算法设计与分析

第4章 贪心算法
(2)

谢晓芹
哈尔滨工程大学计算机科学与技术学院

学习要点

- 通过应用范例学习贪心设计策略
  - 哈夫曼编码
  - 最优装载问题
  - 单源最短路径
  - 最小生成树

哈夫曼编码

- 文件压缩问题
  - 输入：给定符号表和一段数据
  - 输出：给每个符号赋一个编码，使给定数据在编码下的长度最短
- 二进制字符编码：每个字符用一个二进制0、1 串来表示
  - 固定长编码
    每个字符都用相同长的0、1 串表示
  - 可变长编码
    经常出现的字符用短码，不经常出现的用长码
- 哈夫曼编码
  - 用于数据文件压缩的十分有效的编码方法。压缩率通常在20%～90%之间
  - 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

哈夫曼编码

	a	b	c	d	e	f
频率(千次)	45	13	12	16	9	5
8位ASCII码	01100001	01100010	01100011	01100100	01100101	01100110
定长码	000	001	010	011	100	101
变长码	0	101	100	111	1101	1100

- 定长码
  - $45\times3+13\times3+12\times3+16\times3+9\times3+5\times3=300$
- 变长码
  - $45\times1+13\times3+12\times3+16\times3+9\times4+5\times4=224$

减少约25%

- 频率越高，其编码越短，频率越低，编码越长

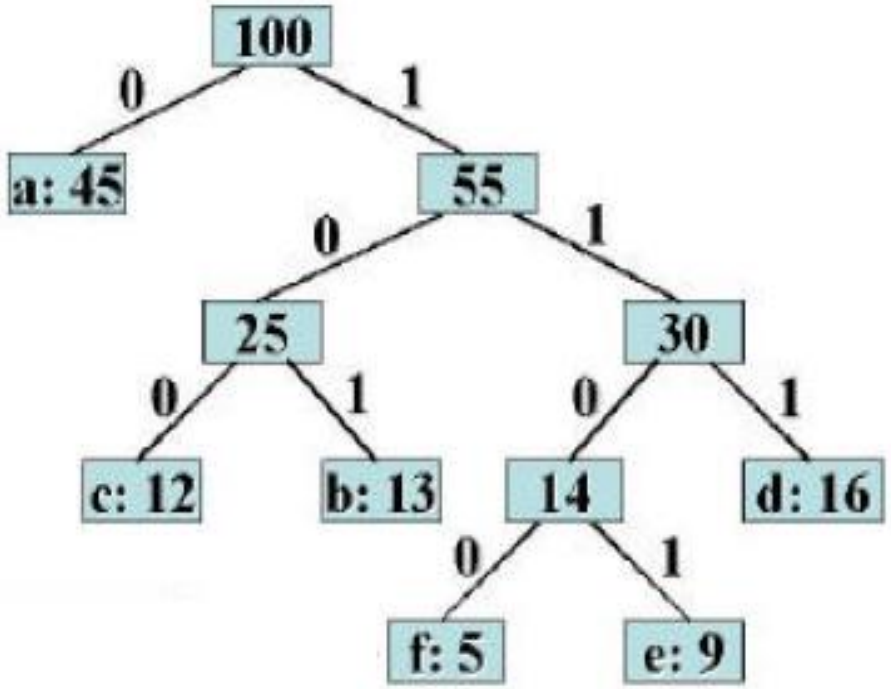
哈夫曼编码

- 可变长码的问题？
  - 译码歧义：编码后的消息可能有多种译码方式
- 前缀码
  - 对每一个字符规定一个0,1串作为其编码，并且任一字符的编码都不是其它字符的编码的前缀。这种编码称为具有前缀性质（或者前缀码）
- 例如
  - 001011101分解为0,0,101,1101,　　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

哈夫曼编码

- 通常用二叉树来作为前缀编码的数据结构
- 编码树
  - 叶结点：用字符及其出现频率标记。
  - 内结点(包括根结点)：其子树的叶子的频率和。
  - 边标记：
    一个结点的左侧边标记0，右侧边标记1。
- 前缀码
  - 从树根到树叶的一条路径
- 这种树唯一吗？ 怎么构建？



哈夫曼编码

- C:编码字符集，C中任一字符以频率f(c)在数据文件中出现.前缀编码方案对应于二叉树T, 字符c在T中的深度为$d_T(c)$
- 最优编码树问题
  - 输入：字母表C={c_1, c_2,…., c_n}, 频率表F={f(c_1),f(c_2),…,f(c_n)}
  - 输出：C的具有最小平均码长B(T)的前缀编码树,即哈夫曼树
- 分析
  - (编码方案)平均码长定义为： $B(T)=\sum_{c\in C} f(c)d_T(c)$
  - 最优前缀码
    - 使**平均码长达到最小**的前缀码编码方案称为给定编码字符集C的最优前缀码
  - 如何实现： 哈夫曼树
    - 表示最优前缀码的二叉树
    - 总是一棵完全二叉树，即树中任一结点都有2个儿子结点

