algorithm

6.贪心法a

贪心法的例子：活动选择问题

- 输入：$S = \{1, 2, \ldots, n\}$为n 项活动的集合，s_i，f_i 分别为活动 i 的开始和结束时间.
  - 活动 i 与 j 相容 $\Leftrightarrow$ $s_i \geq f_j$ 或 $s_j \geq f_i$.
- 求：最大的两两相容的活动集 A
- 输入实例：

i	1	2	3	4	5	6	7	8	9	10
s_i	1	3	2	5	4	5	6	8	8	2
f_i	4	5	6	7	9	9	10	11	12	13

- 解：$\{1, 4, 8\}$

贪心算法

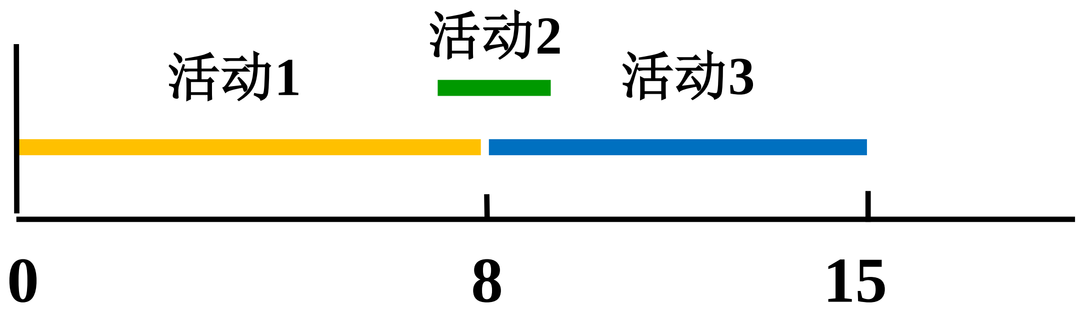
- 挑选过程是多步判断，每步依据某种“短视”的策略进行活动选择，选择时注意满足相容性条件.
- 策略1：开始时间早的优先
- 排序使 $s_1 \le s_2 \le \ldots \le s_n$，从前向后挑选
- 策略2：占用时间少的优先
- 排序使得 $f_1 - s_1 \le f_2 - s_2 \le \ldots \le f_n - s_n$，从前向后挑选
- 策略3：结束早的优先
- 排序使 $f_1 \le f_2 \le \ldots \le f_n$，从前向后挑选

策略1的反例

- 策略1：开始早的优先
- 反例：$S=\{1,2,3\}$
- $s_1=0,\ f_1=20,\ s_2=2,\ f_2=5,\ s_3=8,\ f_3=15$

策略2的反例

- 策略2： 占时少的优先
- 反例： $S = \{ 1, 2, 3 \}$
- s_1=0, f_1=8, s_2=7, f_2=9, s_3=8, f_3=15



策略3伪码

- 算法 Greedy Select
- 输入：活动集 S, s_i, f_i, $i = 1, 2, \ldots, n$, $f_1 \le \ldots \le f_n$
- 输出：$A \subseteq S$，选中的活动子集

1. $n \leftarrow length[S]$
2. $A \leftarrow \{1\}$
3. $j \leftarrow 1$
4. for $i \leftarrow 2$ to n do
5.   if $s_i \ge f_j$
6.   then $A \leftarrow A \cup \{ i \}$
7. $j \leftarrow i$
8. return A

- 完成时间 $t = \max \{ f_k : k \in A \}$

运行实例

- 输入： $S = \{ 1, 2, \ldots , 10 \}$

i	1	2	3	4	5	6	7	8	9	10
s_i	1	3	0	5	3	5	6	8	8	2
f_i	4	5	6	7	8	9	10	11	12	13

- 解： $A = \{1, 4, 8\}, t = 11$
- 时间复杂度　$O(n\log n) + O(n) = O(n\log n)$

- 如何证明该算法对所有的实例都得到正确的解?

