《算法设计与分析》

第四章 贪心方法

马丙鹏

2023年10月16日

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

第四章 贪心方法

- 4.1 一般方法
- 4.2 背包问题
- 4.3 带有限期的作业排序
- 4.4 最优归并模式
- 4.5 最小生成树
- 4.6 单源点最短路径

4.4 最优归并模式

■ 1. 问题的描述

- 两个文件的归并问题
  - 两个已知文件的一次归并所需的计算时间＝O(两个文件的元素总数)
  - 例：

n个记录的文件

＋

m个记录的文件

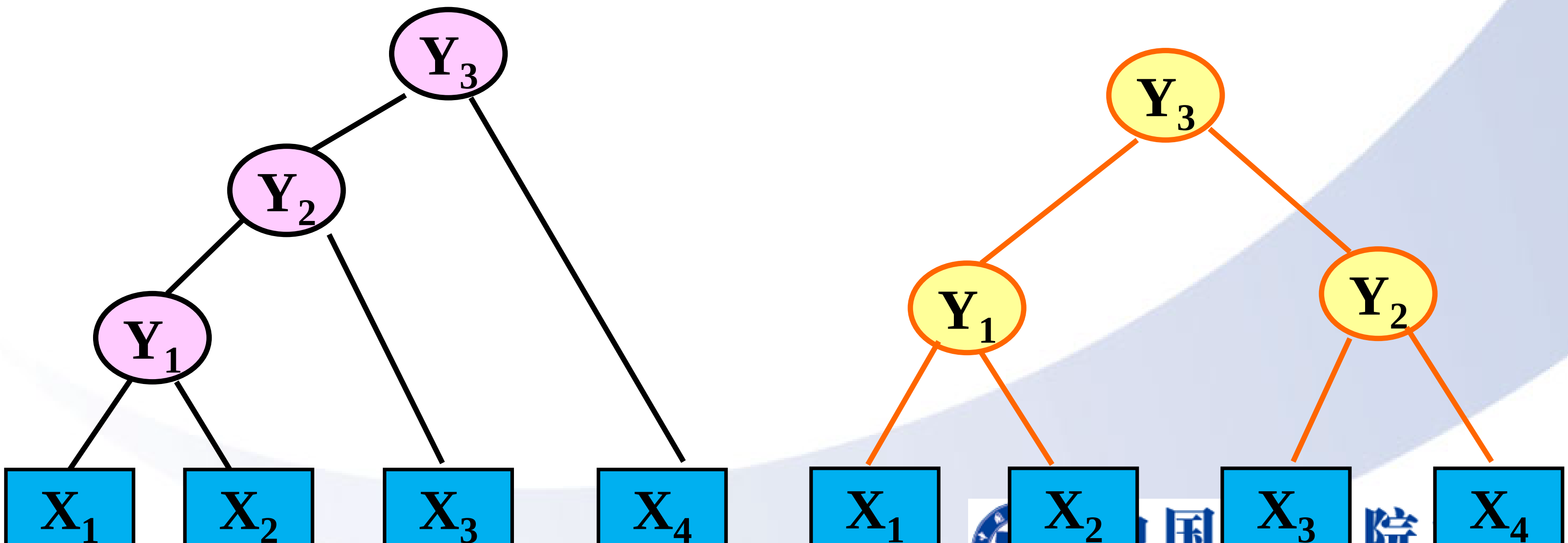
→ (n+m) 个记录的文件

O(n+m)

