相当于极小化加权总完工时间（*total weighted completion time*）

$$C = \sum_{j=1}^{n} w_j C_j$$

（如果 $w_j = 1$ $j = 1,2,\ldots,n$ 即为总完工时间）

$$F = \sum_{j=1}^{n} w_j F_j \Big/ \sum_{j=1}^{n} w_j$$

$$= \sum_{j=1}^{n} w_j C_j \Big/ \sum_{j=1}^{n} w_j - \sum_{j=1}^{n} w_j r_j \Big/ \sum_{j=1}^{n} w_j$$

式中的第一项的分母和第二项都是常数

（3） 最大延误

最大延误（*maximum lateness*）定义为

$$L_{\max} = \max_j \{L_j\}$$

其中 $L_j = C_j - d_j$ 是任务 T_j 的延误时间 .

任务的截止期限

（4） 加权总误工

加权总误工（*total weighted tardiness*）是

$$D = \sum_{j=1}^{n} w_j D_j$$

其中 $D_j = max\{C_j - d_j, 0\}$ 是任务 T_j 的误工时间 .

（5）加权误工任务数

加权误工任务数（*weighted number of tardy tasks*）是

$$U=\sum_{j=1}^{n} w_j U_j$$

其中$U_j=\begin{cases}1 & C_j > d_j \\ 0 & C_j \le d_j\end{cases}$是对任务$T_j$误工的单位惩罚

调度问题的三要素：

机器（处理机）、作业（任务）、目标函数

用三元组 $\alpha|\beta|\gamma$ 描述一个调度模型

α：机器的数量和类型；

β：作业的约束条件；

γ：优化的目标函数.

基本假设：（1）任务或作业和处理机都是有限的；

（2）在任一时刻，任何处理机只能加工一个任务或工序；

（3）极小化单一目标函数.

Definition 1 对于一个可行调度，如果有准备好被加工的任务或工序，不准有空闲的处理机，称这种可行排序为无耽搁调度（*nondelay schedule*）；否则称为耽搁排序（*delay schedule*）。

无耽搁调度相当于**有工作可做就不能闲着**. 对于大多数调度问题，包括所有的可中断调度，最优调度是无耽搁调度，然而也有一些不可中断调度问题的最优调度是耽搁调度.

Example 4 调度问题

$1|r_j|\sum w_j C_j$

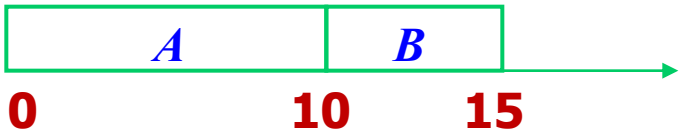
1：表示一台机器

r_j：表示任务有不同的到达时间

$n=2$，$t=(10，5)$，$r=(0，1)$，$w=(1，5)$

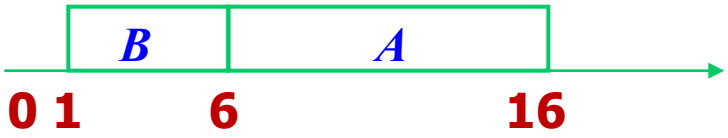
该问题有两个可行调度，用 *Gantt Charts* 表示：

nondelay schedule:



$Z_1 = 10*1+15*5 = 85$

delay schedule:



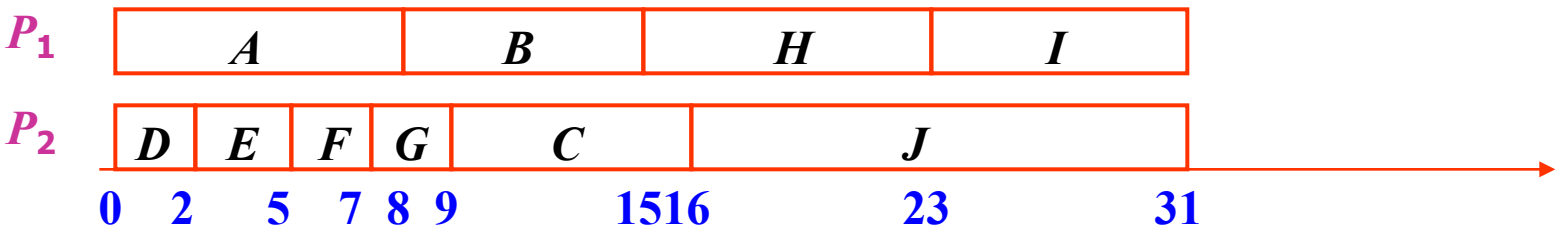
$Z_2 = 6*5 + 16*1 = 46$

Example 5 阿克米自行车的装配问题

工　序	紧前工序	加工时间	工　序	紧前工序	加工时间
A	——	8	*F*	*D*	2
B	*A*	7	*G*	*F*	2
C	*A*，*E*	7	*H*	*E*，*G*	8
D	——	2	*I*	*E*，*G*	8
E	*D*	3	*J*	*B*，*C*	15

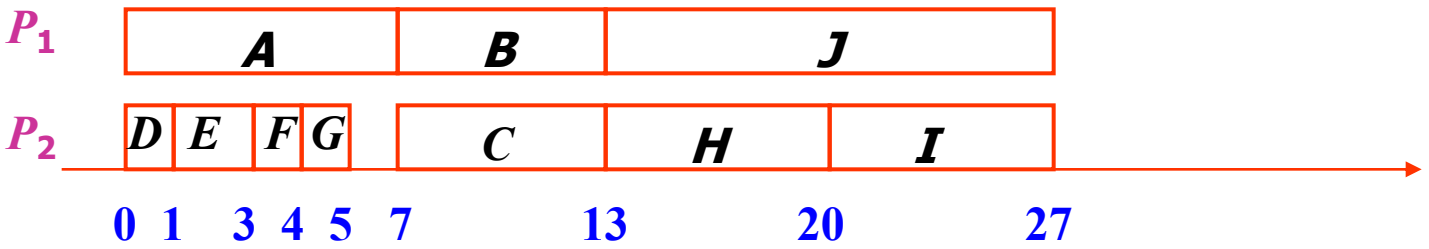
由两名熟练工人进行装配，要求装完时间最早.