贪心算法的特点

- 设计要素:
  - 贪心法适用于组合优化问题.
  - 求解过程是多步判断过程，最终的判断序列对应于问题的最优解.
  - 依据某种“短视的”贪心选择性质判断，性质好坏决定算法的成败.
  - 贪心法必须进行正确性证明.
  - 证明贪心法不正确的技巧：举反例.
- 贪心法的优势：算法简单，时间和空间复杂性低

贪心法正确性证明：活动选择

- 数学归纳法

例：证明对于任何自然数n，

$$1+2+...+n = n(n + 1)/2$$

证　n=1, 左边=1, 右边=1× (1+1)/2=1

假设对任意自然数 n 等式成立，则

$$1+2+...+ (n+1)$$

$$= (1+2+...+n) + (n+1)$$

$$= n(n+1)/2 + (n+1)$$

$$= (n+1)(n/2+1)$$

$$= (n+1)(n+2)/2$$

数学归纳法

- 第一数学归纳法
  - 适合证明涉及自然数的命题 $P(n)$
  - 归纳基础：证明$P(1)$为真 (或$P(0)$为真).
  - 归纳步骤：若对所有n有$P(n)$为真，证明$P(n+1)$为真

$$\forall n, P(n) \rightarrow P(n+1) \quad P(1)$$

$$n=1, \ P(1) \Rightarrow P(2)$$

$$n=2, P(2) \Rightarrow P(3)$$

$$\cdots$$

- 第二数学归纳法
  - 适合证明涉及自然数的命题 $P(n)$
  - 归纳基础：证明 $P(1)$为真 (或$P(0)$为真).
  - 归纳步骤：若对所有小于n 的 k 有 $P(k)$真，证明 $P(n)$为真

$$\forall k \ (\ k<n \wedge P(k)\ ) \rightarrow P(n) \quad P(1)$$

$$n=2, \ P(1) \Rightarrow P(2)$$

$$n=3, P(1) \wedge P(2) \Rightarrow P(3)$$

$$\cdots$$

两种归纳法的区别

- 归纳基础一样 $P(1)$为真
- 归纳步骤不同

- 证明逻辑

归纳法1： $P(1) \Rightarrow P(2) \Rightarrow P(3)\ldots$

归纳法2：

$$P(1) \Rightarrow P(2)$$

$$\left.\begin{array}{r} P(1) \\ P(2) \end{array}\right\} \Rightarrow P(3)$$

$$\left.\begin{array}{l} P(1) \\ P(2) \\ P(3) \end{array}\right\} \Rightarrow P(4)\ldots$$

算法正确性归纳证明

- 证明步骤:
  - 叙述一个有关自然数n的命题，该命题断定该贪心策略的执行最终将导致最优解.
  - 其中自然数 n 可以代 表算法步数或者问题规模 .
  - 证明命题对所有的自然数为真 .
  - 归纳基础(从最小实例规模开始)
  - 归纳步骤(第一或第二数学归纳法)

活动选择算法的命题

- 命题
- 算法 Select执行到第 k 步,选择 k 项活动

$$i_1 = 1,\ \ i_2,\ \ \ldots,\ i_k$$

- 则存在最优解 A包含活动 $i_1=1, i_2,\ldots, i_k$.
- 根据上述命题：对于任何 k，算法前 k 步的选择都将导致最优解，至多到第 n 步将得到问题实例的最优解

归纳证明：归纳基础

- 令 $S=\{1,2,\ldots,n\}$是活动集,且 $f_1\leq\ldots\leq f_n$
- 归纳基础: $k=1$, 证明存在最优解包含活动 1
- 证 任取最优解A，A中活动按截止时间递增排列.
- 如果A的第一个活动为 j，$j\neq 1$, 用1替换A的活动 j 得到解 A'，即 $A' = (A-\{j\})\cup\{1\}$,
- 由于 $f_1\leq f_j$, A' 也是最优解，且含有1.

