贪心算法

陈长建

计算机科学系

期中总结 – 期中考试

- 目的
  - 考核不是主要目的
  - 让大家对当前所学知识进行自我检验
- 总结
  - 大家对分治、动态规划的求解掌握比较好
  - 对分治、动态规划思想的灵活应用还需要提高（这也是本门课最重要的目的之一）

期中总结 – 小班讨论

- 大家准备都很充分，讲解也很清晰
- 特别好的地方
  - 不少同学利用分治思想尝试新的解法
  - 有些同学能对现有的方法进一步改进
  - 针对算法特点，尝试算法之外的一些优化（比如缓存）

- 小班讨论
  - 一等奖（1组）：键盘 IKBC W200
  - 二等奖（2组）：鼠标（罗技M330）
  - 三等奖（3组）： 32G U盘

期中总结 – 课堂回答问题

- 奖品不是最终目的，知识才是
  - Bonus，额外奖励
- 目前情况：
  - 第五名：2次
- 还有空间

- 课堂回答问题奖励（5位）： 小礼物

第二次小班讨论

- 14周进行
  - 算法分析题 3-2、3-3(要求：有ppt代码演示)
  - 算法实现题 3-2、3-11(要求：有ppt和代码演示)
  - 数学之美分主题 2个(要求：有ppt)

    （1）P111 第12章 有限状态机和动态规划、P227 第26章 维特比和他的维特比算法

    （2）第31章 大数据的威力

    每组讲解时间：10-12分钟+3-5分钟提问讨论

第三次上机实验

- **院楼103，12月1日**上午8:30-12:00（现场验收）
- 第二次实验离线题（离线准备）
  - 1. 用Dijkstra贪心算法求解单源最短路径问题
  - 2. 收集样本问题（实现题3-15）
  - 3. 字符串比较问题（实现题3-17）
- 在线题
  - acm.hnu.edu.cn

回顾 - 贪心算法的基本要素

- 1. 贪心选择性质
- 2. 最优子结构性质

1. 贪心选择性质

- 贪心选择性质：所求问题的整体最优解可以通过一系列局部最优的选择，即贪心选择来达到。这是贪心算法可行的第一个基本要素，也是贪心算法与动态规划算法的主要区别。
- 动态规划算法通常以自底向上的方式解各子问题，每步所作的选择依赖于相关子问题的解。贪心算法仅在当前状态下作出局部最优选择，再去解作出这个选择后产生的相应的子问题。

分析

- 对于一个具体问题，要确定它是否具有贪心选择的性质，我们必须证明**每一步所作的贪心选择最终能够导致问题的最优解。**
- 通常可以首先证明问题的一个整体最优解是从贪心选择开始的，而且作了贪心选择后，原问题简化为一个规模更小的类似子问题。然后，用反证法、数学归纳法等证明，通过每一步作贪心选择，最终可得到问题的一个整体最优解。
- 其中，证明贪心选择后的问题简化为规模更小的类似子问题的关键在于利用该问题的最优子结构性质。

2. 最优子结构性质

- 当一个问题的最优解包含其子问题的最优解时，称此问题具有最优子结构性质。
- 问题的最优子结构性质是该问题可用动态规划算法或贪心算法求解的**关键特征。**

例1 - 活动安排问题

- 活动安排问题：要求高效地安排一系列争用某一公共资源的活动。
- 举例：

活动序号	1	2	3	4	5	6	7	8	9	10	11
起始时间	1	3	0	5	3	5	6	8	8	2	12
结束时间	4	5	6	7	8	9	10	11	12	13	14

