《计算机算法设计与分析》

第五章 动态规划

马丙鹏

2019年10月27日

中国科学院大学

University of Chinese Academy of Sciences

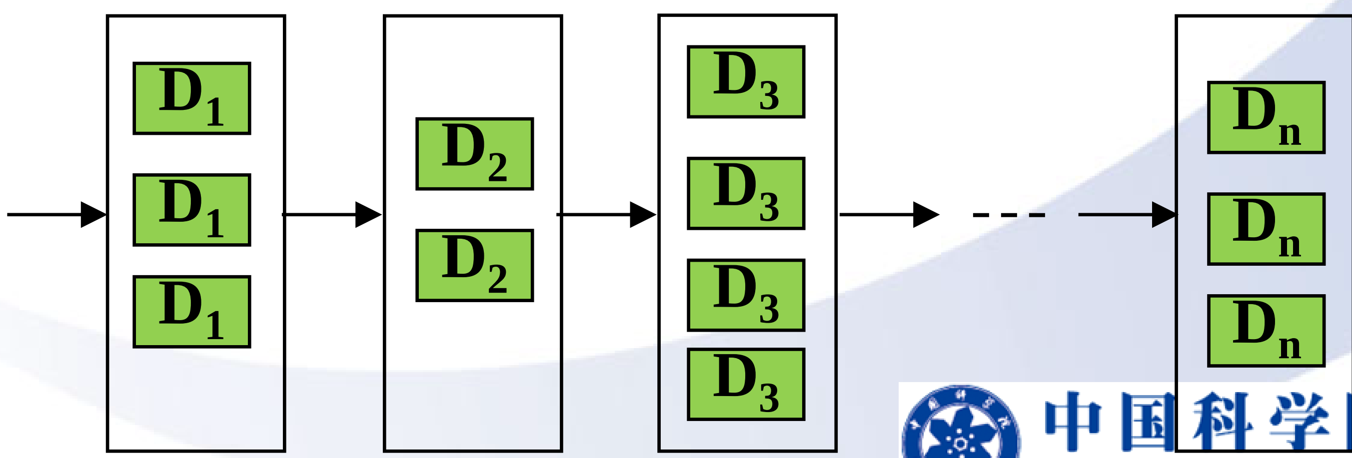
第五章 动态规划

- 5.1 一般方法
- 5.2 多段图问题
- 5.3 每对结点之间的最短路径
- 5.4 最优二分检索树
- 5.5 0/1背包问题
- 5.6 可靠性设计
- 5.7 货郎担问题
- 5.8 流水线调度问题

5.6 可靠性设计

- ■ 问题描述
  - □乘数最优化问题

  - □设r_i是设备D_i正常运转的概率，则整个系统的可靠性就是$\prod r_i$
  - □若$n=10$, $r_i=0.99$, $1 \le i \le 10$，则：$\prod r_i = 0.904$



5.6 可靠性设计

■ 问题描述

- 第i级每个设备出故障的概率为$1-r_i$，第i级有m_i个设备并联，则同时出故障的概率为$(1-r_i)^{m_i}$，可靠性为 $1-(1-r_i)^{m_i}$
- 第 i 级设备的可靠性由函数$\phi_i(m_i)$给定，$1\le i\le n$，那么整个系统的可靠性是 $\prod_{1\le i\le n}\phi_i(m_i)$
- 设计系统时需考虑成本，c_i表示第i级一台设备的成本，C表示要设计的系统允许的最大成本。
- 系统中每种设备至少有一台，设备i允许配置的台数至多为:

$$u_i = \left\lfloor \left(C - \sum_{k=1,k\ne i}^{n} c_k \right) \Big/ c_i \right\rfloor = \left\lfloor \left(C + c_i - \sum_{k=1}^{n} c_k \right) \Big/ c_i \right\rfloor$$

5.6 可靠性设计

- 问题描述
  - 可靠性设计最优化问题是在可容许的最大成本C的约束下，如何使系统的可靠性达到最优的问题。
- 数学模型
  - RELI(1, i, X)：表示在可容许成本X约束下，对第1种到第i种设备的可靠性设计问题。

    目标函数： $\max \prod_{1\le j\le i} \varphi_j(m_j)$

    约束条件： $\sum_{1\le j\le i} c_j m_j \le X$

    其中 $c_j > 0, 1 \le m_j \le u_j = \left\lfloor (X + c_j - \sum_{k=1}^{i} c_k) / c_j \right\rfloor$
  - RELI(1, n, c)：表示整个系统的可靠性设计问题。

5.6 可靠性设计

- 最优性原理对可靠性设计问题成立
  - 假设$m_1, m_2, \ldots, m_n$为RELI(1, n, c)的最优解，
  - 假设m_1为第1级的最优选择，
  - 则$m_2, \ldots, m_n$为RELI(2, n, c-m_1c_1)的最优解，
  - 否则，设$m'_2, \ldots, m'_n$为RELI(2, n, c-m_1c_1)的最优解，
  - 则$m_1, m'_2, \ldots, m'_n$为RELI(1, n, c)的最优解。与假设矛盾。
  - 系统可靠性设计问题RELI(1, n, c)的最优解是对$m_1, m_2, \ldots, m_n$的一系列决策的结果。
  - 每次决策可以确定一个m_i。