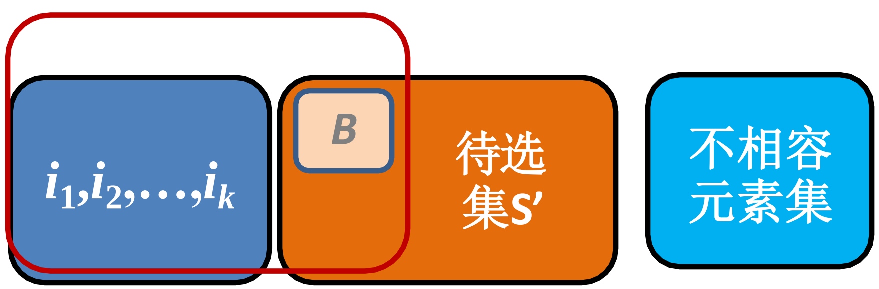
$f_1\leq f_j$

归纳步骤

- 假设命题对k为真,证明对k+1也为真.
- 证 算法执行到第 k 步, 选择了活动$i_1=1$, i_2, …, i_k, 根据归纳假设存在最优解 A包含$i_1=1,i_2,\ldots,i_k$, A中剩下活动选自集合S'

$$S' = \{\ i \mid i \in S,\ s_i \geq f_k \}$$

$$A = \{\ i_1,\ i_2,\ \ldots\ ,\ i_k\ \} \cup B$$

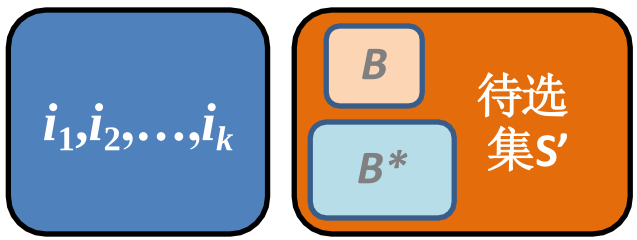


归纳步骤

- B是 S' 的最优解.
- 若不然, S' 的最优解为B^*, B^*的活动比B多，那么

$$B^* \cup \{1, i_2, \ldots, i_k\}$$

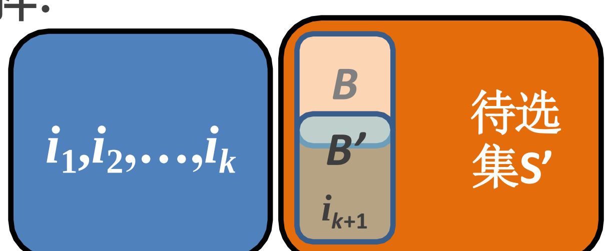
- 是 S 的最优解，且比 A的活动多,与 A 的最优性矛盾.



- 将S' 看成子问题，根据归纳基础，存在 S' 的最优解B' 有S' 中的第一个 活动 i_{k+1}, 且 $|B'| = |B|$, 于是

$$\{ i_1, i_2, \ldots, i_k \} \cup B' = \{ i_1, i_2, \ldots, i_k, i_{k+1} \} \cup (B' - \{ i_{k+1} \})$$

- 也是原问题的最优解.



小结

- 贪心法正确性证明方法：
  - 数学归纳法
  - 第一数学归纳法
  - 第二数学归纳法
- 活动选择问题的贪心法证明：
  - 叙述一个涉及步数的算法正确性命题
  - 证明归纳基础
  - 证明归纳步骤

最优装载问题

- 问题:
- n 个集装箱1, 2, … , n 装上轮船，集装箱 i 的重量 w_i, 轮船装载重量限制为C, 无体积限制.
- 问如何装使得上船的集装箱最多?
- 不妨设每个箱子的重量 $w_i \le C$.
- 该问题是0-1背包问题的子问题.
  - 集装箱相当于物品，物品重量是 w_i，价值v_i都等于1，
  - 轮船载重限制 C 相当于背包重量限制 b.