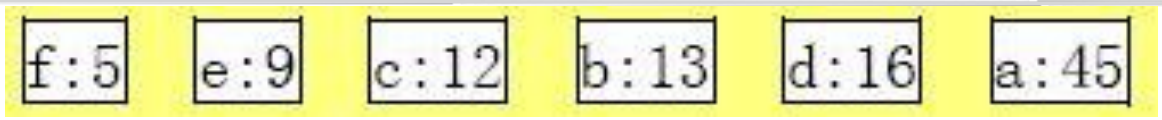
哈夫曼编码

- 哈夫曼编码
  - 哈夫曼提出构造最优前缀码的贪心算法，产生的编码方案称为哈夫曼编码
  - 基本思想
    - 将字符按照出现频率从高到低排序
    - 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，以大大缩短总码长

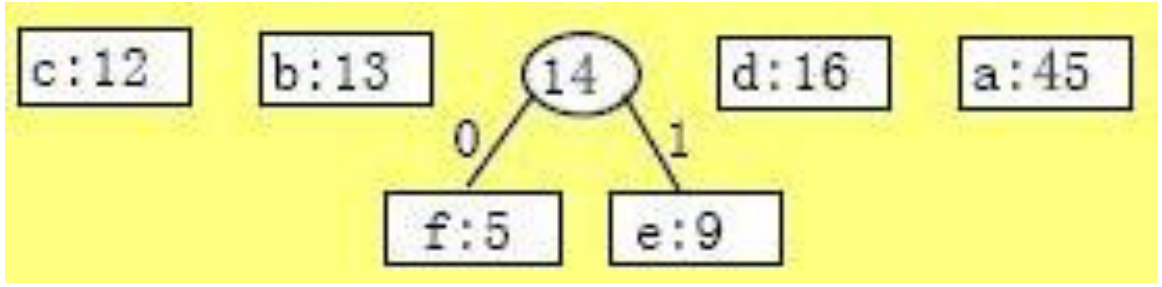
- 频率最高？　排序？
- 最短编码？　哈夫曼树

哈夫曼编码

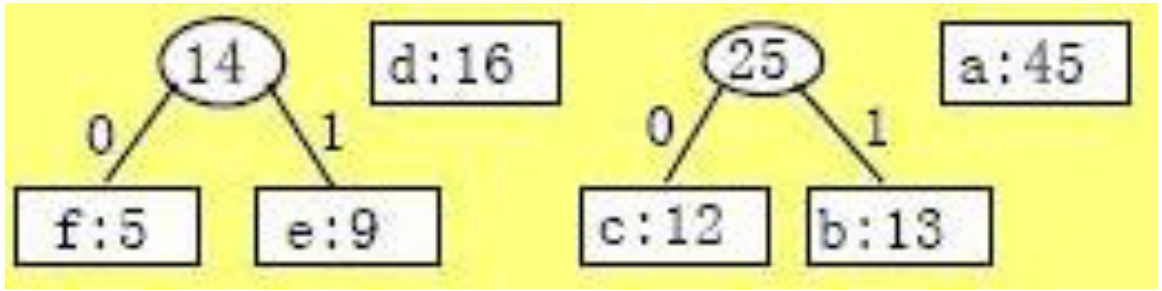
初始：