5.6 可靠性设计

- 可靠性设计的向后递推
  - 设$f_i(X)$是在成本不超过X的约束下，前i种设备组成的子系统RELI(1, i, X)的可靠性的最优值，即

$$f_i(X) = \max \prod_{1 \le j \le i} \phi_j(m_j)$$

$$f_i(X) = \max_{1 \le m_i \le u_i} \{\phi_i(m_i) f_{i-1}(X - c_i m_i)\}$$

  - 则RELI(1, n, c)可靠性设计的最优值为：

$$f_n(c) = \max \prod_{1 \le j \le n} \phi_j(m_j)$$

  - 则 $f_n(c) = \max_{1 \le m_n \le u_n} \{\phi_n(m_n) f_{n-1}(c - c_n m_n)\}$

5.6 可靠性设计

- **可靠性设计的求解**
  - 初始条件：$f_0(X)=1$，$0 \le X \le c$
  - S^i由(f, X)形式的序偶所组成，其中$f= f_i(X)$。
  - $S^i=\{ (f, X) \mid f = f_i(X)\}$为可靠性设计问题RELI(1, i, X)的最优解
  - 用类似于解0/1背包问题的方法可以求解递归关系

$$f_i(X) = \max_{1 \le m_i \le u_i} \left\{ \phi_i(m_i) f_{i-1}(X - c_i m_i) \right\}$$

  - 支配规则对这个问题也适用，即当且仅当$f_1 \ge f_2$而$X_1 \le X_2$时，(f_1, X_1)支配(f_2, X_2)
  - S^i_j: 第i级配备了j个设备时前i级子系统的可靠性。

5.6 可靠性设计

■ 可靠性设计的求解

S^i的生成：

$S^0=\{(1, 0)\}$

由S^{i-1}求S^i

① 分别求S^i_j，$1\le j\le u_i$。对于m_i的所有可能值，依次求出$m_i=j$，$1\le j\le u_i$时，有可能得到的所有序偶的集合

$S^i_j=\{(f*\phi_i(j), x+j*c_i)|(f, x)\in S^{i-1}\}$

② 合并所有的S^i_j为S^i，使用支配规则。

5.6 可靠性设计

- 例
  - 设计一个由设备D_1，D_2，D_3组成的三级系统。每台设备的成本分别为30元，15元和20元，可靠性分别是0.9，0.8和0.5，计划建立该系统的投资不得超过105元。假定，若i级有m_i台设备D_i并联，则该级的可靠性 $\varphi_i(m_i)=1-(1-r_i)^{m_i}$ 。
  - 应如何设计使可靠性达到最高。
  - 解：上述条件可以表示为：$c=105$；$c_1=30$，$c_2=15$，$c_3=20$；$r_1=0.9$，$r_2=0.8$，$r_3=0.5$。

设备	D_1	D_2	D_3
单价	30	15	20
可靠性	0.9	0.8	0.5

5.6 可靠性设计

设备	D_l	D_2	D_3
单价	30	15	20
可靠性	0.9	0.8	0.5

$u_1 = \lfloor (105-15-20)/30 \rfloor = 2$;

$u_2 = 3$, $u_3 = 3$,

$S^0 = \{(1, 0)\}$

$\phi_1(1) = 0.9$, $\phi_1(2) = 1-(1-0.9)^2 = 0.99$

$$\left.\begin{array}{l} S^1_1 = \{(0.9, 30)\} \\ S^1_2 = \{(0.99, 60)\} \end{array}\right\} S^1 = \{(0.9, 30), (0.99, 60)\}$$

5.6 可靠性设计

设备	D_1	D_2	D_3
单价	30	15	20
可靠性	0.9	0.8	0.5

- $u_2 = 3, c_2 = 15, r_2 = 0.8,$
- $S^1 = \{(0.9, 30), (0.99, 60)\}$
- $\phi_2(1) = 0.8, \quad \phi_2(2) = 1-(1-0.8)^2 = 0.96,$

$\phi_2(3) = 1-(1-0.8)^3 = 0.992$

$S^2_1 = \{(0.9*0.8, 30+15), (0.99*0.8, 60+15)\}$

$= \{(0.72, 45), (0.792, 75)\}$

$S^2_2 = \{(0.9*0.96, 30+15*2), \cancel{(0.99*0.96, 60+15*2)}\}$

$= \{(0.864, 60)\}$

$S^2_3 = \{(0.9*0.992, 30+15*3), \cancel{(0.99*0.992, 60+15*3)}\}$

$= \{(0.8928, 75)\}$

$S^2 = \{(0.72, 45), \cancel{(0.792, 75)}, (0.864, 60), (0.8928, 75)\}$

如果第二级设备配备2台，第三级设备不能购买

设备	D_1	D_2	D_3
单价	30	15	20
可靠性	0.9	0.8	0.5

- $u_3 = 3$, $c_3=20$, $r_3=0.5$,
- $S^2=\{(0.72, 45), (0.864, 60), (0.8928, 75)\}$
- $\phi_3(1)=0.5$, $\phi_3(2)=1-(1-0.5)^2=0.75$,

  $\phi_3(3)=1-(1-0.5)^3=0.875$

  $S^3{}_1=\{(0.72*0.5, 45+20), (0.864*0.5, 60+20),$

  $(0.8928*0.5, 75+20)\}$

  $=\{(0.36, 65), (0.432, 80), (0.4464, 95)\}$

  $S^3{}_2=\{(0.72*0.75, 45+20*2), (0.864*0.75, 60+20*2),$

  $\cancel{(0.8928, 75+20*2)}\}=\{(0.54, 85), (0.648, 100)\}$

  $S^3{}_3=\{(0.72*0.875, 45+20*3)\}=\{(0.63, 105)\}$

$S^3=\{(0.36, 65), (0.432, 80), \cancel{(0.4464, 95)}, (0.54, 85), (0.648, 100), \cancel{(0.63,105)}\}$

中国科学院大学 University of Chinese Academy of Sciences

5.6 可靠性设计

- 由S^n, S^{n-1}, …, S^1回溯求m_n, m_{n-1}, ……, m_1
- 判断由S^n中f最大的序偶来自哪个S^n_j, 则n级设备上的设备数量为j.
- $S^3=\{(0.36, 65), (0.432, 80), (0.54, 85), (0.648, 100)\}$
- $(0.648, 100) \in S^3_2$, 所以$m_3=2$