4.4 最优归并模式

■ 1. 问题的描述

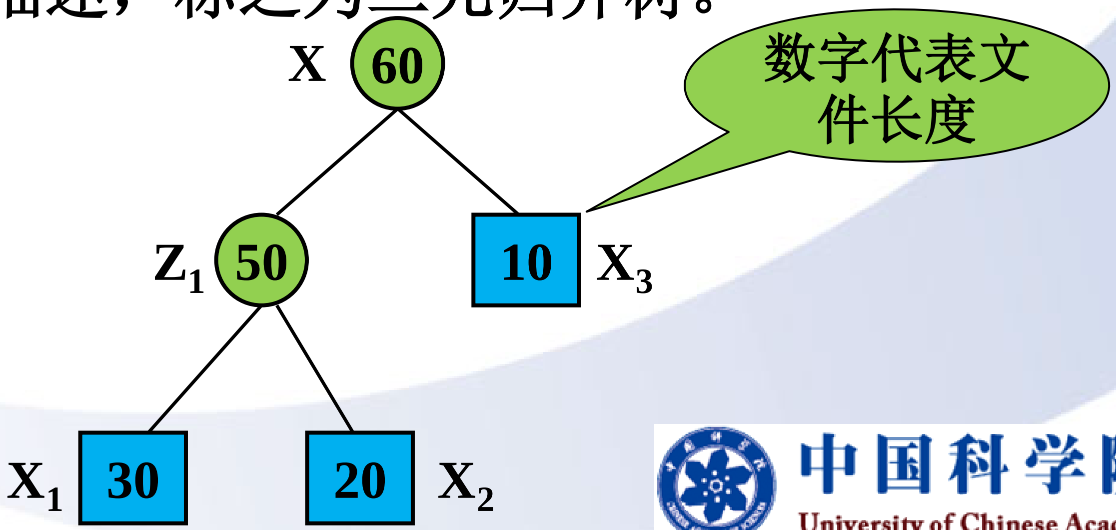
- □多个文件的归并
  - ➢已知n个文件，将之归并成一个单一的文件
  - ➢例：假定文件X_1, X_2, X_3, X_4，采用两两归并的方式，可能的归并模式有：



4.4 最优归并模式

■ 1. 问题的描述

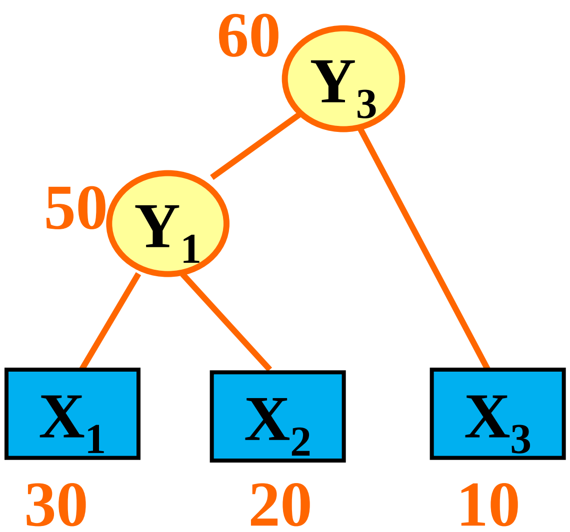
- □二路归并模式：
  - ➢每次作两个文件的归并；当有多个文件时，采用两两归并的模式，最终得到一个完整的记录文件。
- □二元归并树：
  - ➢二路归并模式的归并过程可以用一个二元树的形式描述，称之为二元归并树。