建模

- 设 $<x_1, x_2, \ldots, x_n>$ 表示解向量，$x_i = 0,1$,
- $x_i = 1$当且仅当第 i 个集装箱装上船

- 目标函数 $\max \sum_{i=1}^{n} x_i$

- 约束条件 $\sum_{i=1}^{n} w_i x_i \le C$

$$x_i = 0,1 \quad i = 1,2,\ldots,n$$

算法设计

- 贪心策略：轻者优先
- 算法设计：
- 将集装箱排序，使得

$$w_1 \leq w_2 \leq \ldots \leq w_n$$

- 按照标号从小到大装箱，直到装入下一个箱子将使得集装箱总重超过轮船装载重量限制，则停止.

正确性证明思路

- 命题：对装载问题任何规模为 n 的输入实例，算法得到最优解.
- 设集装箱从轻到重记为1, 2, ... , n.
- 归纳基础 证明对任何只含 1个箱子的输入实例，贪心法得到最优解. 显然正确.
- 归纳步骤
- 证明:
  - 假设对于任何n个箱子的输入实例贪心法都能得到最优解,
  - 那么对于任何n+1个箱子的输入实例贪心法也得到最优解.

归纳步骤证明思路

$N=\{1,2,...,n+1\}$, $w_1 \le w_2 \le ... \le w_{n+1}$

去掉箱子1，令 $C' = C - \{w_1\}$,

得到规模 n 的输入 $N' = \{2,3,...,n+1\}$

关于输入 N' 和C'的最优解 I'

在 I' 加入箱子1，得到 I

证明 I 是关于输入 N 的最优解

正确性证明

- 假设对于 n 个集装箱的输入，贪心法都可以得到最优解，考虑输入

$$N = \{ 1, 2, \dots , n+1 \}$$

- 其中 $w_1 \leq w_2 \leq \dots \leq w_{n+1}$.
- 由归纳假设，对于

$$N' = \{2, 3, \dots, n+1\}, \quad C' = C - w_1,$$

- 贪心法得到最优解 I'. 令

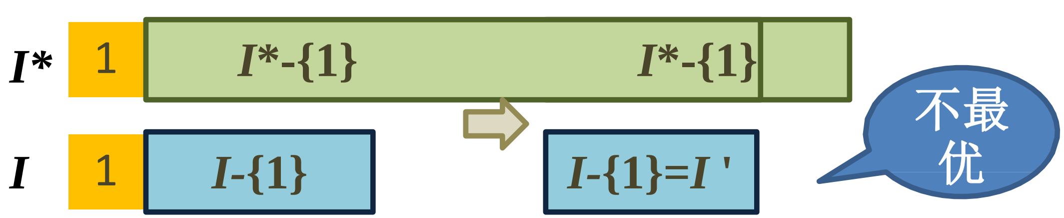
$$I = I' \cup \{1\}$$

正确性证明

- I (算法解)是关于 N 的最优解.
- 若不然，存在包含 1 的关于 N 的最优解 I^*(如果 I^* 中没有1，用 1 替换 I^* 中的第一个元素得到的解也是最优解), 且$|I^*|>|I|$; 那么$I^*-\{1\}$是 N' 和C' 的解且

$$|I^*-\{1\}|>|I-\{1\}|=|I'|$$

- 与 I'是关于N' 和C' 的最优解矛盾.