这是一个 $P2|prec|C_{\max}$ 调度问题.



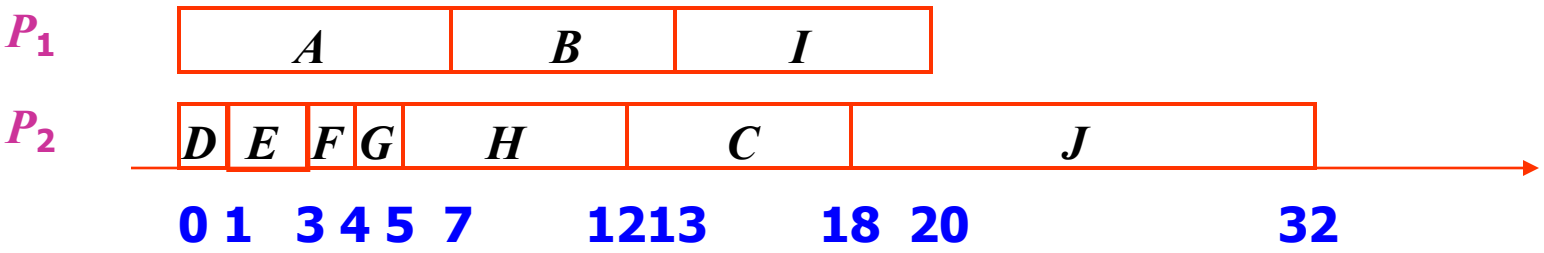
如果每道工序的加工时间减少1，最优时间表会小于 31 吗？是 26 吗？

工　序	紧前工序	加工时间	工　序	紧前工序	加工时间
A	——	7	*F*	*D*	1
B	*A*	6	*G*	*F*	1
C	*A*，*E*	6	*H*	*E*，*G*	7
D	——	1	*I*	*E*，*G*	7
E	*D*	2	*J*	*B*，*C*	14



最优耽搁调度

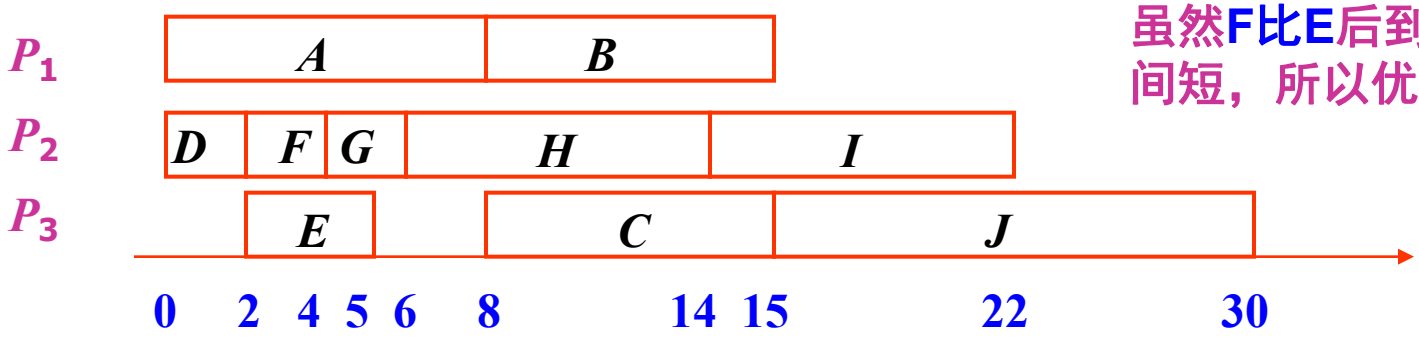
工　序	紧前工序	加工时间	工　序	紧前工序	加工时间
A	——	7	F	D	1
B	A	6	G	F	1
C	A，E	6	H	E，G	7
D	——	1	I	E，G	7
E	D	2	J	B，C	14



最优无耽搁调度

如果加工时间不变而增加一个装配工人，最优时间表会小于31 吗？

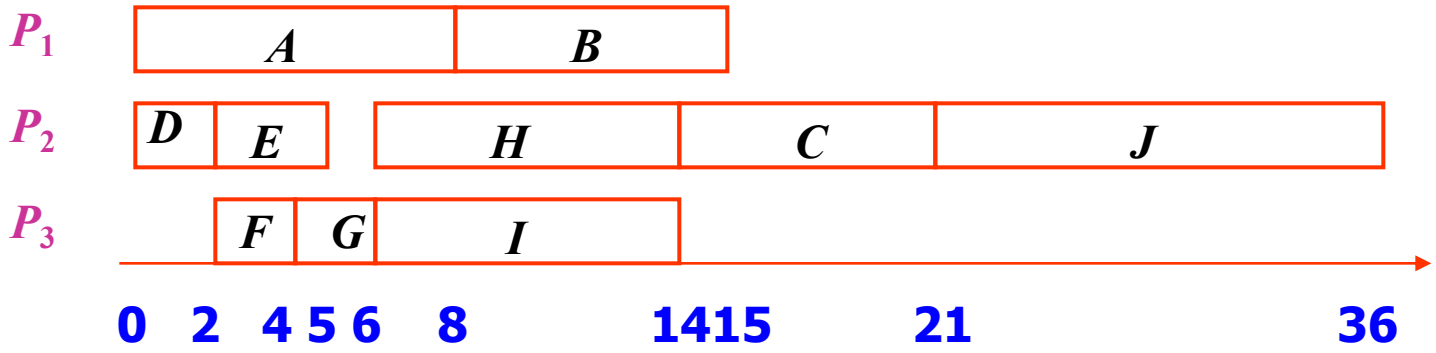
工　序	紧前工序	加工时间	工　序	紧前工序	加工时间
A	——	8	*F*	*D*	2
B	A	7	*G*	*F*	2
C	*A*，*E*	7	*H*	*E*，*G*	8
D	——	2	*I*	*E*，*G*	8
E	*D*	3	*J*	*B*，*C*	15