4.4 最优归并模式

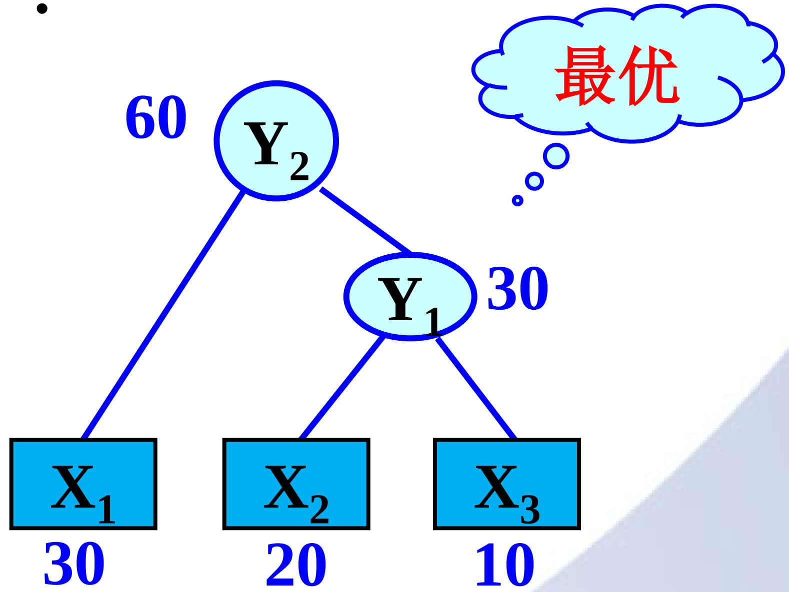
- **1. 问题的描述**
  - 归并树的构造
    - 外结点：n个原始文件，
    - 内结点：一次归并后得到的文件，
    - 在两路归并模式下，每个内结点刚好有两个儿子，代表把它的两个儿子表示的文件归并成其本身所代表的文件，
    - 不同的归并顺序带来的计算时间是不同的。

例4.5 已知X_1, X_2, X_3是分别为30、20、10个记录长度的已分类文件。将这3个文件归并成长度为60的文件。可能的归并过程和相应的记录移动次数如下：



记录移动的总次数:

50+60=110



记录移动的总次数:

30+60=90

问题：采用怎样的归并顺序才能使归并过程中元素的移动次数最小(或执行的速度最快)?

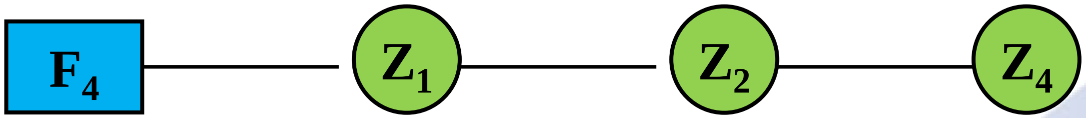
4.4 最优归并模式

■ 2. 贪心求解

□ 目标函数

- ➢ 目标：元素移动的次数最少
- ➢ 实例：为得到归并树根结点表示的归并文件，外部结点中每个文件记录需要移动的次数 = 该外部结点到根的距离，即根到该外部结点路径的长度。如，

F_4 :