$S^3_2=\{(0.72*0.75, 45+20*2), (0.864*0.75, 60+20*2)\}$

$=\{(0.54, 85), (0.648, 100)\}$

$(0.864, 60) \in S^2_2$, 所以$m_2=2$

$S^2_2=\{(0.9*0.96, 30+15*2)\}=\{(0.864, 60)\}$

$(0.9, 30) \in S^1_1$, $m_1=1$

第五章 动态规划

- 5.1 一般方法
- 5.2 多段图问题
- 5.3 每对结点之间的最短路径
- 5.4 最优二分检索树
- 5.5 0/1背包问题
- 5.6 可靠性设计
- 5.7 货郎担问题
- 5.8 流水线调度问题

5.7 货郎担问题

■ 1. 问题描述

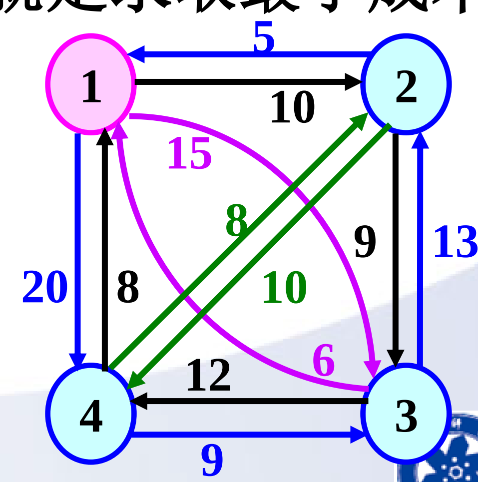
- 货郎担问题也叫推销商问题(traveling salesman problem), 其一般提法为：有n个城市，城市之间的距离已知，有一个货郎从城1出发到其他城市一次且仅一次，最后回到城市1，怎样选择行走路线使总路程最短？

5.7 货郎担问题

■ 1. 问题描述

□抽象化描述

- ➢G =(V, E)是一个有向图，边的成本为C_{ij}，若<i, j>不属于E，则$c_{ij} = \infty$ 。
- ➢G的一条周游路线是包含V中每个节点的有向环。周游路线的成本是此路线上所有边的成本和。
- ➢货郎担问题就是求取最小成本的周游路线问题。



5.7 货郎担问题

■ 1. 问题描述

- 邮路问题
  - 假定有一辆邮车要到n个不同的地点收集邮件，这种情况可以用n+1个结点的图来表示。
  - 一个结点表示此邮车出发并要返回的那个邮局，其余的n个结点表示要收集邮件的n 个地点。
  - 由地点i到j的距离则由边<i, j>上所赋予的成本来表示。
  - 邮车所经的路线是一条周游路线，希望求出具有最小长度的周游路线。

5.7 货郎担问题

- ■ 1. 问题描述
  - □机械手运动问题
    - ➢在一条装配线上用一个机械手去紧固待装配部件上的螺帽。
    - ➢机械手由初始位置(该位置在第一个要紧固的螺帽上方)开始，依次移动到其余的每一个螺帽，最后返回到初始位置。
    - ➢机械手移动的路线就是以螺帽为结点的一个图中的一条周游路线。
    - ➢一条最小成本路线将使这机械手完成其工作所用的时间取最小值。

5.7 货郎担问题

- **2. 货郎担问题满足最优性原理**
  - 假设周游路线是开始于结点1并终止于结点1的一条简单路径。
  - 每一条周游路线都由一条边<1, k>和一条由结点k到结点1的路径所组成，其中k∈V-{1};
  - 而这条由结点k到结点1的路径通过V-{1, k}的每个结点各一次。
  - 容易看出，如果这条周游路线是最优的，那么这条由k到1的路径必定是通过V-{1, k}中所有结点的由k到1的最短路径，
  - 因此最优性原理成立。

5.7 货郎担问题

■ 2. 货郎担问题满足最优性原理

设g(i, S)是由结点i开始, 通过S中的所有结点, 在结点1终止的一条最短路径的长度,

g(1, V−{1})是一条最优的周游路线长度

于是可以得到: $g(1, V-\{1\})=\min_{2\le k\le n}\{c_{1k}+g(k, V-\{1, k\})\}$

上式一般化可得: $g(i, S)=\min_{j\in S}\{c_{ij}+g(j, S-\{j\})\}$

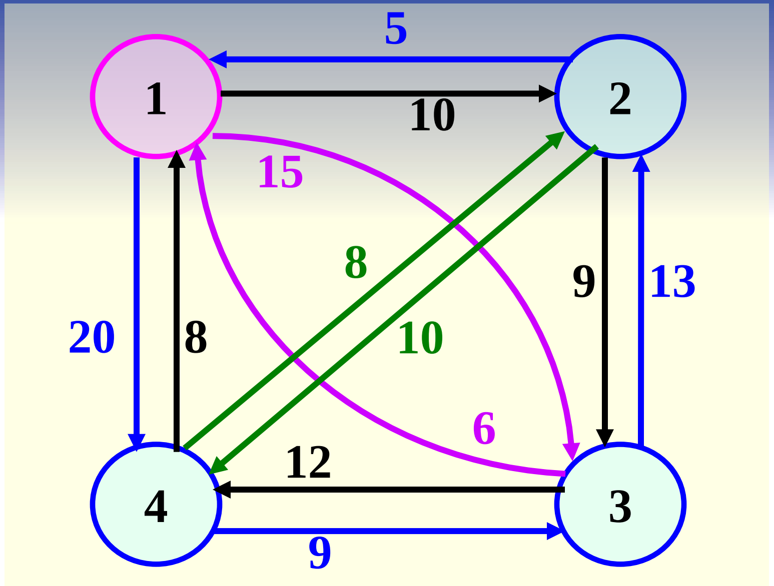
5.7 货郎担问题

- 2. 货郎担问题的求解过程
  - $g(i, \varnothing)=c_{i1}$,
  - 计算 $|S|=0$时， $g(i, S)= g(i, \varnothing)=c_{i1}$