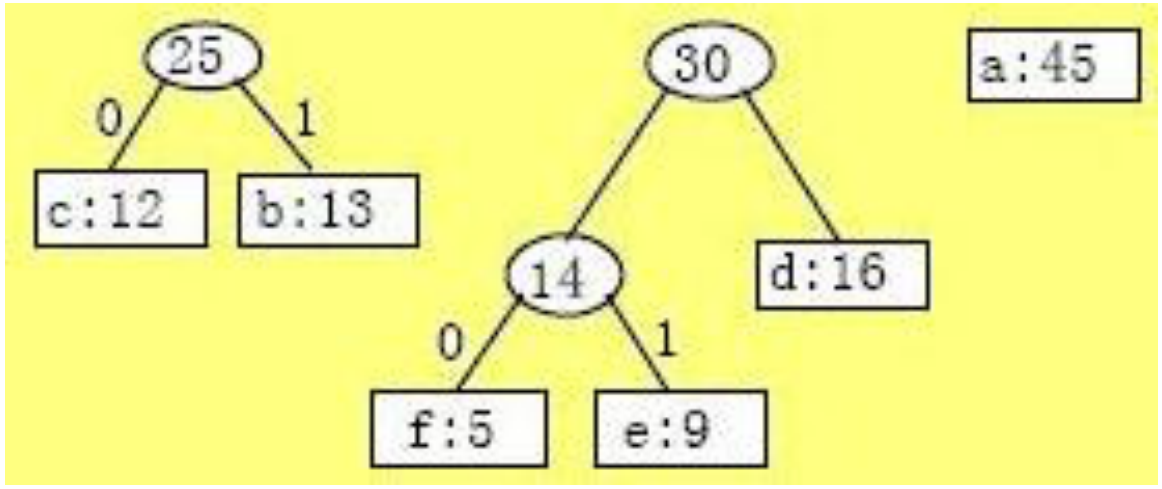
第1步：



第2步：



第3步：

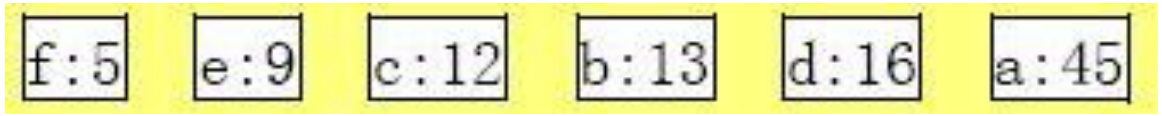


贪什么？

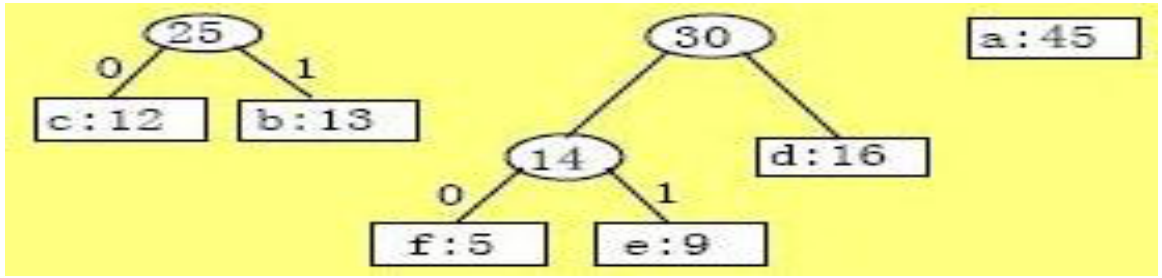
怎么贪？

哈夫曼编码

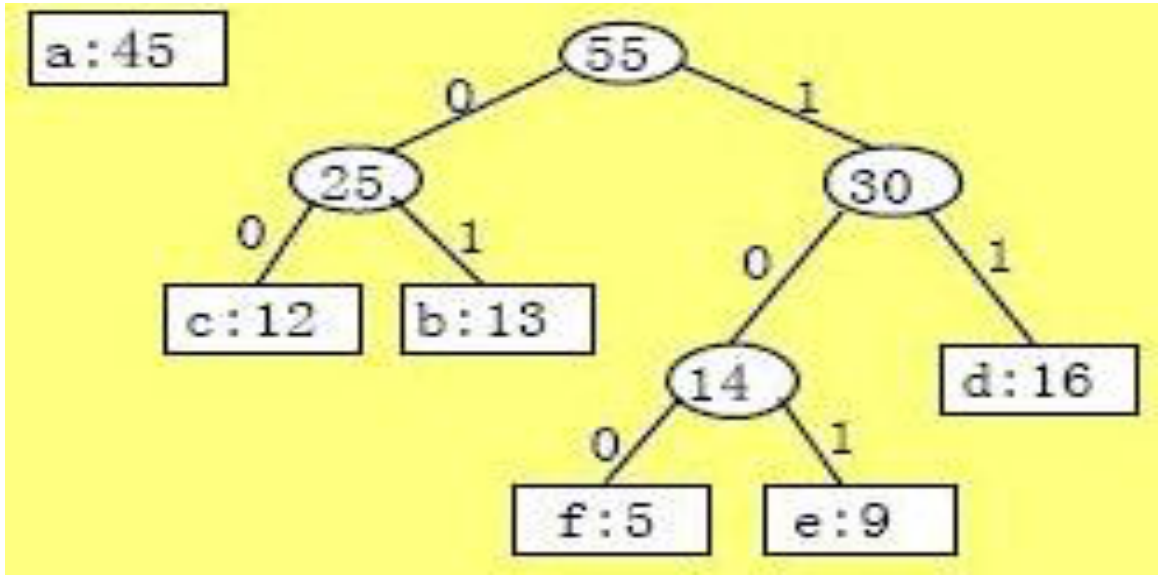
初始：



第3步：



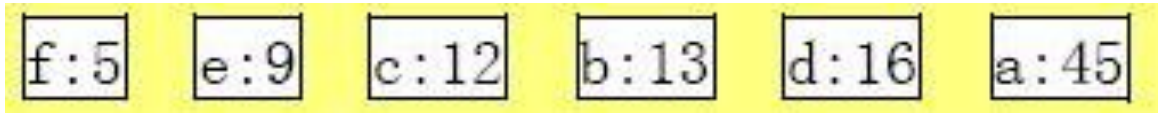
第4步：



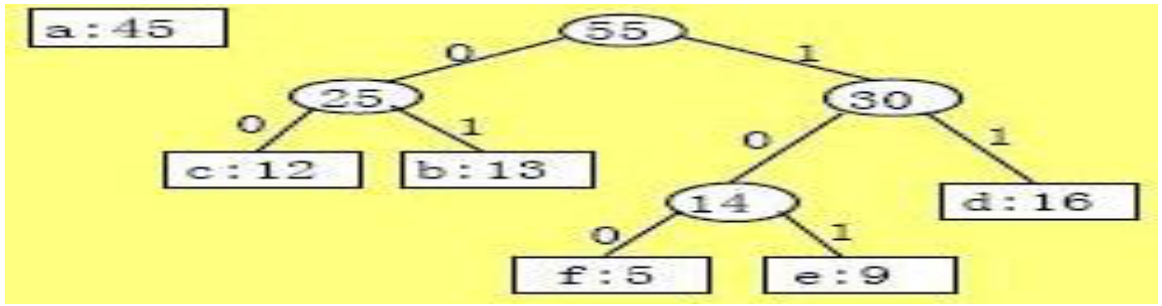
何时结束？

哈夫曼编码