最小延迟调度问题

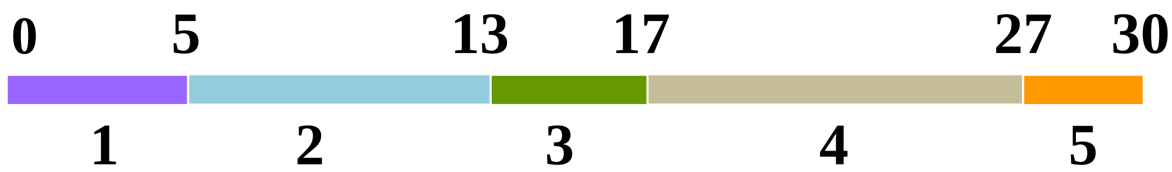
- 问题：
- 客户集合A，$\forall i \in A$，t_i 为服务时间，d_i 为要求完成时间，t_i，d_i 为正整数. 一个调度 $f: A \to \mathrm{N}$，$f(i)$为客户 i 的开始时间. 求
- 最大延迟达到最小的调度，即求 f 使得

$$\min_{f}\{\max_{i\in A}\{f(i)+t_i-d_i\}\}$$

$$\forall i, j \in A, i \neq j, f(i)+t_i \leq f(j) \ \text{or} \ f(j)+t_j \leq f(i)$$

实例：调度1

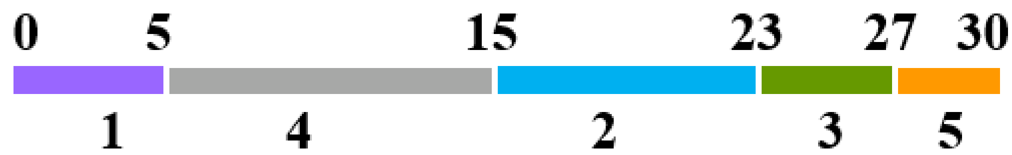
- $A = \{1, 2, 3, 4, 5\}$, $T = <5, 8, 4, 10, 3>$,
- $D = <10, 12, 15, 11, 20>$
- 调度1：顺序安排
- $f_1(1)=0$, $f_1(2)=5$, $f_1(3)=13$, $f_1(4)=17$, $f_1(5)=27$



- 各任务延迟：0, 1, 2, 16, 10;
- 最大延迟：16

更优的调度2

- $A = \{1, 2, 3, 4, 5\}$, $T = \langle 5, 8, 4, 10, 3 \rangle$,
- $D = \langle 10, 12, 15, 11, 20 \rangle$
- 调度2：按截止时间从前到后安排
- $f_2(1)=0$, $f_2(2)=15$, $f_2(3)=23$, $f_2(4)=5$, $f_2(5)=27$



- 各任务延迟：0, 11, 12, 4, 10;
- 最大延迟：12

贪心策略

- 贪心策略1：按照 t_i 从小到大安排
- 贪心策略2：按照 $d_i - t_i$ 从小到大安排
- 贪心策略3：按照 d_i 从小到大安排

- 策略1 对某些实例得不到最优解. 反例：$t_1=1, d_1=100, t_2=10, d_2=10$

- 策略2 对某些实例得不到最优解. 反例：$t_1=1, d_1=2, t_2=10, d_2=10$

策略3伪码

算法 Schedule

输入：A, T, D

输出：f

1. 排序 A 使得 $d_1 \leq d_2 \leq \ldots \leq d_n$
2. $f(1) \leftarrow 0$
3. $i \leftarrow 2$
4. while $i \leq n$ do
5. $\quad f(i) \leftarrow f(i-1) + t_{i-1}$
6. $\quad i \leftarrow i + 1$

设计思想：

按完成时间从早到晚安排任务，没有空闲.

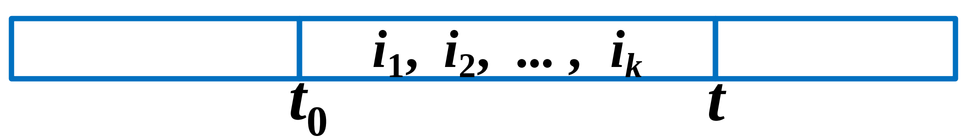
交换论证：正确性证明

- 证明思路:
- 分析一般最优解与算法解的区别（成分,排列顺序不同）
- 设计一种转换操作（替换成分或交换次序），可以在有限步将任意一个普通最优解逐步转换成算法的解
- 上述每步转换都不降低解的最优性质
- 贪心算法的解的性质:
- 没有空闲时间, 没有逆序.
- 逆序 (i, j): $f(i) < f(j)$ 且 $d_i > d_j$