虽然F比E后到，但加工时间短，所以优先选择F加工

最优耽搁调度

工　序	紧前工序	加工时间	工　序	紧前工序	加工时间
A	——	8	F	D	2
B	A	7	G	F	2
C	A，E	7	H	E，G	8
D	——	2	I	E，G	8
E	D	3	J	B，C	15



最优无耽搁调度

§1 单机调度问题

单机调度问题是最简单的一类调度问题，同时也是最重要的调度问题之一．首先单机调度问题比较容易求出解决方法，这些方法对于研究较复杂的调度问题具有指导作用，可为处理复杂调度问题提供近似算法；其次，单机调度问题大量存在于现实生活中，具有广泛的实际背景，许多实际问题都可以归结为单机调度问题．

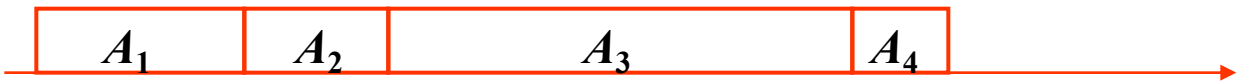
§1 单机调度问题

问题 $1||C_{max}$ 如何调度？

一、问题 $1||\sum w_j C_j$

Example 6 设一个机修车间有 n 台不同的机床要进行大修，它们的维修时间已知为 $t_1, t_2, \ldots, t_n$，而机床 A_i 在车间逗留的过程中每单位时间的损失费为 $w_i (i=1,\ldots,n)$ 试求一种调度，使得 n 台机床在修理完毕时，总的损失为最小.

如何调度？



***Solution*：**

令：$H=\{(k_1,k_2,\ldots,k_n)\,|\,(k_1,k_2,\ldots,k_n)$为1～n的一种排序$\}$

设 n台机床维修的调度为 $(k_1,\ k_2,\ \ldots,\ k_n)$ 则机床S的维修完毕的时间为 $C_{k_s}=\sum_{i=1}^{s} t_{k_i}$

n 台机床按此调度维修完时，总的损失费为

$$\sum_{i=1}^{n} w_{k_i} C_{k_i}$$

本题要寻找一种调度 $(r_1, r_2, \ldots, r_n)$ 满足

$$\sum_{i=1}^{n} w_{r_i} C_{r_i} = \min\left\{\sum_{i=1}^{n} w_{k_i} C_{k_i}\,|\,(k_1,k_2,\ldots,k_n)\in H\right\}$$

§1 单机调度问题

设有两调度 $(k_1,\cdots k_m,k_{m+1},\cdots k_n)$ （1）

$(k_1,\cdots k_{m+1},k_m,\cdots k_n)$ （2）

分析调度（1）与（2）的优劣

总损失费仅在 k_m,k_{m+1}处有区别

按（1）调度

A_{k_m} 和 $A_{k_{m+1}}$ 的损失费

$$w_{k_m}C_{k_{m-1}}+w_{k_m}t_{k_m}+w_{k_{m+1}}C_{k_{m-1}}+w_{k_{m+1}}t_{k_m}+w_{k_{m+1}}t_{k_{m+1}}$$

按（2）调度

A_{k_m} 和 $A_{k_{m+1}}$ 的损失费

$$w_{k_{m+1}}C_{k_{m-1}}+w_{k_{m+1}}t_{k_{m+1}}+w_{k_m}C_{k_{m-1}}+w_{k_m}t_{k_{m+1}}+w_{k_m}t_{k_m}$$

当$w_{k_{m+1}}t_{k_m} > w_{k_m}t_{k_{m+1}}$ 即$\frac{t_{k_m}}{w_{k_m}} > \frac{t_{k_{m+1}}}{w_{k_{m+1}}}$时，

调度（2）优于调度（1）.

Theorem 1 满足下列条件的调度 $(r_1, r_2, \ldots, r_n)$

$$\frac{t_{r_1}}{w_{r_1}} \le \frac{t_{r_2}}{w_{r_2}} \le \ldots \le \frac{t_{r_n}}{w_{r_n}}$$

为问题 $1\|\sum w_j C_j$ 的最优调度.

如：考虑调度问题 $1\|\sum w_jC_j$ 其中 $n=5,$

$t=(12,\ 4,\ 7,\ 11,\ 6),\quad w=(4,\ 2,\ 5,\ 5,\ 6)$

由 $\frac{t_1}{w_1}=3,\frac{t_2}{w_2}=2,\frac{t_3}{w_3}=1.4,\frac{t_4}{w_4}=2.2,\frac{t_5}{w_5}=1$

得最优调度为 $(A_5,\ A_3,\ A_2,\ A_4,\ A_1)$

此时 $\sum w_jC_j=6\times6+5\times13+2\times17$

$+5\times28+4\times40=435$

在 上例中，如果考虑各待维修的机床在机修车间**平均逗留时间（或总逗留时间）最短**，

如何调度？

或 $\frac{1}{n}\sum C_j$ （或 $\sum C_j$）

这只是 上例中 $w_j = 1$ 的特例

所以，满足下列条件的调度（$r_1, r_2, \ldots, r_n$）

$$t_{r_1} \le t_{r_2} \le \ldots \le t_{r_n}$$

为最优调度 .

§1 单机调度问题

以下讨论的调度问题都与工期有关，即每个任务均有一个工期。工期 d_j 表示对任务 T_j 限定的完工时间. 如果不按期完工，应受到一定的惩罚.

二、问题 $1\|L_{max}$

$$L_{max} = \max\{C_j - d_j\}$$

任务没有准备时间的**最大延误的调度问题**比较简单，只需将任务按最早工期优先（*Earliest Due Date first*，简记 *EDD*）规则，就可以得到最优调度. 按照这一规则，任务按 d_j 不减的顺序进行调度.

Theorem 2：对于问题$1\|L_{\max}$，*EDD* 规则可以得到最优调度 .