则F_4中所有记录在整个归并过程中移动的总量 = $|F_4|*3$

4.4 最优归并模式

■ 2. 贪心求解

□ 目标函数

➢ 带权外部路径长度：

✓ 记d_i是由根到代表文件F_i的外部结点的距离，

✓ q_i是F_i的长度，

✓ 则这棵树代表的归并过程的元素移动总量是：

$$\sum_{1 \le i \le n} q_i d_i$$

➢ 最优的二路归并模式：

✓ 与一棵具有最小外部带权路径长度的二元树相对应。

4.4 最优归并模式

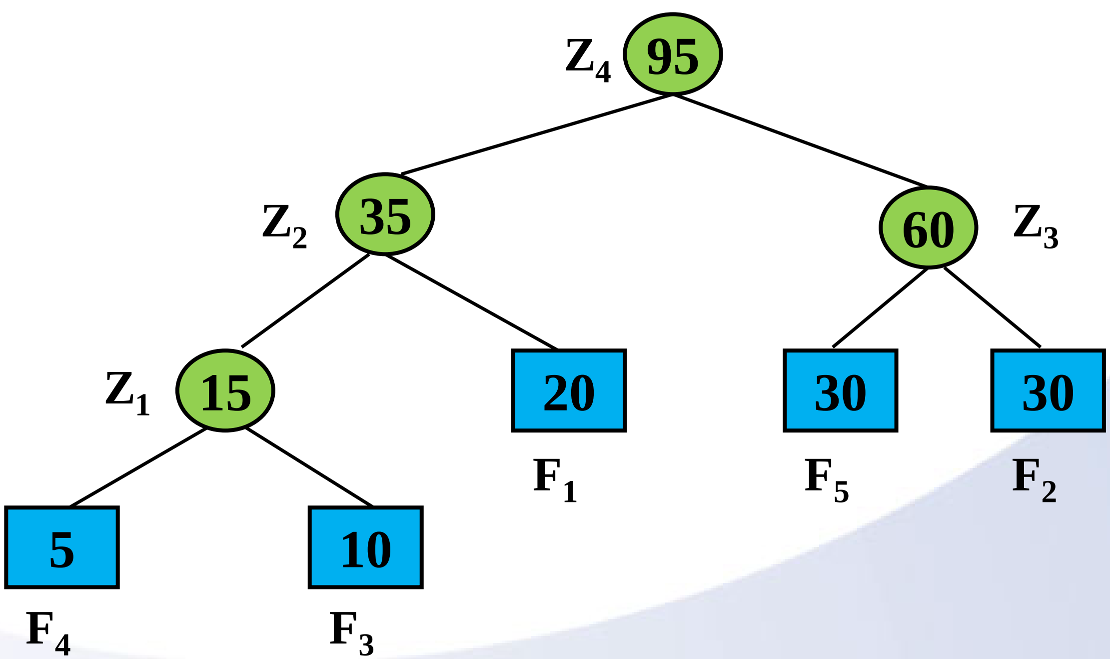
- **2. 贪心求解**
  - 度量标准的选择
    - 任意两个文件的归并所需的元素移动次数与这两个文件的长度之和成正比；
    - 度量标准：
      - ✓每次选择需要移动次数最少的两个集合进行归并；
    - 处理规则：
      - ✓每次选择长度最小的两个文件进行归并。

4.4 最优归并模式

■ 2. 贪心求解

□度量标准的选择

➢例， $(F_1, F_2, F_3, F_4, F_5) = (20, 30, 10, 5, 30)$



4.4 最优归并模式

■ 3.生成二元归并树的算法

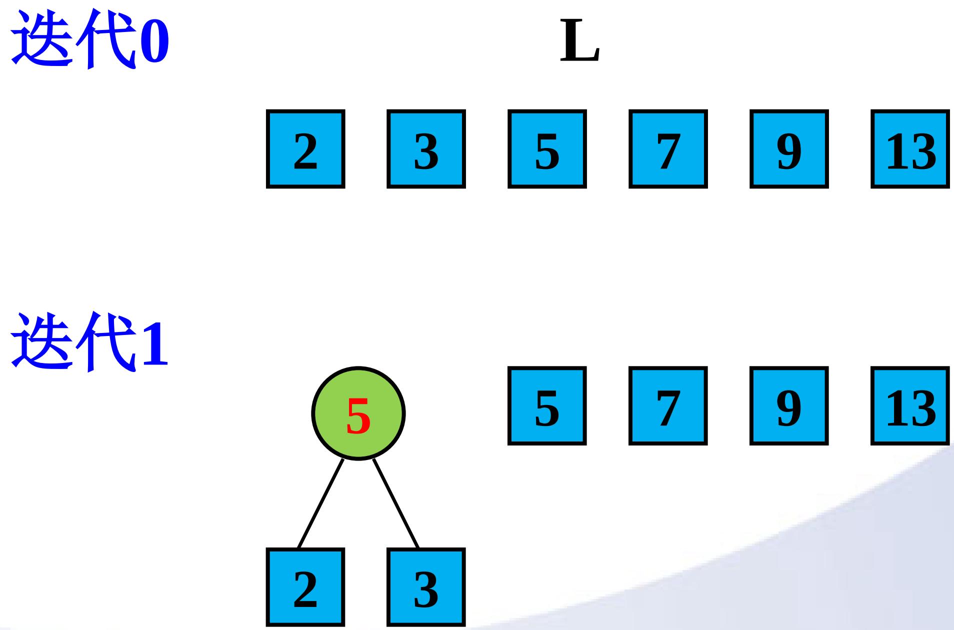
算法4.6 生成二元归并树的算法

```
procedure TREE(L, n)
//L是n个单结点的二元树表//
for i←1 to n-1 do
    call GETNODE(T)          //构造一颗新树T//
    LCHILD(T) ←LEAST(L)  //从表L中选当前根WEIGHT最小的树，
                                     并从中删除//
    RCHILD(T) ←LEAST(L)
    WEIGHT(T) ←WEIGHT(LCHILD(T)) + WEIGHT(RCHILD(T))
    call INSERT(L, T)          //将归并的树T加入到表L中//
repeat
return (LEAST(L))     //此时，L中的树即为归并的结果//
end TREE
```