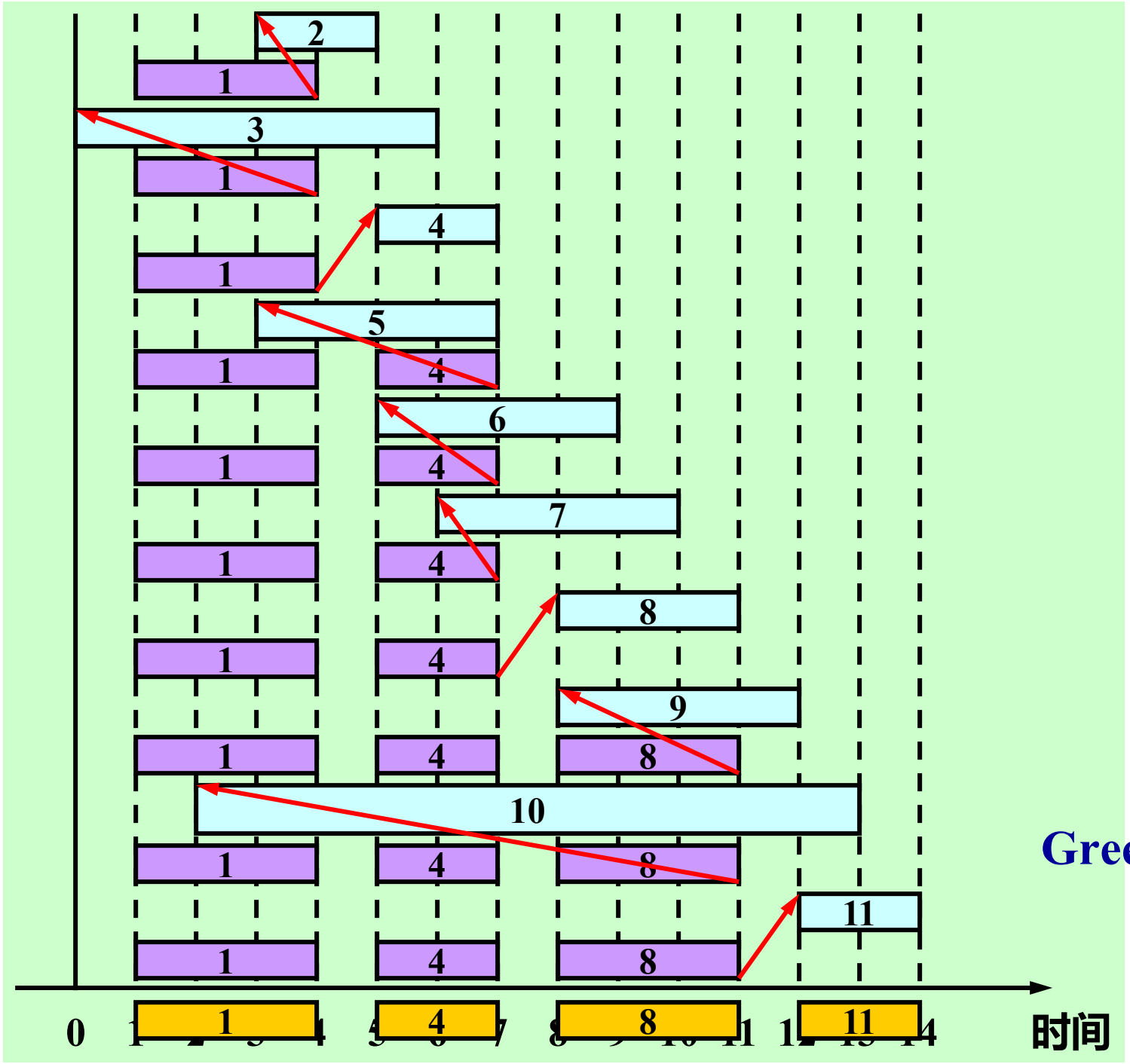


图4-1 算法GreedySelector的计算过程

补充说明

- 贪心算法并不总能求得问题的整体最优解。
- 但对于活动安排问题，贪心算法GreedySelector却总能求得的整体最优解，即它最终所确定的相容活动集合a的规模最大。这个结论可以用数学归纳法证明。

用数学归纳法证明活动安排问题

- 设集合E={1，2，…，n}为所给的活动集合。由于E中活动按结束时间的非减序排列，故活动1有最早完成时间。
- 证明I：活动安排问题有一个最优解以贪心选择开始，即该最优解中包含活动1。
- 证明II：对集合E中所有与活动1相容的活动进行活动安排求得最优解的子问题。