Example 7 考虑调度问题 $1\|L_{max}$，其中

$n=6$ ，$t=(3,1,4,1,3,2)$，$d=(2,10,6,4,11,12)$

由*EDD*规则可以求得最优调度

$$(T_1,T_4,T_3,T_2,T_5,T_6)$$

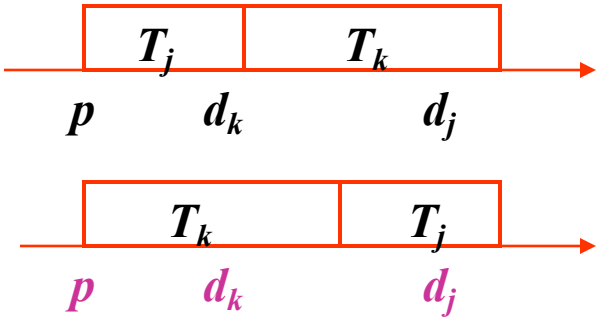
最大延误为 $L_{max}=2$

Theorem 2 的证明

只需证明任何不满足EDD规则的调度，均可转化为满足EDD规则而目标函数不增。

设某一调度 s 违反了 EDD 规则，则在此调度中，至少有两个相邻任务

T_j、T_k，T_j排在T_k之前，而$d_j > d_k$



设 T_j 在时间 p 时开始加工，则

$$L_j = p + t_j - d_j,\quad L_k = p + t_j + t_k - d_k$$

对调 T_j T_k的位置，其余任务位置不变，得一排序S'.

在这排序中 $$L'_j = p + t_j + t_k - d_j,\quad L'_k = p + t_k - d_k$$

因为 $d_j > d_k$，所以 $L_k > L'_j, L_k > L'_k$ 从而 $L_{\max} \geq L'_{\max}$

三、延误问题 $1\|\sum U_j$

误工任务数问题

在许多情况下，延误时间的长短不重要。只要延误发生，造成同样影响。更多关注**延误的最少任务数**。

Example 8 设有 n 个工件 $T_1, T_2, \ldots, T_n$ 要在一台机器上加工，加工时间分别为 $t_1, t_2, \ldots, t_n$，要求的交货日期分别为 $d_1, d_2, \ldots, d_n$. 试求一种加工调度，使得误期交货的工件最少 .

算法:

（1）将任务按**最早工期优先升序**排序建立调度任务队列，同时建立一个延期任务队列；

（2）计算各任务的完工时间，如果当前调度已无延误任务，则转（5），否则转（3）；

（3）从开始查找到第1个延误任务，设该延误是**第 k 个任务**；

（4）从前 k 个任务中，**选取加工时间最长的任务，移入延期任务队列**，剩下的形成部分调度序列，转（2）；

（5）将**延期任务队列的任务**放在所得的**部分调度之后**，得到最优调度.

Theorem 3：对于问题 $1\|\sum U_j$，上述算法给出最优调度

证明：假定 $d_1 \le d_2 \le \ldots \le d_n$，令 F_k 表示前 k 个任务构成的集合是 $\{T_1, T_2, \ldots, T_n\}$ 的子集，满足下述两个条件：

1、在任务集 $\{T_1, T_2, \ldots, T_k\}$ 的所有子集中，F_k 具有最多按期完工的任务，按期完工的任务数记为 N_k；

2、在 $\{T_1, T_2, \ldots, T_k\}$ 的所有含有 N_k 个按期完工任务的子集中，F_k 中的任务所用的总加工时间最少．

集合 F_n 与最优调度相对应．下面用数学归纳法证明算法产生的调度就是 F_n．

当 $k=1$ 时，显然满足；

假设对前 k 个任务算法产生的调度是 F_k，F_k 满足上述两个条件；

对前 $k+1$ 个任务，由 F_k 出发，按算法要求可产生满足上述两个条件的 F_{k+1}，

分两种情况讨论：

Case 1 将任务 T_{k+1} 加入 F_k 后，T_{k+1} 按期完工．

此时，$N_{k+1}=N_k+1$，$F_{k+1}=F_k \cup \{T_{k+1}\}$，显然上述两个条件满足；

Case 2 将T_{k+1}加入F_k后，任务T_{k+1}没有按期完工.

由N_k是任务集$\{T_1,T_2,\ldots T_k\}$的子集中按期完工任务数最大的一个，以及F_k是含有N_k个任务的子集中加工总时间最少的一个，可知$N_{k+1}=N_k$. 将T_{k+1}加入F_k中没有增加按期完工的任务数，但应从任务集$F_k \cup \{T_{k+1}\}$中删除加工时间最大的一个任务，因此F_{k+1}满足上述两个条件.□

Example 9 考虑调度问题 $1\|\sum U_j$，其中 $n=8$

$t=(10,6,3,1,4,8,7,6)$，$d=(35,20,11,8,6,25,28,9)$

Solution： 按***EDD***规则，重新调度得右表.

此时，任务T_8延误，而在前三项任务中，T_8的加工时间最长，所以将T_8放至最后，得一新表.

i	1	2	3	4	5	6	7	8
T_{ri}	T_5	T_4	T_8	T_3	T_2	T_6	T_7	T_1
t_{ri}	4	1	6	3	6	8	7	10
C_{ri}	4	5	11	14	20	28	35	45
d_{ri}	6	8	9	11	20	25	28	35

此时，任务 T_7 延误，而在前六项任务中，T_6 的加工时间最长，所以将 T_6 放至最后，得一新表.

i	1	2	3	4	5	6	7	8
T_{ri}	T_5	T_4	T_3	T_2	T_6	T_7	T_1	T_8
t_{ri}	4	1	3	6	8	7	10	6
C_{ri}	4	5	8	14	22	29	39	45
d_{ri}	6	8	11	20	25	28	35	9

目前，前六项任务中已没有延误任务，所以此时为最优调度.