4.4 最优归并模式

■ 3.生成二元归并树的算法

□例已知六个初始文件，长度分别为：2, 3, 5, 7, 9, 13。采用算法TREE，各阶段的工作状态如图所示：

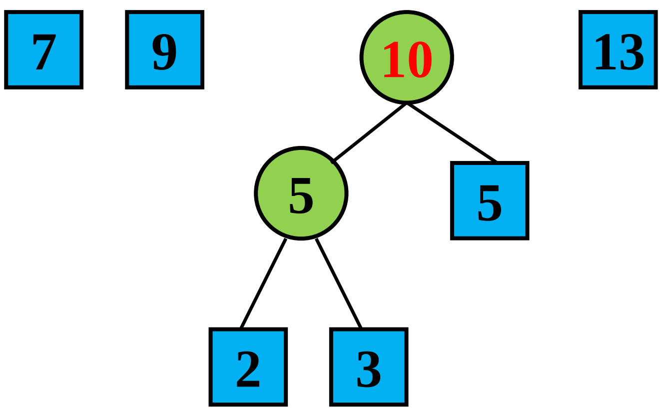


4.4 最优归并模式

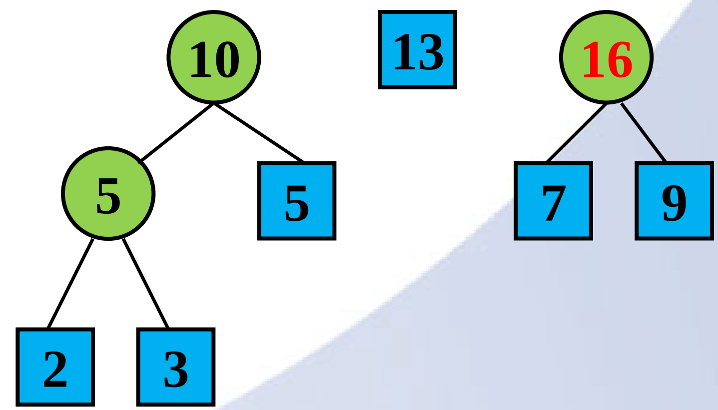
■ 3.生成二元归并树的算法

□例已知六个初始文件，长度分别为：2, 3, 5, 7, 9, 13。采用算法TREE，各阶段的工作状态如图所示：

迭代2



迭代3

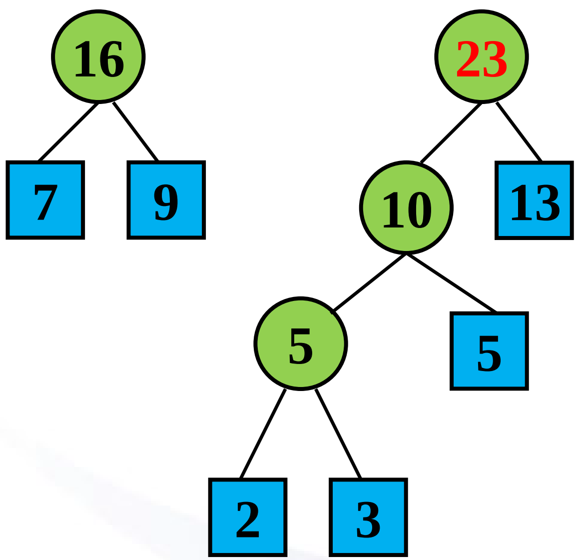


4.4 最优归并模式

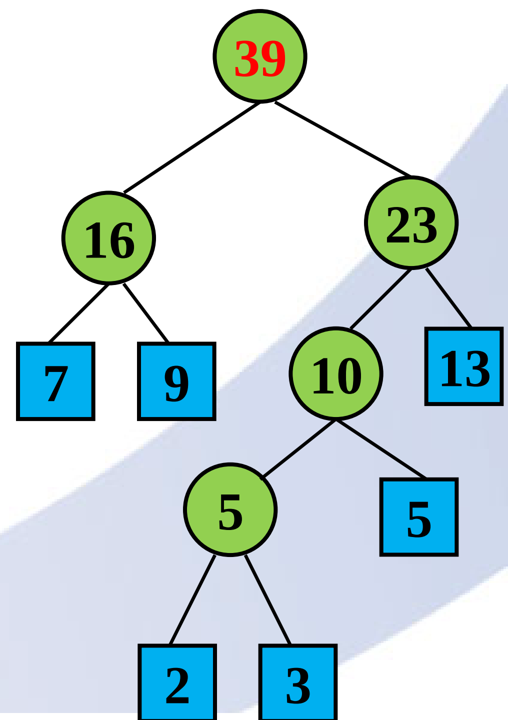
■ 3.生成二元归并树的算法

□例已知六个初始文件，长度分别为：2, 3, 5, 7, 9, 13。采用算法TREE，各阶段的工作状态如图所示：

迭代4



迭代5



4.4 最优归并模式

- 3.生成二元归并树的算法
  - 时间分析
    - ① 循环体：n-1次
    - ② L以有序序列表示
      - LEAST(L): O(1)
      - INSERT(L, T): O(n)
      - 总时间: $O(n^2)$
    - ③ L以min-堆表示