证明I：活动安排问题有一个最优解以贪心选择开始，即该最优解中包含活动1。

设$A \subseteq E$是所给的活动安排问题的一个最优解，

且A中活动也按结束时间非减序排列，A中第一个活动为活动k。

若k=1，则A就是一个以贪心选择开始的最优解。

若k>1，则设 $B = A - \{k\} \cup \{1\}$

由于$f_1 \leq f_k$，且A中的活动是相容的，故B中的活动也是相容的。

结论：由于B中活动个数与A中活动个数相同，且A是最优的，故B也是最优的。总存在以贪心选择开始的最优活动方案。

贪心选择性质

证明II：对E中所有与*活动1相容的活动*进行活动安排求得最优解的子问题。

- 即需证明：若A是原问题的最优解，则$A'=A-\{1\}$是活动安排问题$E'=\{i\in E: s_i \geq f_1\}$的最优解。
- 如果能找到E'的一个最优解B'，它包含比A'更多的活动，则将活动1加入到B'中将产生E的一个解B，它包含比A更多的活动。这与A的最优性矛盾。
- 结论：每一步所做的贪心选择问题都将问题简化为一个更小的与原问题具有相同形式的子问题。

最优子结构性质

证明III：每次选最小结束时间的活动，定可得到最优解。

引理3. 设 $S=\{1, 2, \cdots, n\}$ 是 n 个活动集合，$f_0=0$，l_i 是 $S_i=\{ j\in S \mid s_j \geq f_{i-1}\}$ 中具有最小结束时间 f_{l_i} 的活动. 设A是S的包含活动1的优化解，其中

$f_1 \leq \cdots \leq f_n$，则 $A=\bigcup_{i=1}^{k}\{l_i\}$

证. 对$|A|$作归纳法.

当$|A|=1$时，由引理1，命题成立.

设$|A|<k$时，命题成立.

当$|A|=k$时，由引理2，$A=\{1\}\cup A_1$，

A_1是 $S_2=\{ j\in S \mid s_j \geq f_1 \}$ 的优化解.

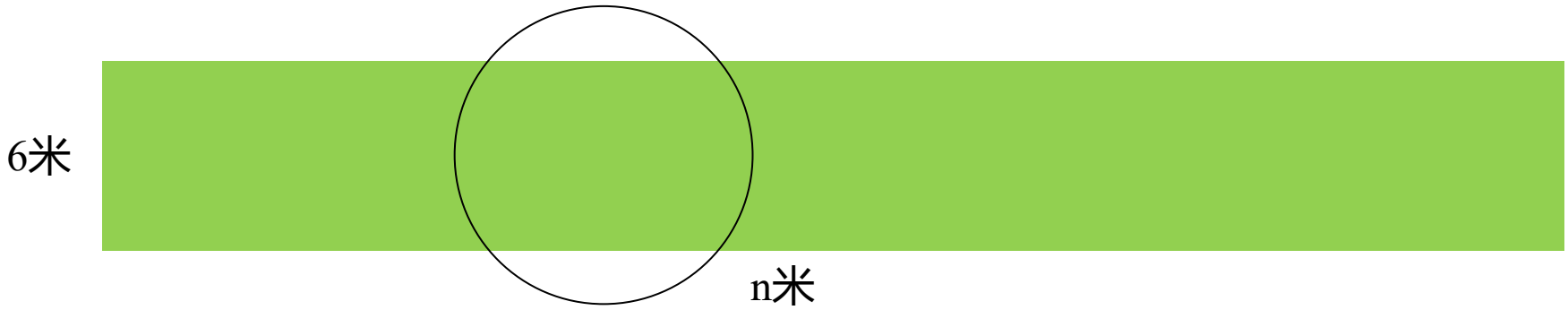
由归纳假设，$A_1=\bigcup_{i=2}^{k}\{l_i\}$

于是，$A=\bigcup_{i=1}^{k}\{l_i\}$.