引理

- 引理1 所有没有逆序、没有空闲时间的调度具有相同的最大延迟.
- 证: 设 f 没有逆序，在 f 中具有相同完成时间 d 的客户 $i_1, i_2, \ldots, i_k$ 连续安排, 其开始时刻为 t_0, 完成这些任务的时刻是 t，最大延迟为最后任务延迟 $t-d$, 与 $i_1, i_2, \ldots, i_k$ 的排列次序无关.

$$t = t_0 + (t_{i_1} + t_{i_2} + \ldots + t_{i_k})$$



证明要点

- 从一个没有空闲的最优解出发，逐步转变成没有逆序的解.根据引理 1，这个解和算法解具有相同的最大延迟.
- 如果一个最优调度存在逆序，那么存在 $i<n$ 使得$(i, i+1)$ 构成一个逆序，称为相邻的逆序.
- 交换相邻逆序 i 和 j, 得到的解仍旧最优.
- 每次交换后逆序数减 1 ，至多经过 $n(n-1)/2$ 次交换得到一个没有逆序的最 优调度——等价于算法的解.

交换相邻逆序仍旧最优

- 设 f_1是一个任意最优解，存在相邻逆序（ i, j ）. 交换 i 和 j 的顺序, 得到解 f_2. 那么 f_2的最大延迟不超过 f_1的最大延迟.
- 理由:

(1) 交换 i, j 与其他客户延迟时间无关

(2) 交换后不增加 j 的延迟，但可能增加 i 的延迟

(3) i 在 f_2的延迟小于 j 在 f_1 的延迟，因此小于f_1的最大延迟 r

i 在f_2的延迟不超过 j 在f_1 的延迟

$$delay(f_2,i)=s+t_j+t_i-d_i$$

$$delay(f_1,j)=s+t_i+t_j-d_j$$

$$d_j<d_i$$

$$delay(f_2,i) < delya(f_1,j) \le r$$

得不到最优解的处理方法

- 输入参数分析：
- 考虑输入参数在什么取值范围内使
- 用贪心法可以得到最优解

- 误差分析：
- 估计贪心法——近似算法所得到的解与最优解的误差（对所有的输入 实例在最坏情况下误差的上界）

找零钱问题

问题 设有 n 种零钱， 重量分别为$w_1, w_2, \ldots, w_n$, 价值分别为$v_1 = 1, v_2, \ldots, v_n$. 需要付的总钱数是 y .不妨设币值和钱数都为正整数.

问：如何付钱使得所付钱的总重最轻?

实例

$$v_1=1, v_2=5, v_3=14, v_4=18, w_i=1, i=1,2,3,4. \quad y=28$$

最优解：$x_3=2, x_1=x_2=x_4=0$，总重2

建模：令选用第 i 种硬币的数目是 x_i， $i = 1, 2, \ldots, n$

$$\min\{\sum_{i=1}^{n} w_i x_i\}$$

$$\sum_{i=1}^{n} v_i x_i = y, \quad x_i \in \mathrm{N}, \quad i = 1,2,\ldots,n$$

动态规划算法和贪心法比较

- 动态规划算法：设$F_k(y)$表示用前 k种零钱，总钱数为 y 的最小重量

$$F_k(y) = \min_{0 \le x_k \le \left\lfloor \frac{y}{v_k} \right\rfloor} \{F_{k-1}(y - v_k x_k) + w_k x_k\}$$

$$F_1(y) = w_1 \left\lfloor \frac{y}{v_1} \right\rfloor = w_1 y$$

- 贪心法：单位价值重量轻的货币优先，设

$$\frac{w_1}{v_1} \ge \frac{w_2}{v_2} \ge \ldots \ge \frac{w_n}{v_n}$$

- 使用前 k 种零钱，总钱数为 y，贪心法的总重为 $G_k(y)$，

$$G_k(y) = w_k \left\lfloor \frac{y}{v_k} \right\rfloor + G_{k-1}(y \bmod v_k) \quad k > 1$$