    $|S|=1$时， $g(i, S)$

    ……

    $|S|=n-1$时， $g(1, V-\{1\})$



	1	2	3	4
1	0	10	15	20
2	5	0	9	10
3	6	13	0	12
4	8	8	9	0

g(i, S)表示由结点i经过S中所有结点到结点1的最短路线长度

|S|=0

- $g(2, \varnothing)=c_{21}=5$
- $g(3, \varnothing)=c_{31}=6$
- $g(4, \varnothing)=c_{41}=8$

|S|=1

- $g(2, \{3\})=c_{23}+g(3, \varnothing)=9+6=15$
- $g(2, \{4\})=c_{24}+g(4, \varnothing)=10+8=18$
- $g(3, \{2\})=c_{32}+g(2, \varnothing)=13+5=18$
- $g(3, \{4\})=c_{34}+g(4, \varnothing)=12+8=20$
- $g(4, \{2\})=c_{42}+g(2, \varnothing)=8+5=13$
- $g(4, \{3\})=c_{43}+g(3, \varnothing)=9+6=15$

$$g(i,S)=\min_{j\in S}\{c_{ij}+g(j,S-\{j\})\}$$

|S|=2

- $g(2, \{3, 4\})$

$=\min\{c_{23}+g(3, \{4\}), c_{24}+g(4, \{3\})\}$

$=\min\{9+20, 10+15\} = 25$

- $g(3, \{2, 4\})$

$=\min\{c_{32}+g(2, \{4\}), c_{34}+g(4, \{2\})\}$

$=\min\{13+18, 12+13\} = 25$

- $g(4, \{2, 3\})=\min\{c_{42}+g(2, \{3\}), c_{43}+g(3, \{2\})\}$

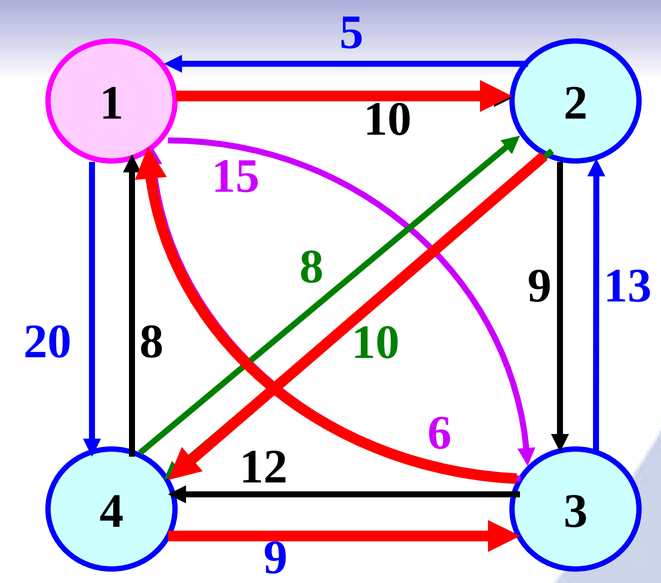
$=\min\{8+15, 9+18\} = 23$

- $g(1, \{2, 3, 4\})$

$=\min\{c_{12}+g(2, \{3, 4\}), c_{13}+g(3, \{2, 4\}), c_{14}+g(4, \{2, 3\})\}$

$=\min\{10+25, 15+25, 20+23\} = 35$

➢可得这条最优周游路线是: 1 → 2 → 4 → 3 → 1



$$g(i,S)=\min_{j\in S}\{c_{ij}+g(j,S-\{j\})\}$$

5.7 货郎担问题

■ 3. 算法性能分析

- □计算时间为$\Theta(n^2 2^n)$
- □货郎担问题当城市数目增加时，用动态规划方法求解，无论是计算量还是存储量都会大大增加，所以本方法只适用于n较小的情况。

|S|=0时， g(i, S)= g(i, ∅) (n-1) *C(n-2, 0)

|S|=1时， g(i, S) (n-1) *C(n-2, 1)

i 的个数为 n-1

……

|S|=n-3时， g(i, S) (n-1)*C(n-2, n-3)

|S|=n-2时， g(i, S) (n-1)*C(n-2, n-2)

|S|=n-1时， g(1, V-{1})

因此， g(i, S)的个数为 Σ (n-1)*C(n-2, k)=$(n-1)2^{n-2}$

第五章 动态规划

- 5.1 一般方法
- 5.2 多段图问题
- 5.3 每对结点之间的最短路径
- 5.4 最优二分检索树
- 5.5 0/1背包问题
- 5.6 可靠性设计
- 5.7 货郎担问题
- 5.8 流水线调度问题