贪心选择性质

练习 – 喷水装置

- 现有一块草坪，长为n米，宽为6米。要在横中心线上放置覆盖半径为5米的喷水装置。现在要求选择尽量少的喷水装置，能把整个草坪全部湿润



练习 – 喷水装置

- 现有一块草坪，长为n米，宽为6米。要在横中心线上放置覆盖半径为5米的喷水装置。现在要求选择尽量少的喷水装置，能把整个草坪全部湿润

- 贪心做法：第一个放在左起4米位置，第二个放在左起12米位置，…

贪心算法与动态规划算法的差异

- 贪心算法和动态规划算法都要求问题具有最优子结构性质，这是两类算法的一个共同点。
- 但是，对于具有最优子结构的问题应该选用贪心算法还是动态规划算法求解？是否能用动态规划算法求解的问题也能用贪心算法求解？

例2- 0-1背包问题（动态规划）

- 给定n种物品和一个背包。物品i的重量是w_i，其价值为v_i，背包的容量为c。应如何选择装入背包的物品，使得装入背包中物品的总价值最大?
- 在选择装入背包中的物品时，对每种物品i只有两种选择：即装入或不装入背包。不能将物品i装入背包多次，也不能只装入部分的物品i。因此，该问题称为0-1背包问题。

问题的形式化描述

- 此问题的形式化描述为，给定c>0,w_i>0,v_i>0,1≤i≤n，要求找出一个n元0-1向量 (x_1,x_2, .. ,x_n)，其中x_i∈{0,1}，使得对w_ix_i求和小于等于c，并且对v_ix_i求和达到最大。
- 因此，0-1背包问题是一个特殊的整数规划问题。

$$\max \sum_{i=1}^{n} v_i x_i$$

$$\begin{cases} \sum_{i=1}^{n} w_i x_i \le c \\ x_i \in \{0,1\}, \quad 1 \le i \le n \end{cases}$$

0-1背包问题的子问题递归关系

- 设所给0-1背包问题的子问题：

$$\max \sum_{k=i}^{n} v_k x_k$$

$$\begin{cases} \sum_{k=1}^{n} w_k x_k \le j \\ x_k \in \{0,1\}, \quad i \le k \le n \end{cases}$$

m(i,j)是背包容量为j，可选择物品为i,i+1,…,n时0-1背包问题的最优值。由于0-1背包问题的最优子结构性质，可以建立计算m(i,j)的递归式如下：

$$m(i,j) = \begin{cases} \max\{m(i+1,j), m(i+1,j-w_i)+v_i\} & j \ge w_i \\ m(i+1,j) & 0 \le j \le w_i \end{cases}$$

$$m(n,j) = \begin{cases} v_n & j \ge w_n \\ 0 & 0 \le j \le w_n \end{cases}$$

贪心算法解背包问题的基本步骤

- 假如有：钻石、银元、梨，怎么装?
- 首先计算每种物品单位重量的价值v_i/w_i；
- 然后，依贪心选择策略，将尽可能多的单位重量价值最高的物品装入背包。
- 若将这种物品全部装入背包后，背包内的物品总重量未超过c，则选择单位重量价值次高的物品并尽可能多地装入背包。依此策略一直地进行下去，直到背包装满为止。