有两个任务T_8、T_6延误.

i	1	2	3	4	5	6	7	8
T_{ri}	T_5	T_4	T_3	T_2	T_7	T_1	T_8	T_6
t_{ri}	4	1	3	6	7	10	6	8
C_{ri}	4	5	8	14	21	31	37	45
d_{ri}	6	8	11	20	28	35	9	25

设 D_j 表示任务 T_j 的误工时间，使整个误工 $\sum D_j$ 最小的调度是十分重要的．因为单纯讨论使**误工任务数最少**可能会使有些任务的**等待时间变得很长**．如果将目标函数换成 $\sum D_j$，研究它的极小化，则不会产生上述现象，这也很有应用背景．

自然会想到能否按 ***EDD*** 规则调度，即按 d_j 不减的顺序进行调度．能得到最优调度吗？

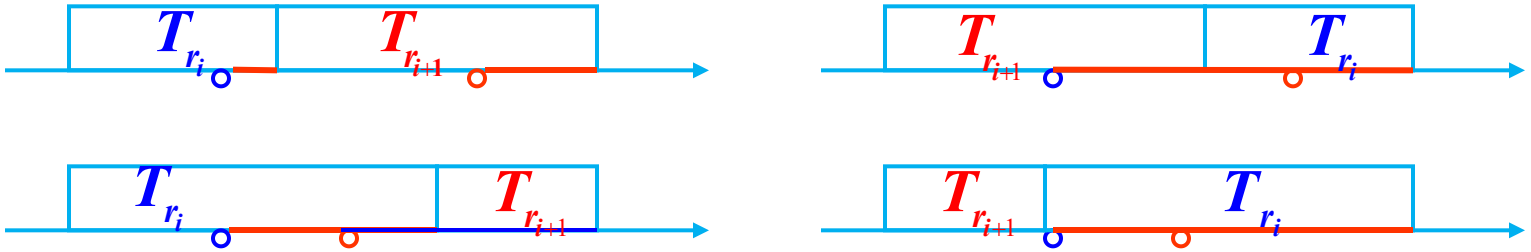
设调度 $(r_1, r_2, \ldots, r_n)$ (1) 满足 $d_{r_1} \le d_{r_2} \le \ldots \le d_{r_n}$ 与调度 $(r_1, \ldots, r_{i+1}, r_i, \ldots, r_n)$ (2)，进行比较：

若 T_{r_i}，$T_{r_{i+1}}$ 在（1）中不误期，则在（2）中 $T_{r_{i+1}}$ 不误期,而在 T_{r_i} 前插入 $t_{r_{i+1}}$ 单位时间,就有误期的可能；

若 T_{r_i} 在（1）中不误期，而 $T_{r_{i+1}}$ 在（1）中误期 l 单位时间，则由于 $d_{r_i} \le d_{r_{i+1}}$，任务 T_{r_i} 在（2）的误期 $\ge l$;

若 T_{r_i} 在(1)中有误期 l 单位时间，而 $T_{r_{i+1}}$ 在(1)中没有误期，则在(2)中 $T_{r_{i+1}}$ 仍没有误期，而在 T_{r_i} 前插入 $t_{r_{i+1}}$ 单位时间，任务 T_{r_i} 在(2)中的误期 $l + t_{r_{i+1}} \geq l$;

若 T_{r_i}、$T_{r_{i+1}}$ 在(1)中都有误期 l、s 单位时间，则



§2 平行机调度问题

平行机调度问题（*Parallel Machine Scheduling*）是多处理机调度问题的一种情况．所谓平行机是指参与完成任务的的处理机具有完全相同的作用，即任务在任一处理机上处理都可以． ***PMS*** 是调度中研究较早，很有代表性的一个问题，在理论上它是单机调度问题的推广，在应用上则具有更广泛的实际背景.

任务集
- 无关（任务间是独立的）
- 相关（任务有紧前要求）

处理机
- 同速机
- 恒速机
- 变速机

一、 问题 $Pm \parallel C_{\max}$

可中断如何？

同速机不可中断地处理无关任务集的时间表长问题.

设有 m 台完全相同的处理机 $P_j(j=1\sim m)$，n个相互独立的任务 $J_i(i=1\sim n)$，J_i 的加工时间为 $t_i(i=1\sim n)$

$$\text{设}\quad x_{ij}=\begin{cases}1 & \text{任务 } J_i \text{ 在处理机 } P_j \text{ 上加工}\\ 0 & \text{否则}\end{cases}$$

则问题 $Pm \parallel C_{\max}$ 可用 *IP* 描述如下：

$$\min \quad f = C_{\max}$$

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

$$C_{\max} \geq \sum_{i=1}^{n} t_i x_{ij} \qquad j = 1\sim m$$

§2 平行机调度问题

该问题与装箱问题是密切相关的，有相同的判定问题，常互称为对偶问题. 把箱子与处理机对应，物品与任务对应， 装箱问题是箱长给定，目标是箱子数最少. 平行机问题是箱子数给定，而使箱子长度最短.

Theorem 4 问题 $P2 \| C_{\max} \in NP-hard.$

(1)考察它的连续松弛问题 $0 \le x_{ij} \le 1 \ (i=1\sim n, \ j=1\sim m)$

则松弛问题的最优值 $\frac{1}{m}\sum_{i=1}^{n} t_i$

(2) 对原问题的任一实例 I，一定有 $f_{opt}(I) \ge \max_{1\le i\le n}\{t_i\}$

Theorem 5 问题 $Pm \| C_{\max}$ 最优值的一个下界为

$$L = \max\{\frac{1}{m}\sum_{i=1}^{n} t_i \ , \max_{1\le i\le n}\{t_i\}\}.$$

二、近似算法

1、*LS* 算法 （*List Scheduling*）

LS算法是由***Graham***于**1966**年首先提出，他在研究***LS***算法的近似程度时，第一次提出了近似算法的最坏情况进行分析的办法. 从此讨论近似算法的**绝对性能比**，就广泛地应用于组合优化的研究中.

绝对性能比 $R = \max\left\{\frac{z(I)}{z_{opt}(I)} \mid \forall 实例 I\right\}$

§2 平行机调度问题

*LS*算法的思想是按任务给定的顺序，将每一个工件分给**最早空闲**的机器（也即使该工件最早完工的机器）加工，在安排当前任务的加工时，**不要求知道下一个工件的信息**，所以特别适用于在线调度问题.