$$G_1(y) = w_1 \left\lfloor \frac{y}{v_1} \right\rfloor = w_1 y$$

n=1,2贪心法是最优解

- $n = 1$ 只有一种零钱，$F_1(y) = G_1(y)$
- $n = 2$，x_2越大，得到的解越好

$$F_2(y) = \min_{0 \le x_2 \le \lfloor y/v_2 \rfloor} \{F_1(y - v_2 x_2) + w_2 x_2\}$$

$$\begin{aligned}
&F_1(y - v_2(x_2+\delta)) + w_2(x_2+\delta)] - [F_1(y-v_2x_2) + w_2x_2] \\
&\quad = [w_1(y-v_2x_2-v_2\delta) + w_2 x_2+w_2\delta)] - [w_1(y-v_2x_2) + w_2x_2] \\
&\quad = -w_1v_2\delta + w_2\delta \\
&\quad = \delta(-w_1v_2 + w_2) \le 0
\end{aligned}$$

判别条件

- 定理对每个正整数 k，假设对所有非负整数 y 有$G_k(y) = F_k(y)$，且存在 p 和 δ 满足 $v_{k+1} = pv_k - \delta$,
- 其中 $0 \le \delta < v_k$，$v_k \le v_{k+1}$, p为正整数，
- 则下面的命题等价：

(1) $G_{k+1}(y) = F_{k+1}(y)$ 对一切正整数 y；

(2) $G_{k+1}(pv_k) = F_{k+1}(pv_k)$；

(3) $w_{k+1} + G_k(\delta) \le pw_k$.

几点说明

- 根据条件(1)与(3)的等价性，可以对 $k = 3, 4, \ldots, n$，依次利用条件(3)对贪心 法是否得到最优解做出判别.
- 条件(3)验证 1 次需 $O(k)$ 时间，$k = O(n)$，整个验证时间$O(n^2)$
- 条件(2)是条件(1) 在 $y = pv_k$ 时的特殊情况.
  - 若条件(1)成立，显然有条件(2) 成立.
  - 反之，若条件(2)不成立，则条件(1)不成立，钱数 $y = pv_k$ 恰好提供了一个贪心法不正确的反例.

验证实例

- 例 $v_1=1, v_2=5, v_3=14, v_4=18$, $w_i=1$, $i=1,2,3,4$. 对一切 y 有

$$G_1(y)=F_1(y), \quad G_2(y)=F_2(y).$$

验证 $G_3(y) = F_3(y)$

$$v_3 = pv_2 - \delta \Rightarrow p = 3, \delta = 1.$$

$$w_3+G_2(\delta)=1+1=2$$

$$pw_2=3\times1=3 \quad w_3+G_2(\delta)\le pw_2$$

贪心法对于 n=3 的实例得到最优解

验证实例

- 例 v_1=1,v_2=5,v_3=14,v_4=18, w_i=1, i=1,2,3,4.
- 对一切 y 有 $G_1(y)=F_1(y), G_2(y)=F_2(y), G_3(y)=F_3(y)$
- 验证 $G_4(y) = F_4(y)$

$$v_4 = pv_3 - \delta \Rightarrow p=2, \delta=10$$

$$w_4+G_3(\delta)=1+2=3$$

$$pw_3=2\times1=2$$

$$w_4 + G_3(\delta) > pw_3$$

n=4, $y=pv_3$=28,

- 最优解: x_3=2, 贪心法: x_4=1, x_2=2

小结

- 贪心策略不一定得到最优解，在这种情况下可以有两种处理方法：
  - 参数化分析：分析参数取什么值可得到最优解
  - 估计贪心法得到的解在最坏情况下与最优解的误差
- 一个参数化分析的例子：找零钱问题