5.8 流水线调度问题

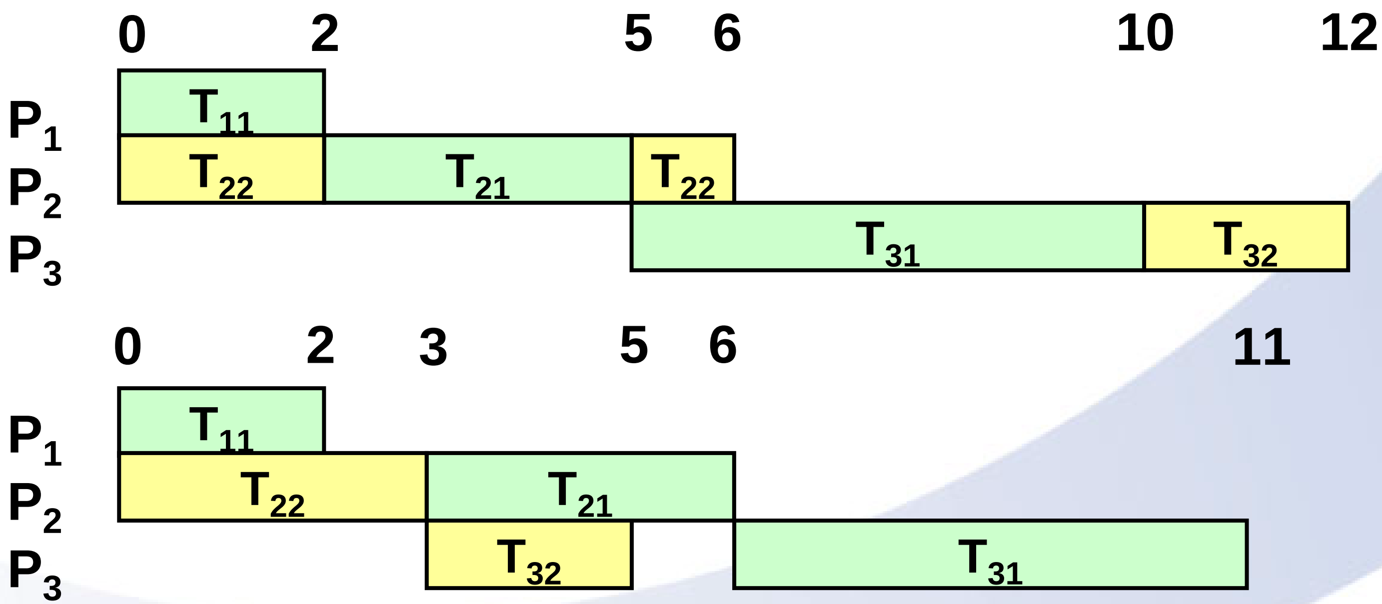
- 问题描述：
  - 大型作业往往由一系列任务组成
  - n个作业要在m台设备组成的流水线上完成加工，每个作业加工的顺序都是先在P_1上加工，然后在P_2上加工，……，最后在P_m上加工
  - 每个任务$T_{j,i}$，$1≤j≤m$，$1≤i≤n$
    - $T_{j,i}$只能在P_j上执行
    - 任何时刻在同一台设备上不能同时处理一个以上的任务
    - $T_{j-1,i}$完成后，$T_{j,i}$才能开始执行，每个任务的执行时间$t_{j,i}$
  - 如何将n×m个任务分配给m台设备，使得n个作业完成？