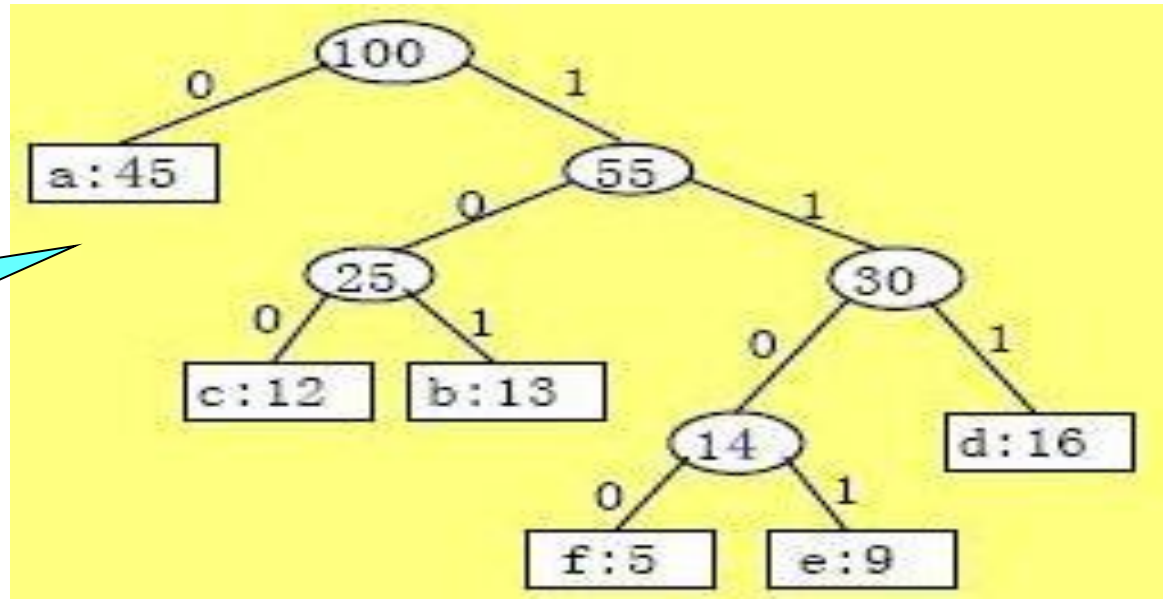
初始：



第4步：



第5步：



选取最小频率子树

编码

哈夫曼编码

- 哈夫曼算法以自底向上的方式构造表示最优前缀码的二叉树T
  - 以|C|个叶结点开始, 执行|C| - 1次的“合并”运算后产生最终所要求的树T
- 过程
  - 贪什么： 频率最小的两个字符
  - 怎么贪
    - **合并**为一棵树
    - 根：频率和， 叶子：两个字符
  - 何时终止： n-1个结点全部生成
  - 编码
    - 左边为0，右边为1
    - 从根到叶结点上所有赋值的连接

哈夫曼树

哈夫曼编码

什么数据结构,方便插入和删除操作?