      - LEAST(L): O(logn)
      - INSERT(L, T): O(logn)
      - 总时间: O(nlogn)

4.4 最优归并模式

■ 4. 最优解的证明

□定理4.4 若L最初包含$n \geq 1$个单结点的树，这些树有WEIGHT值为$(q_1, q_2, \ldots, q_n)$，则算法TREE对于具有这些长度的n个文件生成一棵最优的二元归并树。

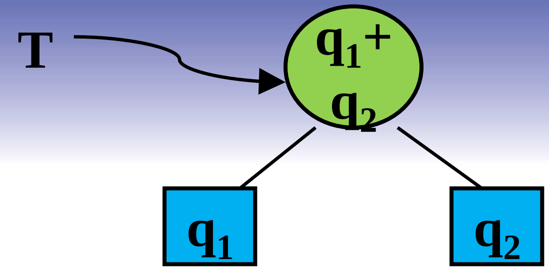
证明：归纳法证明

① 当$n=1$时，返回一棵没有内部结点的树。定理得证。

② 假定算法对所有的$(q_1, q_2, \ldots, q_m)$, $1 \leq m < n$, 生成一棵最优二元归并树。

③ 对于n，假定$q_1 \leq q_2 \leq \ldots \leq q_n$，则$q_1$和$q_2$将是在for循环的第一次迭代中首先选出的具有最小WEIGHT值的两棵树(的WEIGHT值)；如图所示，T是由这样的两棵树构成的子树：

4.4 最优归并模式



■ 4. 最优解的证明

- 设T’是一棵对于$(q_1, q_2, \ldots, q_n)$的最优二元归并树。
- 设P是T’中距离根最远的一个内部结点。

若P的两棵子树不是q_1和q_2，则用q_1和q_2代换P当前的子树而不会增加T’的带权外部路径长度。(?)