5.8 流水线调度问题

$$J=\begin{bmatrix}2 & 0\\ 3 & 3\\ 5 & 2\end{bmatrix}$$

■ 问题描述：

- □例：考虑在三台设备上调度两个作业，每个作业包含三项任务，完成这些任务要求的时间由矩阵J给出。这两个作业的两种可能的调度如下：

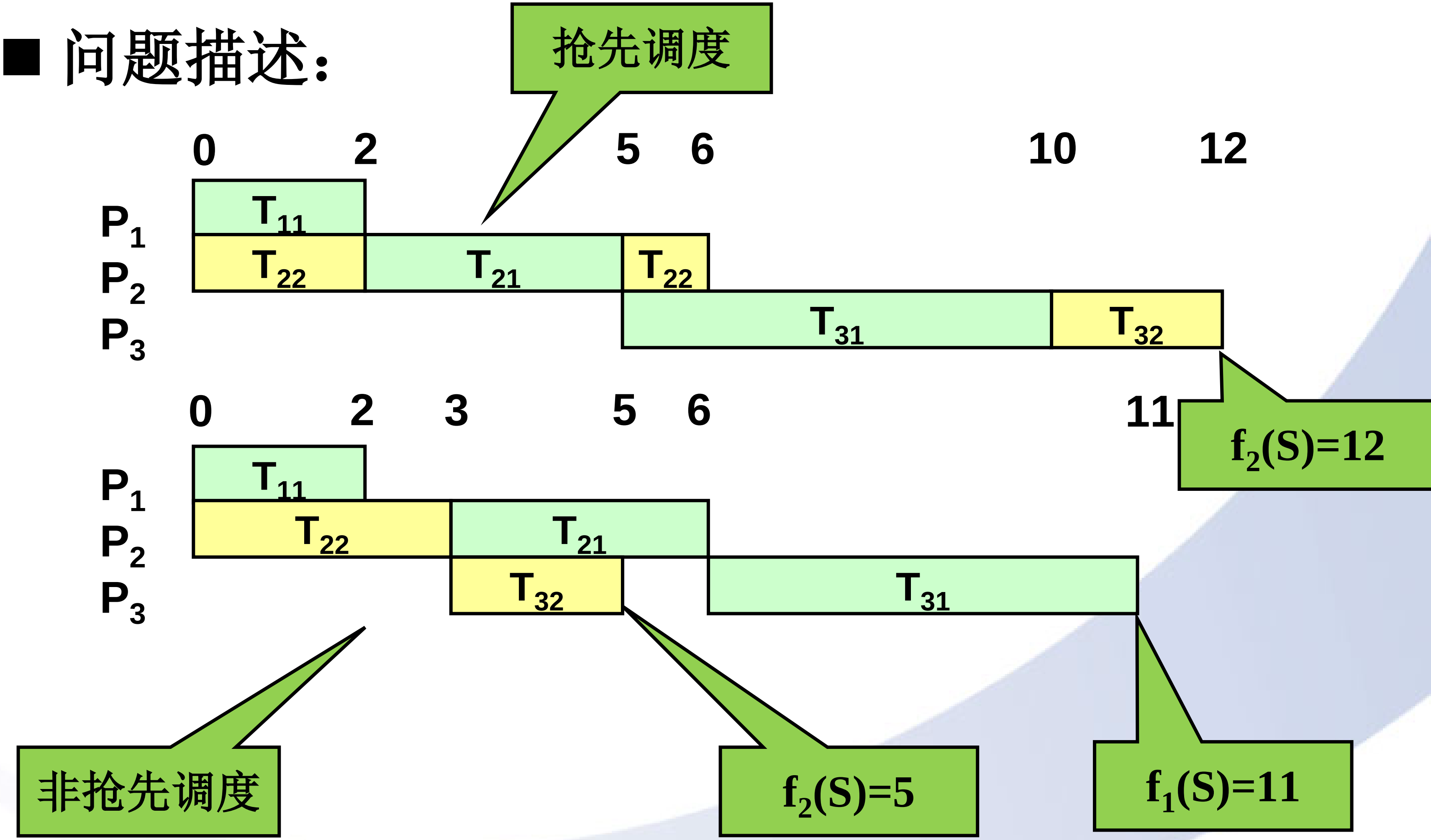


5.8 流水线调度问题

- 问题描述：
  - 非抢先调度：
    - ➢在任何一台设备处理一个任务，一直到它完成才能处理另一个任务，不允许被中断。
  - 抢先调度：
    - ➢设备在处理某任务时被中断，而优先处理优先级更高的任务。
  - 作业i的完成时间$f_i(S)$：
    - ➢在S调度方案下作业i的所有任务得以完成的时间。

5.8 流水线调度问题

■ 问题描述：



5.8 流水线调度问题

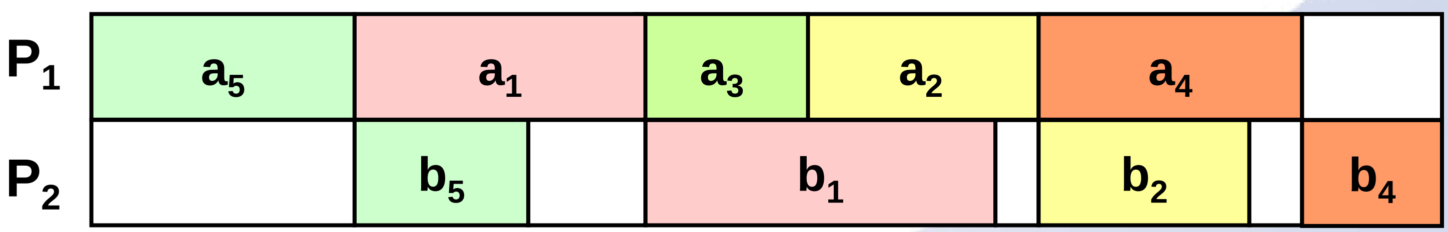
- 问题描述：
  - F(S)：调度S的完成时间 $F(S)=\max_{1\le i\le n}\{f_i(S)\}$
  - MFT(S)：平均流动时间 $MFT(S)=\frac{1}{n}\sum_{1\le i\le n} f_i(S)$
  - OFT：一组给定作业的最优完成时间OFT调度是一种非抢先调度S，它对所有非抢先调度而言F(S)的值最小。
  - POFT：抢先调度下最优完成时间
  - OMFT：最优平均完成时间
  - POMFT：抢先调度下最优平均 完成时间
  - 当m>2时，得到OFT和POFT的调度的一般问题是难于计算的问题。

5.8 流水线调度问题

- 最优的非抢先调度方案设计(m=2)
  - 直观上，一个最优调度应使机器P_1没有空闲时间，且机器P_2的空闲时间最少。
  - a_i表示t_{1i}，b_i表示t_{2i}
  - 在两台设备上按同样的顺序处理作业，不比分别采用不同的处理次序处理作业花费更多的时间
  - 如果允许有$a_i=0$的作业，那么最优调度可通过下法构造出来：
    - 首先对于所有$a_i \neq 0$的作业求出一种最优调度的排列，
    - 然后把所有的$a_i=0$的作业以任意次序加到这一排列的前面。

5.8 流水线调度问题

- 最优的非抢先调度方案设计(m=2)
  - 下图的调度由作业的排列次序5, 1, 3, 2, 4确定。
  - 为讨论方便，假定$a_i \neq 0$, $1 \leq i \leq n$。
  - 最优调度的排列满足最优性原理：
    - 在给出了这个排列的第一个作业后，
    - 剩下的排列相对于这两台设备在完成第一个作业时所处的状态而言是最优的。



5.8 流水线调度问题

- 最优的非抢先调度方案设计(m=2)
  - 建立递推关系
    - 假设对作业1, 2, …, k的一种调度排为 $\sigma_1, \sigma_2, \cdots, \sigma_k$。
    - h_l和h_2分别是在设备P_1和P_2上完成作业1, 2…, k的时间, $t=h_2-h_l$ 。
    - 在对作业1, 2, …, k作了一系列决策后，这两台设备所处的状态可完全由t确定。
    - t的含义：
      - ✓如果要在设备P_1和P_2上处理后面的作业，则必须在这两台设备同时处理前k个作业的不同任务后，设备P_2还要用大小为t的时间段处理前k个作业中没处理完的任务，
      - ✓即在t这段时间及其以前，设备P_2不能用来处理别的作业的任务。