算法设计技术: 利用堆来管理算法执行中的信息

- 堆: n个元素的序列{$k_1,k_2,...,k_n$},当且仅当满足以下关系时,称之为堆.

$$\begin{cases} k_i \le k_{2i} \\ k_i \le k_{2i+1} \end{cases} \quad \begin{cases} k_i \ge k_{2i} \\ k_i \ge k_{2i+1} \end{cases} \quad (i = 1,2,...,\lfloor n/2 \rfloor)$$

- 含义:
  - 完全二叉树中所有非终端结点的值不大于(或不小于)其左右孩子结点的值.
  - 堆顶元素必为序列中n个元素的最小值(或最大值)

哈夫曼编码

n个字符的集合　　频度$f(c_1),f(c_2),\ldots,f(c_n)$

Huffman(C，F)　　(使用堆操作实现)

```
n ← |C|;
Q ← C;     /* 用BUILD-HEAP 建立堆 */                          O(n)
for i ← 1 to n-1{      //终止条件：n-1个节点全部生成            n-1
   z ← Allocate-Node( );
   x ← left[z] ← Extract-MIN(Q);          /* 堆操作*/  O(logn)
   y ← right[z] ← Extract-MIN(Q);         /* 堆操作*/  O(logn)
   f(z) ← f(x)+f(y);
   insert(Q,z); }                         /* 堆操作*/
Return Extract-MIN(Q).
```

哈夫曼编码

- 哈夫曼算法的计算时间为O(nlogn)
- 分析
  - 初始化优先队列需要O(n)计算时间
  - 最小堆的ExtractMin和Insert运算均需O(logn)时间
  - n-1次的合并总共需要O(nlogn)计算时间。

```
Huffman(C，F) {
    n ← |C|;
    Q ← C;        //用BUILD-HEAP 建立堆
    for i ← 1 to n-1{      //终止条件
        z ← Allocate-Node( );
        x ← left[z] ← Extract-MIN(Q); / //堆操作
        y ← right[z] ← Extract-MIN(Q); //堆操作
        f(z) ← f(x)+f(y);
        insert(Q,z); }                    ///堆操作
    Return Extract-MIN(Q)}
```

哈夫曼编码

- 哈夫曼算法的正确性
  - 证明哈夫曼算法的正确性，只要证明最优前缀码问题具有如下两个性质：
    (1)贪心选择性质
    (2)最优子结构性质。

哈夫曼编码 - 贪心性质证明

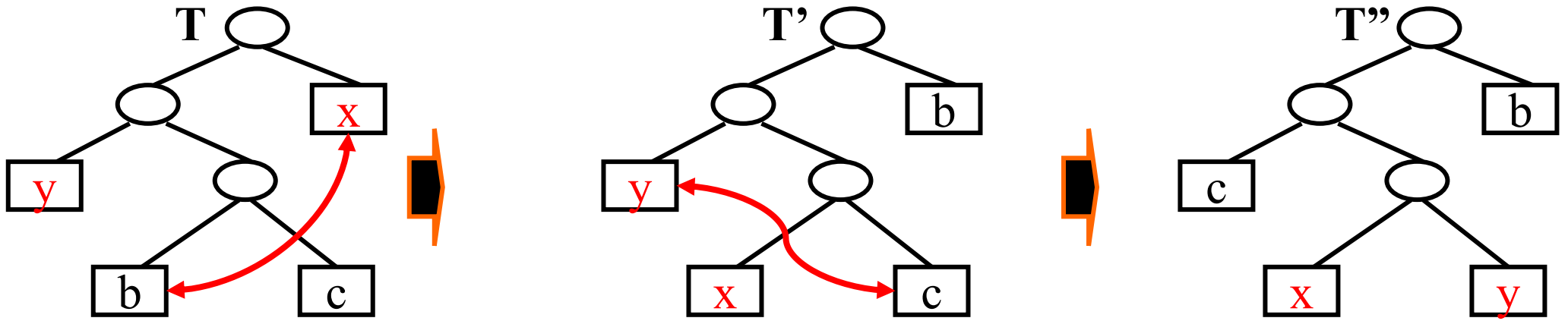
- 引理1(贪心选择性). 设C是字母表，c∈C，c具有频率f(c)，x、y 是C 中具有最少频率的两个字符，则存在一个C 的最优前缀树，x 与y 的编码具有相同长度，且仅在最末一位不同
- 分析
  - 贪心选择 是贪频率最小的两个字符→ 树上是兄弟结点，左右不同而已
  - 哈夫曼编码得到一个最优解，是以贪心选择开始

哈夫曼编码 - 贪心性质证明

- 引理1(贪心选择性). 设C是字母表，c∈C，c具有频率f(c)，x、y 是C 中具有最少频率的两个字符，则存在一个C 的最优前缀树，x 与y 的编码具有相同长度，且仅在最末一位不同
- 证明： 设T 是一个C 的最优前缀树，且b 和c 是具有最大深度且为兄弟的两个字符
  不失一般性，设f(b)≤f(c),f(x) ≤ f(y).