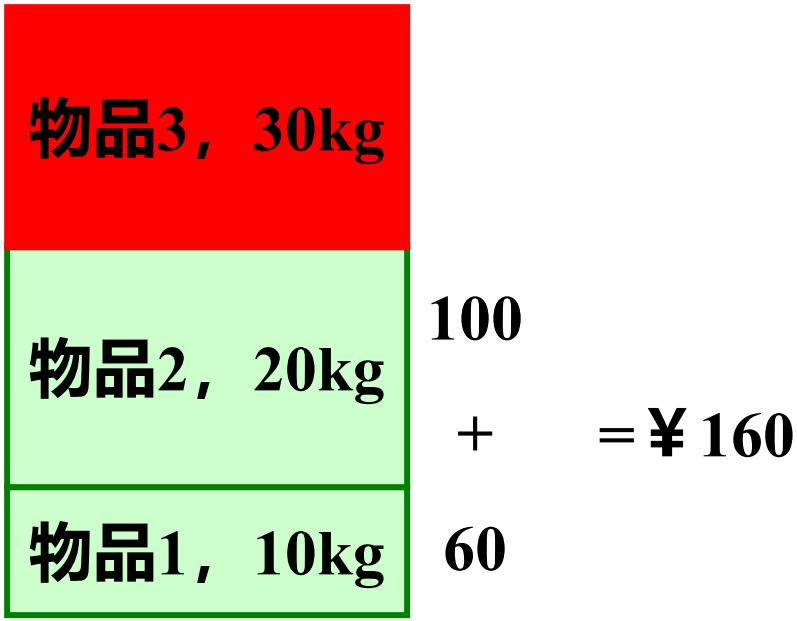
举例

- 有3种物品，背包的容量为50千克。物品1重10千克，价值60元；物品2重20千克，价值100元；物品3重30千克，价值120元。
- 用贪心算法求背包问题。

背包

有3种物品，背包的容量为50千克。物品1重10千克，价值60元；物品2重20千克，价值100元；物品3重30千克，价值120元。

- 贪心策略：物品1，6元/千克；物品2，5元/千克；物品3，4元/千克。



具体算法

```
void Knapsack(int n, float M, float v[],float w[], float x[])
{  sort(n,v,w);
   int i;
   for(i=1;i<=n;i++)  x[i]=0;
   float c=M;
   for(i=1;i<=n;i++)  {
         if(w[i]>c)  break;
         x[i]=1;
         c-=w[i];
        }
    if(i<=n)  x[i]=c/w[i];
}
```

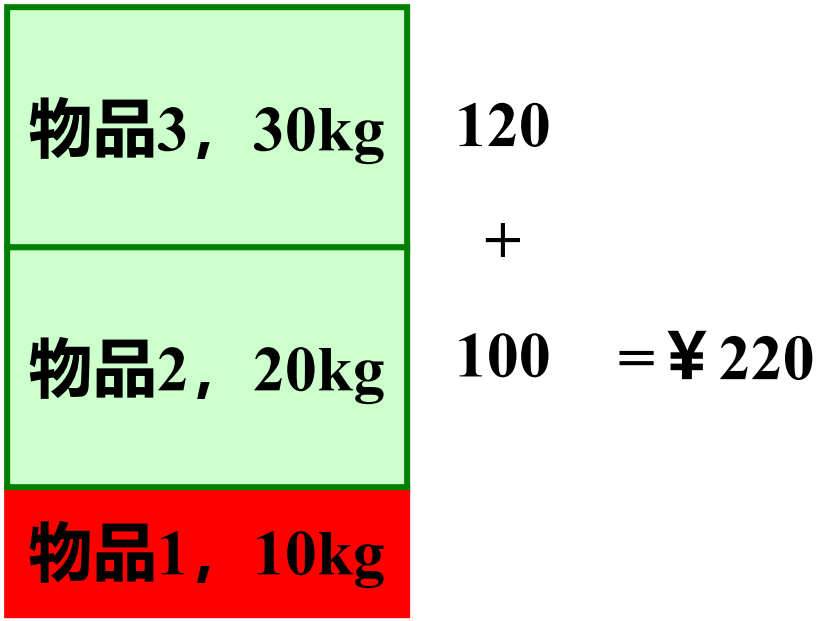
该算法前提：

所有物品在集合中按其单位重量的价值从小到大排列。

背包

有3种物品，背包的容量为50千克。物品1重10千克，价值60元；物品2重20千克，价值100元；物品3重30千克，价值120元。

- 贪心策略：物品1，6元/千克；物品2，5元/千克；物品3，4元/千克。



- 算法Knapspack的主要计算时间在于将各种物品按其单位重量的价值从小到大排序，算法的时间复杂度O(nlogn)。
- 对于0-1背包问题，贪心选择之所以不能得到最优解是因为在这种情况下，它无法保证最终能将背包装满，部分闲置的背包空间使每公斤背包空间的价值降低了。

部分背包问题

- 与0-1背包问题类似，所不同的是在选择物品i装入背包时，可以选择物品i的一部分，而不一定要全部装入背包。
- 此问题的形式化描述为，给定$c>0, w_i>0, v_i>0, 1\le i\le n$，要求找出一个n元0-1向量$(x_1, x_2, \ldots, x_n)$，其中$0\le x_i\le 1$，$1\le i\le n$，使得对$w_i x_i$求和小于等于c，并且对$v_i x_i$求和达到最大。

贪心算法能否达到最优?