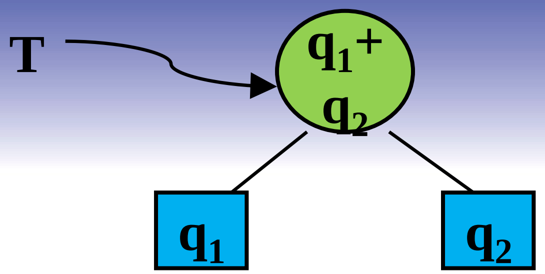
故，T应是最优归并树中的子树。

则在T’中用一个权值为q_1+q_2的外部结点代换T，得到的是一棵关于$(q_1+q_2, \ldots, q_n)$最优归并树T”。

而由归纳假设，在用权值为q_1+q_2的外部结点代换了T之后，过程TREE将针对$(q_1+q_2, \ldots, q_n)$得到一棵最优归并树。将T带入该树，根据以上讨论，将得到关于$(q_1, q_2, \ldots, q_n)$的最优归并树。

故，TREE生成一棵关于$(q_1, q_2, \ldots, q_n)$的最优归并树。

4.4 最优归并模式



■ 4. 最优解的证明

- 若P的两棵子树不是q_1和q_2，则不妨假设为q_3和q_4，则q_3和q_4大于等于q_1和q_2
- 设q_3和q_4到根结点的距离为d_1
- 设q_1和q_2在树中某个的位置，其到根结点的距离为d_2和d_3，则$d_1 \ge d_2$，$d_1 \ge d_3$
- 将q_1和q_3互换，q_2和q_4互换，互换后带权路径长度变化为：

$$q_3d_1 + q_4d_1 + q_1d_2 + q_2d_3 - q_1d_1 - q_2d_1 - q_3d_2 - q_4d_3$$

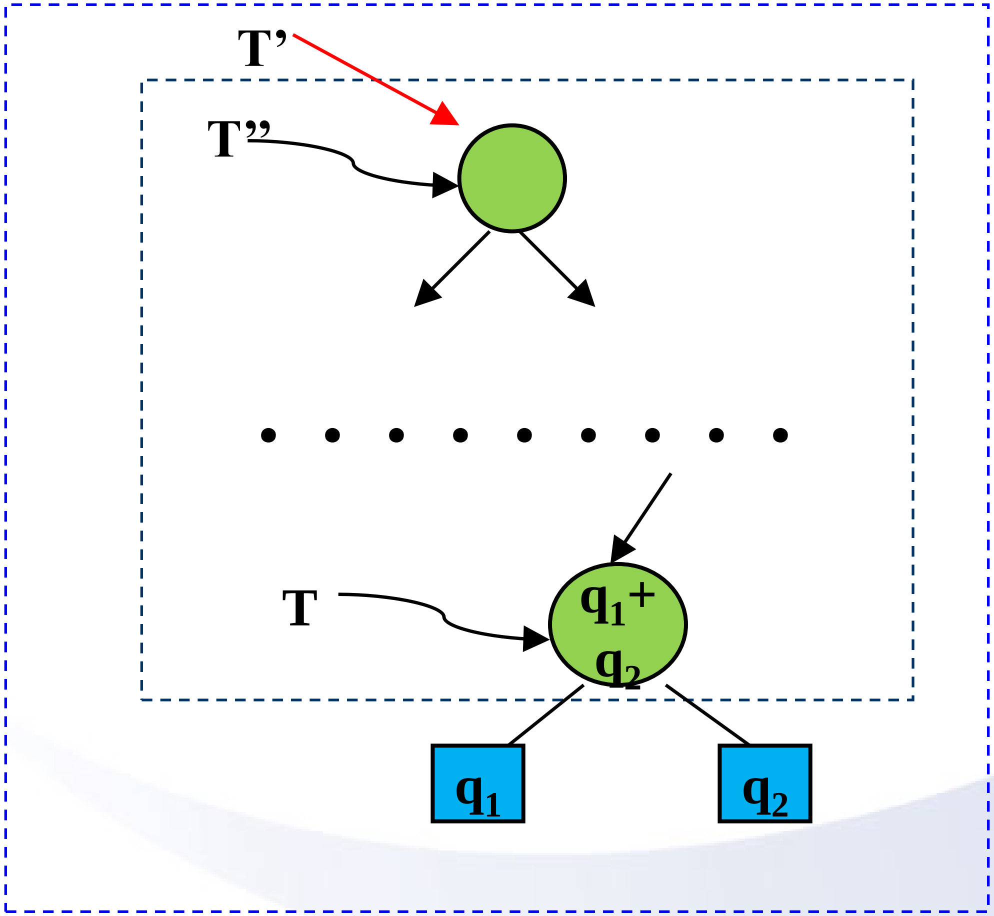
$$= (q_3-q_1)d_1 + (q_4-q_2)d_1 + (q_1-q_3)d_2 + (q_2-q_4)d_3$$