5.8 流水线调度问题

- **最优的非抢先调度方案设计(m=2)**
  - 建立递推关系
    - 设g(S, t)是在状态t下调度方案S的最优调度长度。
    - 作业集合{1, 2, …, *n*}的最优长度是g({1, 2, …, *n*}, 0)
    - 递推关系

$$g(\{1,2,\cdots,n\},0)=\min_{1\le i\le n}\{a_i+g(\{1,2,\cdots,n\}-\{i\},b_i)\}$$

    - 一般递推关系

$$g(S,t)=\min_{i\in S}\{a_i+g(S-\{i\},b_i+\max\{t-a_i,0\})\}$$

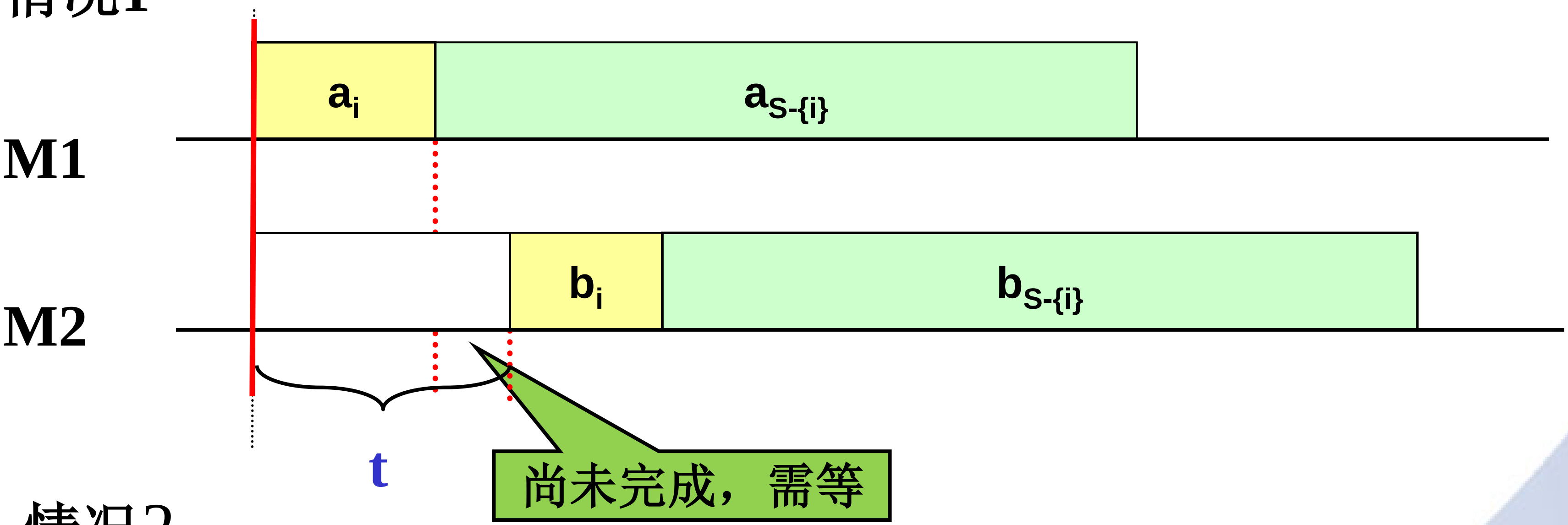
    - 结束条件

$$g(\Phi,t)=\max\{t,0\}$$

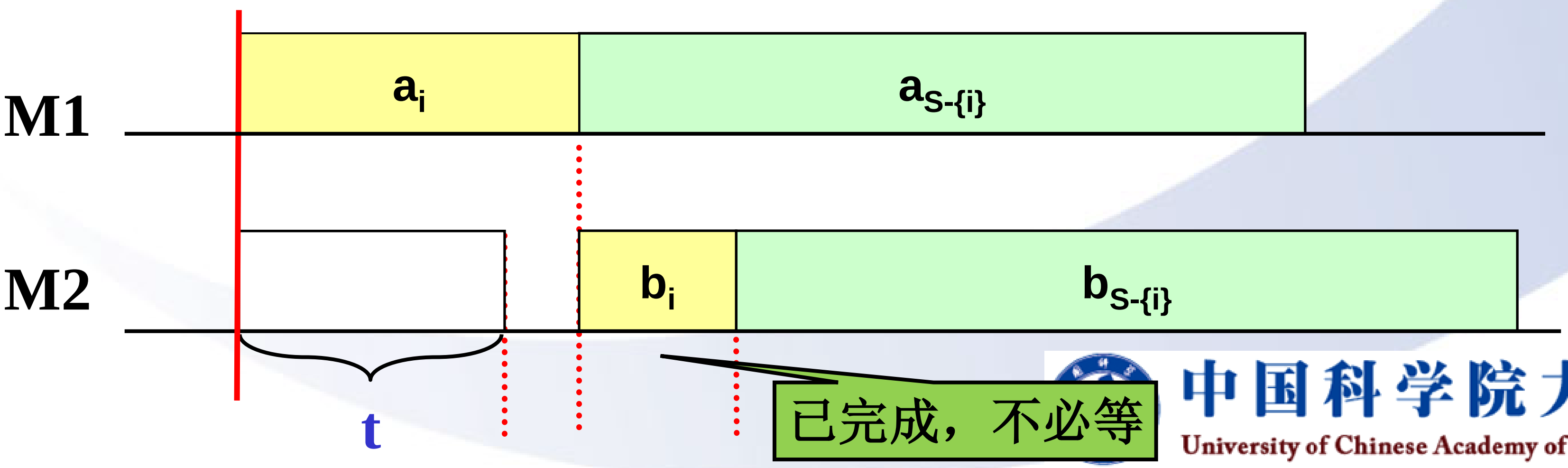
  - 时间分析
    - 时间O(2^n)