例3 – 最优装载

- 问有一批集装箱要装上一艘载重量为c的轮船。其中集装箱i的重量为w_i。最优装载问题要求确定在装载体积不受限制的情况下，将尽可能多的集装箱装上轮船。
- [数学模型] 输入：$(x_1, x_2, \ldots, x_n)$, x_i=0表示集装箱i不装船；x_i=1表示集装箱i装船
  - 可行解：满足约束条件 $\sum_{i=1}^{n} w_i x_i \le c$的输入
  - 优化函数：$\sum_{i=1}^{n} x_i$

例3 – 最优装载

- **[算法思路]** 将装船过程划为多步选择，每步装一个货箱，每次从剩下的货箱中选择重量最轻的货箱. 如此下去直到所有货箱均装上船或船上不能再容纳其他任何一个货箱。
- [例]设n=8，$[w_1, \ldots, w_8]$=[100，200，50，90，150，50 ， 20 ， 80]，c=400.
  - 所考察货箱的次序为:7，3，6，8，4，1，5 ， 2。货箱7，3，6，8，4，1的总重量为390个单位且已被装载，剩下的装载能力为10,小于任意货箱. 所以得到解

    [1，0，1，1，0，1，1，1]

例3 – 最优装载

- 贪心选择性质
- 最优子结构性质

- 最优装载问题具有最优子结构性质。由最优装载问题的贪心选择性质和最优子结构性质容易证明算法loading的正确性。

- 复杂度：O(nlogn)

例子4 - 哈夫曼编码

- 1951年，哈夫曼和他在MIT信息论的同学需要选择是完成学期报告还是期末考试。导师Robert M. Fano给他们的学期报告的题目是，寻找最有效的二进制编码。由于无法证明哪个已有编码是最有效的，哈夫曼放弃对已有编码的研究，转向新的探索，最终发现了基于有序频率二叉树编码的想法，并很快证明了这个方法是最有效的。

例子4 - 哈夫曼编码

- **哈夫曼编码**是广泛地用于数据文件压缩的十分有效的编码方法。其**压缩率**通常在20%～90%之间。
- 哈夫曼编码算法是用字符在文件中出现的频率表来建立一个用0,1串表示各字符的**最优表示方式**。
- **编码目标：**给出现频率高的字符较短的编码，出现频率较低的字符以较长的编码，可以大大缩短总码长。

举例

- 一个数据文件包含100000个字符，要用压缩的方式来存储它。该文件中各字符出现的频率如表所示。

	a	b	c	d	e	f
频率(千次)	45	13	12	16	9	5
定长码	000	001	010	011	100	101
变长码	0	101	100	111	1101	1100

使用变长码要比使用定长码好得多。通过给出现频率高的字符较短的编码，出现频率较低的字符以较长的编码，可以大大缩短总码长。

1. 前缀码

- 定义：对每一个字符规定一个0,1串作为其代码，并要求任一字符的代码都不是其他字符代码的前缀。这种编码称为**前缀码**。
- 编码的前缀性质可以使译码方法非常简单。由于任一字符的代码都不是其他字符代码的前缀，从编码文件中不断取出代表某一字符的前缀码，转换为原字符，即可逐个译出文件中的所有字符。

举例

- 给定序列：001011101。

可唯一地分解为0,0,101,1101，因而其译码为aabe。

	a	b	c	d	e	f
频率(千次)	45	13	12	16	9	5
定长码	000	001	010	011	100	101
变长码	0	101	100	111	1101	1100