LS 算法

step 1 设 $L_j = 0 \quad j = 1 \sim m \quad k = 1$

step 2 若 $L_{j_0} = \min_{1 \le j \le m} \{ L_j \}$ 令 $x_{kj_0} = 1,$

$x_{kj} = 0 \ j \neq j_0, j \in \{1, 2, \ldots, m\} \quad L_{j_0} = L_{j_0} + t_k \quad k = k+1$

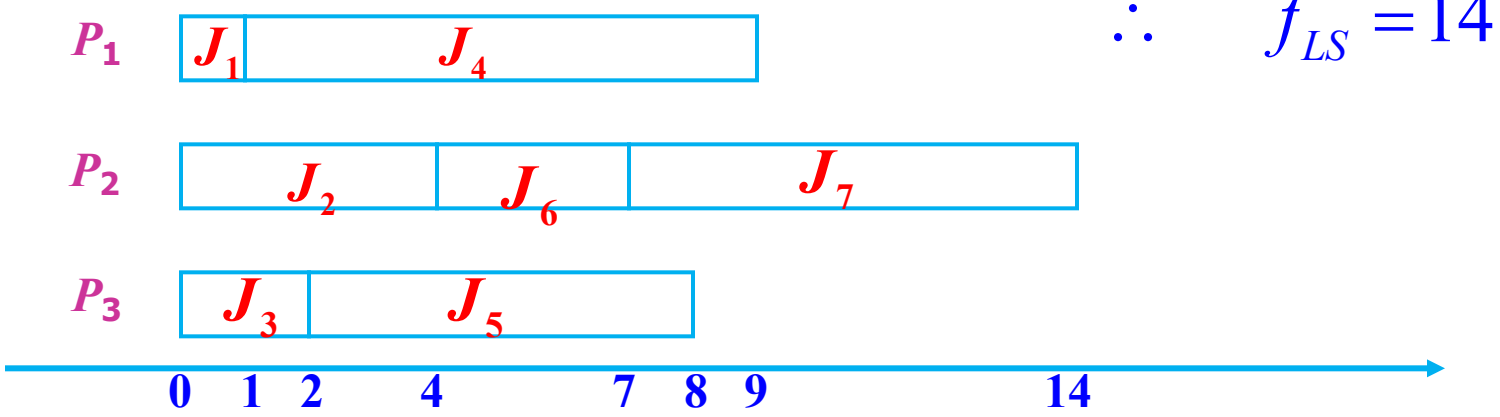
若 $k = n+1$ 时，停止；否则 重复 *step* 2 .

Theorem 6 $R_{LS} = 2 - \dfrac{1}{m}$

Example 10 考虑调度问题 $Pm \parallel C_{\max}$，其中

$m = 3,\ n = 7,\ t = (1, 4, 2, 8, 6, 3, 7)$.

Solution :



$\therefore \quad f_{LS} = 14$

Theorem 6 的证明

***Proof*：** 分两步

(1) 证明对任意的实例 I, $\dfrac{f_{LS}(I)}{f_{opt}(I)} \le 2-\dfrac{1}{m}$

(2) 说明该界不可改进 .

（1）用反证法证明 $\dfrac{f_{LS}(I)}{f_{opt}(I)} \le 2-\dfrac{1}{m}$

假设该结论不成立，则存在反例 I 使 $\dfrac{f_{LS}(I)}{f_{opt}(I)} > 2-\dfrac{1}{m}$,

考虑反例中任务数最少的一个（称为最小反例）.

由于 I 为最小反例，具有性质 $f_{LS}(I)$ 等于最后一个任务 J_n 的完工时间 .

下面证明最小反例的该性质成立。

因为若不然，设 J_k 的完工时间等于 $f_{LS}(I)$, $k < n$.

考虑新的任务集 $J_1, J_2, \ldots, J_k$, 则对由此任务得到的新实例 I^* 有 $f_{LS}(I^*) = f_{LS}(I)$, 而且 $f_{opt}(I^*) \le f_{opt}(I)$,

因此 有 $$\frac{f_{LS}(I^*)}{f_{opt}(I^*)} \ge \frac{f_{LS}(I)}{f_{opt}(I)} > 2 - \frac{1}{m}$$

说明 I^* 是一个更小的反例 .

由 s 为开始加工 J_n 时刻，则 $f_{LS}(I) = s + t_n$.

由 LS 规则，J_n 是分给最早空闲的机器加工，$s \le \frac{1}{m}\sum_{i=1}^{n-1} t_i$

由 $Th\ 5$ 知 $f_{opt}(I) \ge t_n$ 及 $f_{opt}(I) \ge \frac{1}{m}\sum_{i=1}^{n} t_i$

因此

$$\frac{f_{LS}(I)}{f_{opt}(I)} = \frac{s + t_n}{f_{opt}(I)} \le \frac{\frac{1}{m}\sum_{i=1}^{n-1} t_i + t_n}{f_{opt}(I)} = \frac{\frac{1}{m}\sum_{i=1}^{n} t_i + (1 - \frac{1}{m})t_n}{f_{opt}(I)}$$

$$\le 1 + (1 - \frac{1}{m}) = 2 - \frac{1}{m}$$

这与 I 是反例矛盾，此矛盾说明 $R_{LS} \le 2 - \frac{1}{m}$.

（2）考虑任务集 $\{J_1, J_2, \ldots, J_{m^2-m+1}\}$，其加工时间分别为：$t_1 = t_2 = \ldots = t_{m^2-m} = 1$，$t_{m^2-m+1} = m$，

需分给 m 台机器加工，易证 $f_{LS}(I) = 2m - 1, f_{opt}(I) = m$.