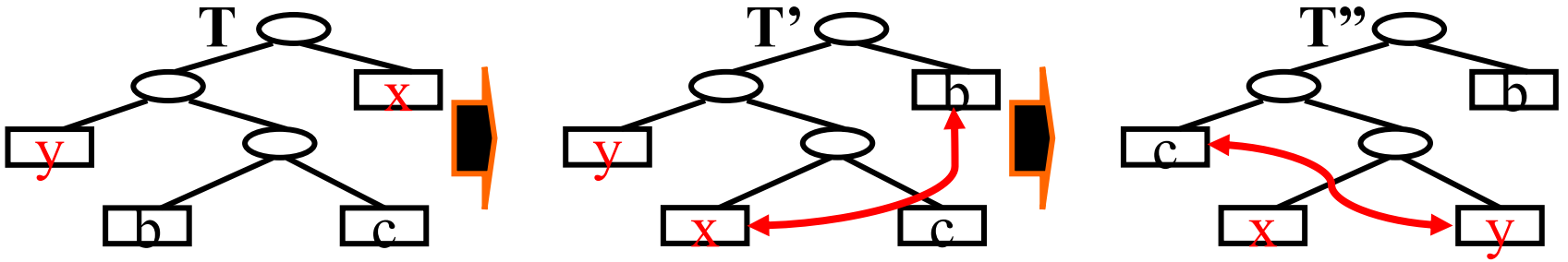
  因x 与y 是具有最低频率的字符, f(b) ≥f(x),f(c) ≥f(y)

从T 构造T′，交换T 的b 和x；　　从T′构造T″，交换T′ 的y 和c；



哈夫曼编码 - 贪心性质证明

T''是最优前缀树.



- **证:** $B(T)-B(T')=\sum_{c\in C}f(c)d_T(c)-\sum_{c\in C}f(c)d_{T'}(c)$

$= f(x)d_T(x)+f(b)d_T(b)-f(x)d_{T'}(x)- f(b)d_{T'}(b)$

$= f(x)d_T(x)+f(b)d_T(b)-f(x)d_T(b)- f(b)d_T(x)$

$= (f(b)-f(x))(d_T(b)-d_T(x))$

$\because f(b) \geq f(x), d_T(b) \geq d_T(x)$ (因为b 的深度最大)

$\therefore B(T)-B(T') \geq 0,\ B(T) \geq B(T')$

同理可证$B(T') \geq B(T'')$. 于是$B(T) \geq B(T'')$.

由于T 是最优化的，所以$B(T) \leq B(T'')$.

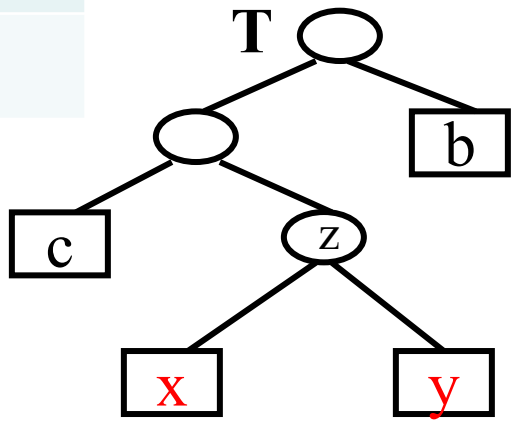
于是,$B(T)=B(T'')$，所以T''是C 的最优化前缀编码树.

在T''中, x 和y 具有相同长度编码, 而且仅最后一位不同。

哈夫曼编码 - 最优子结构性质证明

- 引理2(最优子结构).设T 是字母表C 的最优前缀树，c∈C，f(c)是c在文件中出现的频率。设x、y 是T中任意两个相邻叶结点，z 是它们的父结点，则z看作为频率为f(z)=f(x)+f(y)的字符，T′ =T-{x,y}表示了字母表C′ =C-{x,y}∪{z}的最优前缀编码树
- 分析