5.8 流水线调度问题

情况1



情况2



5.8 流水线调度问题

■ 流水线作业调度的Johnson法则

□设i和j是S的调度R中排在前面的两个作业

$g(S, t)$

$=a_i+g(S-\{i\}, \boxed{t'})$

$= a_i+a_j+g(S-\{i, j\}, b_j+\max\{\boxed{t'}-a_j, 0\})$

$= a_i+a_j+g(S-\{i, j\}, t_{ij})$

作业i和j，若a_i, a_j, b_i, b_j中a_i最小，则首先处理作业i更优；若b_j最小，则最后处理作业j更优。

□将作业i, j在R中易位

$g'(S, t)=a_i+a_j+g(S-\{i, j\}, t_{ji})$

$\min\{b_i, a_j\} \geq \min\{b_j, a_i\} \longrightarrow t_{ij} \leq t_{ji} \longrightarrow g(S, t) \leq g'(S, t)$

5.8 流水线调度问题

- 流水线作业调度的Johnson法则
  - 补充证明

$$
\begin{aligned}
t_{ij} &= b_j + \max\{b_i + \max\{t - a_i, 0\} - a_j, 0\} \\
&= b_j + b_i - a_j + \max\{\max\{t - a_i, 0\}, a_j - b_i\} \\
&= b_j + b_i - a_j + \max\{t - a_i, a_j - b_i, 0\} \\
&= b_j + b_i - a_j - a_i + \max\{t, a_i + a_j - b_i, a_i\}
\end{aligned}
$$

$$
t_{ji} = b_j + b_i - a_j - a_i + \max\{t, a_i + a_j - b_j, a_j\}
$$

5.8 流水线调度问题

- ■ 流水线作业调度的Johnson法则
  - □补充证明

若对t的所有取值：$\max\{t, a_i + a_j - b_i, a_i\} \le \max\{t, a_i + a_j - b_j, a_j\}$

则：$\max\{a_i + a_j - b_i, a_i) \le \max\{a_i + a_j - b_j, a_j)$

即：$a_i + a_j + \max\{-b_i, -a_j\} \le a_i + a_j + \max\{-b_j, -a_i\}$

或： $\min\{b_i, a_j\} \ge \min\{b_j, a_i\}$

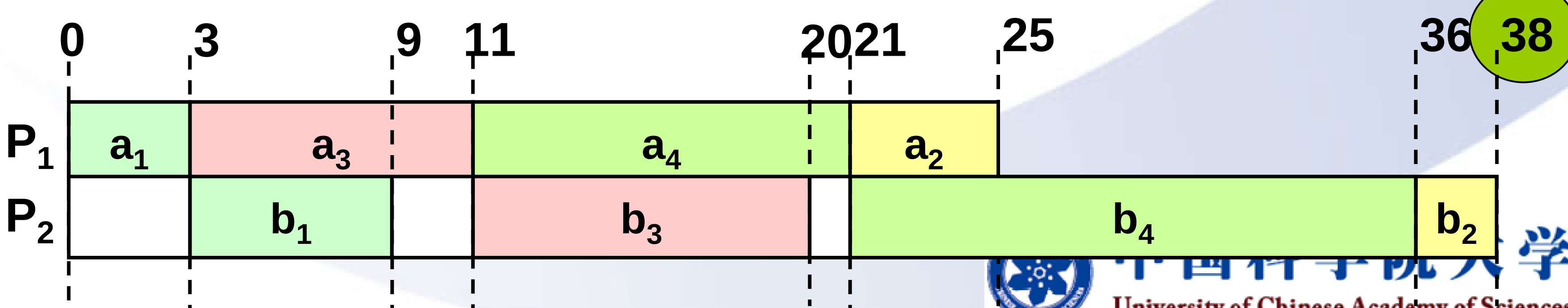
5.8 流水线调度问题

- 流水线作业调度的Johnson法则
  - 调度规则
    - 把全部a_i和b_j分类成非降序列。
    - 按照这一分类次序考察此序列：
      - 如果序列中下一个数是a_i且作业i还没调度，那么在还没使用的最左位置调度作业i；
      - 如果下个数是b_j且作业j还没调度，那么在还没使用的最右位置调度作业j。
      - 如果已经调度了作业，则转到该序列的下一个数。

- 例5.19 设n=4，$(a_1, a_2, a_3, a_4)=(3, 4, 8, 10)$，
  $(b_1, b_2, b_3, b_4)=(6, 2, 9, 15)$
- 求解：
  - 非降次序排列，获得序列
  - $(b_2, a_1, \cancel{a_2}, \cancel{b_1}, a_3, \cancel{b_3}, a_4, \cancel{b_4}) =(2, 3, 4, 6, 8, 9, 10, 15)$

调度序列

σ_1	σ_2	σ_3	σ_4
1	3	4	2



作业-课后练习18

■ 问题描述

- □某工业生产部门根据国家计划的安排，拟将某种高效率的5台机器，分别分配给A，B，C三个工厂，各工厂在获得不同数量的这种机器后，可以为国家盈利如下表所示。请找出一种5台机器的分配方式，使得这5台机器盈利最大。（15分）
- □2018年本课程的考试试题

工厂＼台数	0	1	2	3	4	5
A	0万	3万	7万	9万	12万	13万
B	0万	5万	8万	10万	11万	12万
C	0万	4万	6万	11万	12万	12万

作业-课后练习18

- 要求
  - 作业提交到课程网站上
  - Word文档即可

作业-算法实现6

- 问题
  - 输入：整数序列$a_1, a_2, \ldots, a_n$
  - 输出：序列的一个子段，其和$\sum_{k=i}^{j} a_k$最大
  - 注意：当所有整数都为负数时，定义最大子段和为0
- 要求
  - 作业提交到课程网站上
  - 用C(C++)或者matlab实现
  - 要有算法的求解说明

End