故 $$\frac{f_{LS}(I)}{f_{opt}(I)} = 2 - \frac{1}{m}$$

因此 有 $$R_{LS} = 2 - \frac{1}{m}$$ □

2、*LPT* 算法 （*Largest Processing Time*）

LPT 算法思想是**先将任务按其加工时间从大到小的顺序排列**，然后用*LS*算法调度。这也是*Graham* 给出的，它要求任务的信息全部已知后才开始加工。

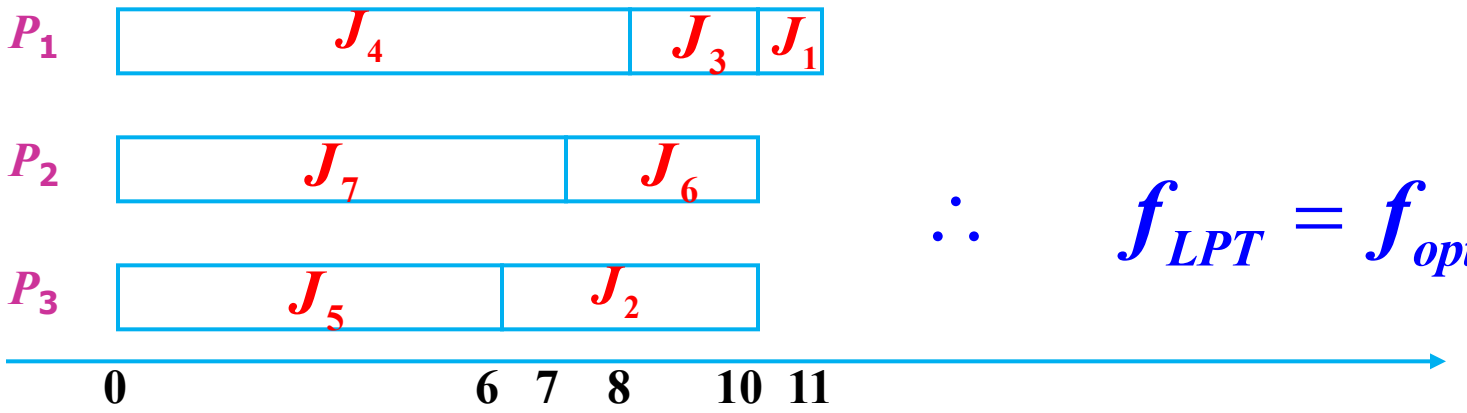
Theorem 7 $R_{LPT} = \frac{4}{3} - \frac{1}{3m}$

见前例 $t = (1, 4, 2, 8, 6, 3, 7)$

按加工时间重新排列

$J = (J_4, J_7, J_5, J_2, J_6, J_3, J_1)$ $t = (8, 7, 6, 4, 3, 2, 1)$

$$J = (J_4, J_7, J_5, J_2, J_6, J_3, J_1) \qquad t = (8, 7, 6, 4, 3, 2, 1)$$



$$\therefore \qquad f_{LPT} = f_{opt} = 11$$

§3 车间作业调度问题

车间作业调度问题是多处理机中多类型机调度问题

设有作业集 $J=\{J_1,J_2,\ldots,J_n\}$

处理机集 $P=\{P_1,P_2,\ldots,P_m\}$

m个处理机具有不同的功能

每个作业 J_j 有 m 道工序：T_{1j}，T_{2j}，…，T_{mj}，

工序 T_{ij} 的加工时间为 t_{ij} $(t_{ij}\geq 0)$.

各作业分别在处理机 P_1，$P_2,\ldots,P_m$ 上完成各道工序.

车间作业调度问题： 1、同顺序(流水)作业调度问题

2、异顺序作业调度问题

3、自由(开放)作业调度问题

Note: 在流水作业调度问题中，各作业均依次在处理机 $P_1, P_2, \ldots, P_m$ 上完成各道工序. 但对于同一台处理机, 各作业在其上的加工顺序可能不同.

排列调度（*permutation schedule*）

各作业在全部处理机上的加工顺序相同的调度

所有调度共有调度数 $(n!)^m$ 其中排列调度共有 $n!$

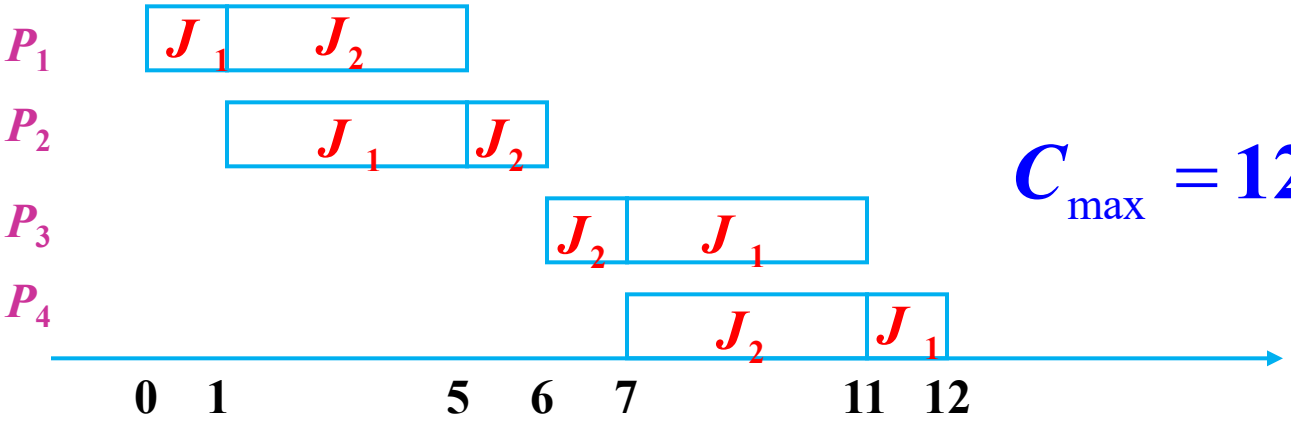
对于 $m \geq 4$ 的情况，同顺序作业调度问题的最优调度未必是排列调度. 即排列调度中可能不含有最优调度 .

Example 11 考虑调度问题 $Fm\|C_{\max}$ 其中 $m=4, n=2.$

$$t=\begin{pmatrix}1 & 4\\ 4 & 1\\ 4 & 1\\ 1 & 4\end{pmatrix}$$

m 个处理机，流水作业

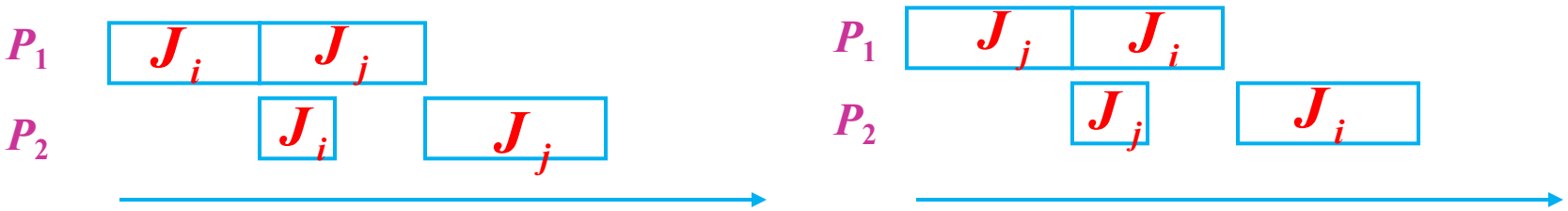
排列调度共有两个，调度时间表长均为



$C_{\max}=12\ (opt)$

最优调度不是排列调度，也不是无耽搁调度

Theorem 8 对于流水作业调度问题，至少存在一个最优调度，在此最优调度中，其最前面两台处理P_1，P_2上各作业的加工顺序相同.