		最优解
原问题	C	T
子问题	$C' = \{C - \{x, y\}\} \cup \{z\}$	$T' = T - \{x, y\}$



哈夫曼编码 - 最优子结构性质证明

- 引理2(最优子结构).设T 是字母表C 的最优前缀树，c∈C，f(c)是c在文件中出现的频率。设x、y 是T中任意两个相邻叶结点，z 是它们的父结点，则z看作为频率为f(z)=f(x)+f(y)的字符，T′ =T-{x,y}表示了字母表C′ =C-{x,y}∪{z}的最优前缀编码树。

➢ **证: 先证** B(T)=B(T′)+f(x)+f(y)

∀v∈C-{x,y}, $d_T(v)=d_{T'}(v)$。于是$f(v)d_T(v)=f(v)d_{T'}(v)$。

$$B(T)=\sum_{v\in C-\{x,y\}} f(v)d_T(v) + f(x)d_T(x) + f(y)d_T(y)$$

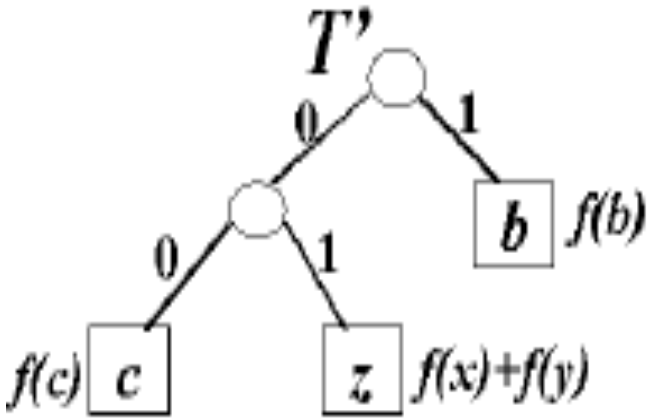
$$= \sum_{v\in C-\{x,y\}} f(v)d_T(v) + \big(f(x)+f(y)\big)(d_{T'}(z)+1)$$

$$B(T')= \sum_{v\in C-\{x,y\}} f(v)d_{T'}(v) + f(z)d_{T'}(z)$$

由于f(x)+f(y)=f(z), $f(x)d_T(x)+f(y)d_T(y)=f(z)d_{T'}(z)+(f(x)+f(y))$.

于是 B(T)=B(T')+f(x)+f(y)

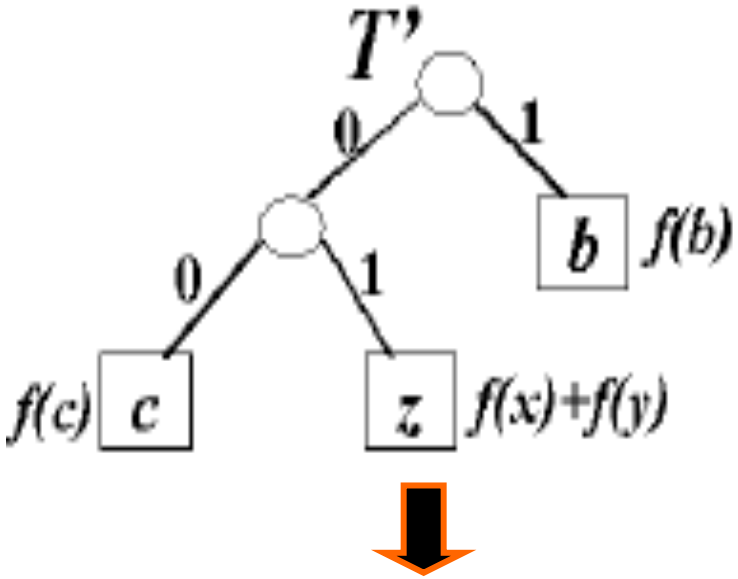
由于$d_T(x)=d_T(y)=d_{T'}(z)+1$，
$f(x)d_T(x)+f(y)d_T(y)=(f(x)+f(y))(d_{T'}(z)+1)$
$=(f(x)+f(y))d_{T'}(z)+(f(x)+f(y))$



反证法：若T'是C'的非最优前缀编码树，则必存在T''，使B(T'')<B(T').

哈夫曼编码 - 最优子结构性质证明

反证法：若T’是C’的非最优前缀编码树，则必存在T’’，使B(T’’)<B(T’).