Theorem 9 对于流水作业调度问题，至少存在一个最优调度，在此最优调度中，其最后两台处理P_{m-1}，P_m上各作业的加工顺序相同.

一定有在第一台处理机上无耽搁的最优调度

§3 车间作业调度问题

问题 $F2\|C_{\max}(\in P)$ ①

Johnson 算法 (*SPT-LPT*)

Shortest Processing Time first
Longest Processing Time first

(1) 把作业按工序加工时间分成两个子集：

$$J1=\{J_j \mid t_{1j}<t_{2j}\},\quad J2=\{J_j \mid t_{1j}>t_{2j}\}$$

对于满足 $t_{1j}=t_{2j}$ 的作业可分在任一集中；

(2) 先将集 J1 中的作业按 t_{1j} 不减排列 (*SPT* 规则)，

再将集 J2 中的作业按 t_{2j} 不增排列 (*LPT* 规则).

***Theorem* 10** 对于调度问题 ①，*Johnson* 算法产生最优调度.

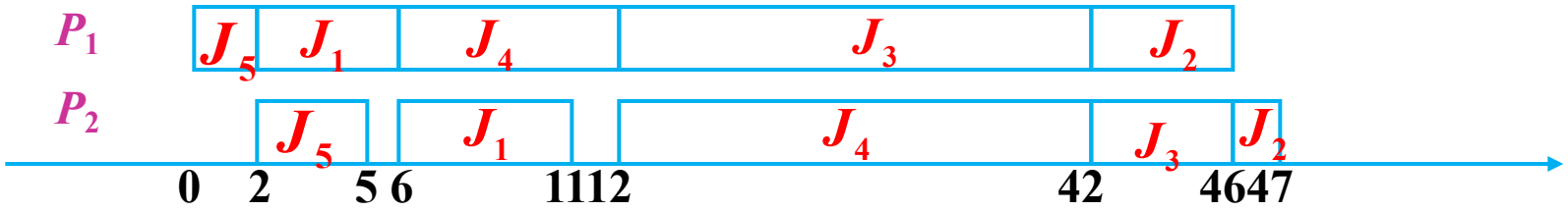
Example 12 考虑调度 $F2 \parallel C_{\max}$ 其中 $n=5$

$$t=\begin{pmatrix} 4 & 4 & 30 & 6 & 2 \\ 5 & 1 & 4 & 30 & 3 \end{pmatrix}$$

Solution： 由 *Johnson* 算法可得：

$$J1=\{J_1,J_4,J_5\},\quad J2=\{J_2,J_3\}.$$

J1 中的作业按 t_{1j} 不减排列：J_5,J_1,J_4; J2 中的作业按 t_{2j} 不增排列：J_3, J_2, 所以最优调度为 $[J_5, J_1, J_4, J_3, J_2]$, 时间表长为 $C_{\max}=47$.



***Example* 13** 考虑调度 $F4 \parallel C_{\max}$ 其中 $n=3$

$$t=\begin{pmatrix} 8 & 5 & 0 \\ 0 & 9 & 2 \\ 1 & 0 & 10 \\ 2 & 6 & 0 \end{pmatrix}$$

建立整数规划模型.

***Solution*:** 设 x_{ij} 为作业 J_i 在处理机 P_j 上开始加工的时间，第一组约束为每一作业在处理机上加工顺序：

J_1: $x_{11}+8\le x_{13}$ $x_{13}+1\le x_{14}$

J_2: $x_{21}+5\le x_{22}$ $x_{22}+9\le x_{24}$ J_3: $x_{32}+2\le x_{33}$

第二组约束为一族选择性的约束条件，以保证每一处理机同一时间只能处理一个作业：如对 P_1

有： $x_{11}+8\le x_{21}$ 或 $x_{21}+5\le x_{11}$

引进 0-1 变量 y_1，上述选择性约束条件为：$t=\begin{pmatrix}8&5&0\\0&9&2\\1&0&10\\2&6&0\end{pmatrix}$

$$x_{11}+8\le x_{21}+My_1 \qquad x_{21}+5\le x_{11}+M(1-y_1)$$

类似 y_2, y_3, y_4 为 0-1 变量，对处理机 $P_2, P_3. P_4$ 有

$$x_{22}+9\le x_{32}+My_2 \qquad x_{32}+2\le x_{22}+M(1-y_2)$$
$$x_{13}+1\le x_{33}+My_3 \qquad x_{33}+10\le x_{13}+M(1-y_3)$$
$$x_{14}+2\le x_{24}+My_4 \qquad x_{24}+6\le x_{14}+M(1-y_4)$$

第三组约束条件为三个作业的完工时间

$$C_{\max} = \max \{ x_{14} + 2, x_{24} + 6, x_{33} + 10 \}$$

$$t = \begin{pmatrix} 8 & 5 & 0 \\ 0 & 9 & 2 \\ 1 & 0 & 10 \\ 2 & 6 & 0 \end{pmatrix}$$

将它化为线性约束

$$C \geq x_{14} + 2, \quad C \geq x_{24} + 6, \quad C \geq x_{33} + 10$$

目标函数为

$$\min f = C$$

若在原问题上再要求作业 J_2 在各处理机上的加工和等待时间总和不超过 21 .

则 $x_{24} + 6 - x_{21} \leq 21$

本章结束