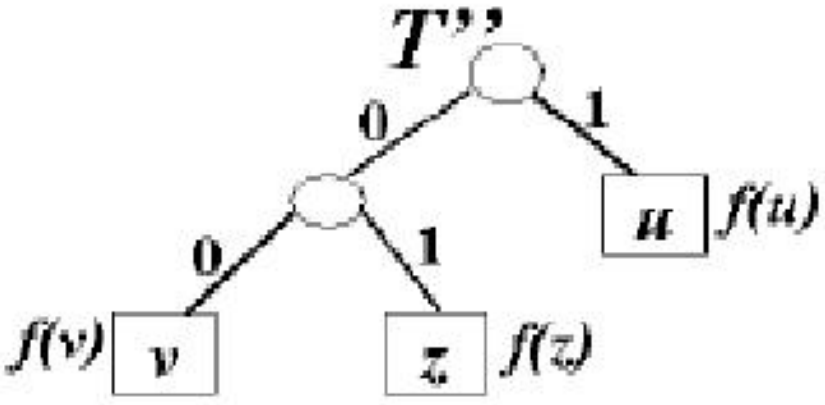


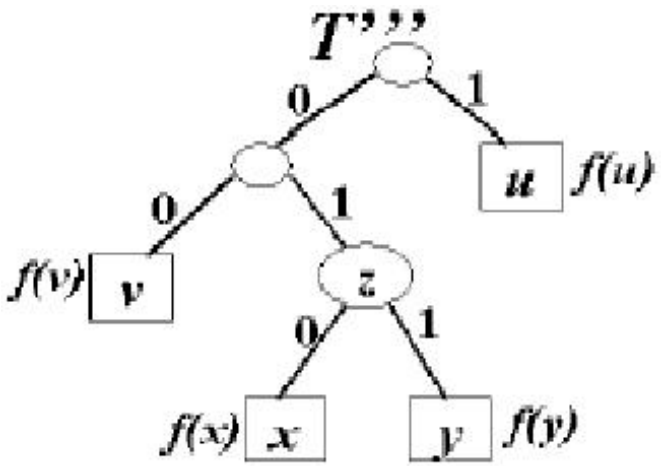
证: (1)因为z 在C’中是一个字符，它必是T’’中的叶子。把结点 x 与y 加入T’’，作为z 的子结点，则得到C 的一个如下的前缀编码树T’’’

(2) T’’’代价为：

$$B(T''') = \ldots\ldots + (f(x)+f(y))(d_{T'''}(z)+1)$$
$$= \ldots\ldots + f(z)d_{T'''}(z)+(f(x)+f(y))$$
$$= B(T'')+f(x)+f(y)$$
$$< B(T')+f(x)+f(y) = B(T)$$

(3) 与T是最优的相矛盾，故T’是最优的





哈夫曼编码

- 定理. Huffman 算法产生一个最优前缀编码树。
- 证:
  由于引理1、引理2 成立,而且Huffman 算法按照引理1 的贪心选择性确定的规则进行局部优化选择，所以Huffman 算法产生一个最优前缀编码树。

哈夫曼编码

- 实例
  - 1.周杰伦(Z) ，f(Z)=20
  - 2.刘德华(L) ，f(L)=7
  - 3.韩愈(H)， f(H)=10
  - 4.李小龙(D)，f(D)=4
  - 5.易中天(Y)，f(Y)=18
- 构造哈夫曼编码树，并计算编码树的平均码长？

使用贪心策略的典型步骤

- 步骤1 将原问题表述为一个做出一个选择后剩下唯一的一个子问题的形式
- 步骤2 证明在所有的最优选择里面总有一个是贪心选择
- 步骤3 证明贪心选择加上对剩下子问题的最优解就是原问题的最优解

分治、动态规划和贪心的比较

	标准分治	动态规划	贪心
适用类型	通用问题	优化问题	优化问题
子问题结构	每个子问题不同	很多子问题重复	只有一个子问题
最优子结构	不要求	必须满足	必须满足
解决子问题数	全部子问题都要解决	全部子问题都要解决	只要解决一个子问题
子问题依赖性	先选择后解决子问题	先解决子问题后选择	先选择后解决子问题
求解顺序	递归与递推都可	从底往上	从顶往下

小结

- 哈夫曼编码
  - 问题描述
  - 求解算法

- 重点
  - 贪心算法的证明过程
  - 方法
    - 构造法，反证法，反证+构造法，数学归纳法
- 难点
  - 怎么表述贪心选择性质
  - 怎么表述最优子结构性质

实践创新能力训练

- 一辆汽车加满油后可以行使n公里。旅途中有若干加油站。请设计一个有效算法，指出应在哪些加油站停靠加油，使得沿途加油次数最少，并证明算法能产生一个最优解。
- 提示：对于给定的n和k个加油站位置，计算最少加油次数。
- 输入数据：n，k，d[k]表示第k个加油站到第k-1个加油站之间的距离。第0个加油站表示出发地，汽车已加满油，第k+1个加油站表示目的地。
- 输出: (x1,x2,...,xk)， xi=1表示停留，xi=0表示不停

  加油策略 strat[ ](strat[i]=j表示第i次加油是在第j个加油站)