$$= (q_3-q_1)(d_1-d_2) + (q_4-q_2)(d_1-d_3)$$

$$\ge 0$$

4.4 最优归并模式

■ 4. 最优解的证明



$$F(T') = F(T'') + q_1 + q_2$$

则，

$$F_{min}(T') = \min(F(T'))$$

$$= \min(F(T'') + q_1 + q_2)$$

$$= \min(F(T'')) + q_1 + q_2$$

$$= F_{min}(T'') + q_1 + q_2$$

4.4 最优归并模式

■ 5. k路归并模式

- 每次同时归并k个文件。
- k元归并树：可能需要增加“虚”结点，以补充不足的外部结点。
  - 如果一棵树的所有内部结点的度都为k，则外部结点数n满足 n mod (k-1) = 1。
  - 对于满足 n mod (k-1) =1的整数n，存在一棵具有n个外部结点的k元树T，且T中所有结点的度为k。
  - 至多需要增加k-2个外部结点。
- k路最优归并模式得贪心规则：
  - 每一步选取k棵具有最小长度的子树归并。

4.4 最优归并模式

- **6. Huffman编码**
  - 编码：将原码转成ASCII码，然后按其二进制编码。

字符	ASCII码	正被考虑作业
'A'	65	01000001
'B'	66	01000010
'C'	67	01000011
....	....	....
'Z'	90	01011010

4.4 最优归并模式

■ 6. Huffman编码

- □编码：将原码转成ASCII码，然后按其二进制编码。

ZERO

01011010 01000101 01010010 01001111

0101101001000101010100100100 1111

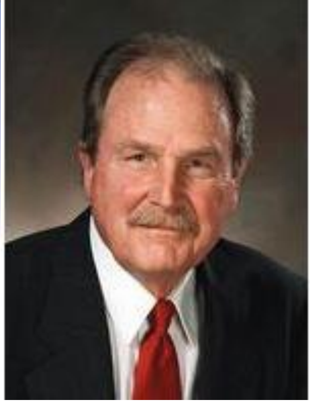
4.4 最优归并模式

- **6. Huffman编码**
  - 据统计：英文字母出现概率如下：

E出现的概率远大于Z，把他们定成相同长度就成了浪费。

空格	E	T	O	A	N	I	R	S
0.196	0.105	0.072	0.065	0.063	0.059	0.055	0.054	0.052
H	D	L	C	F	U	M	P	Y
0.047	0.035	0.029	0.023	0.0225	0.0225	0.021	0.0175	0.012
W	G	B	V	K	X	J	Q	Z
0.012	0.011	0.0105	0.008	0.003	0.002	0.001	0.001	0.001

4.4 最优归并模式


Huffman

- **6. Huffman编码**
  - Huffman编码是可变字长编码(VLC)的一种。
  - 1952年，David A. Huffman在麻省理工攻读博士时所提出一种编码方法，并发表于《一种构建极小多余编码的方法》(A Method for the Construction of Minimum-Redundancy Codes)一文。
  - 该方法完全依据字符出现概率来构造异字头的平均长度最短的码字，有时称之为最佳编码，一般就叫作Huffman编码。

4.4 最优归并模式

■ 6. Huffman编码

◻编码步骤

① 将信源符号的概率按减小的顺序排队。

② 把两个最小的概率相加，并继续这一步骤，始终将较高的概率分支放在上部，直到最后变成概率1。

③ 将每对组合的上边一个指定为1，下边一个指定为0（或相反）。

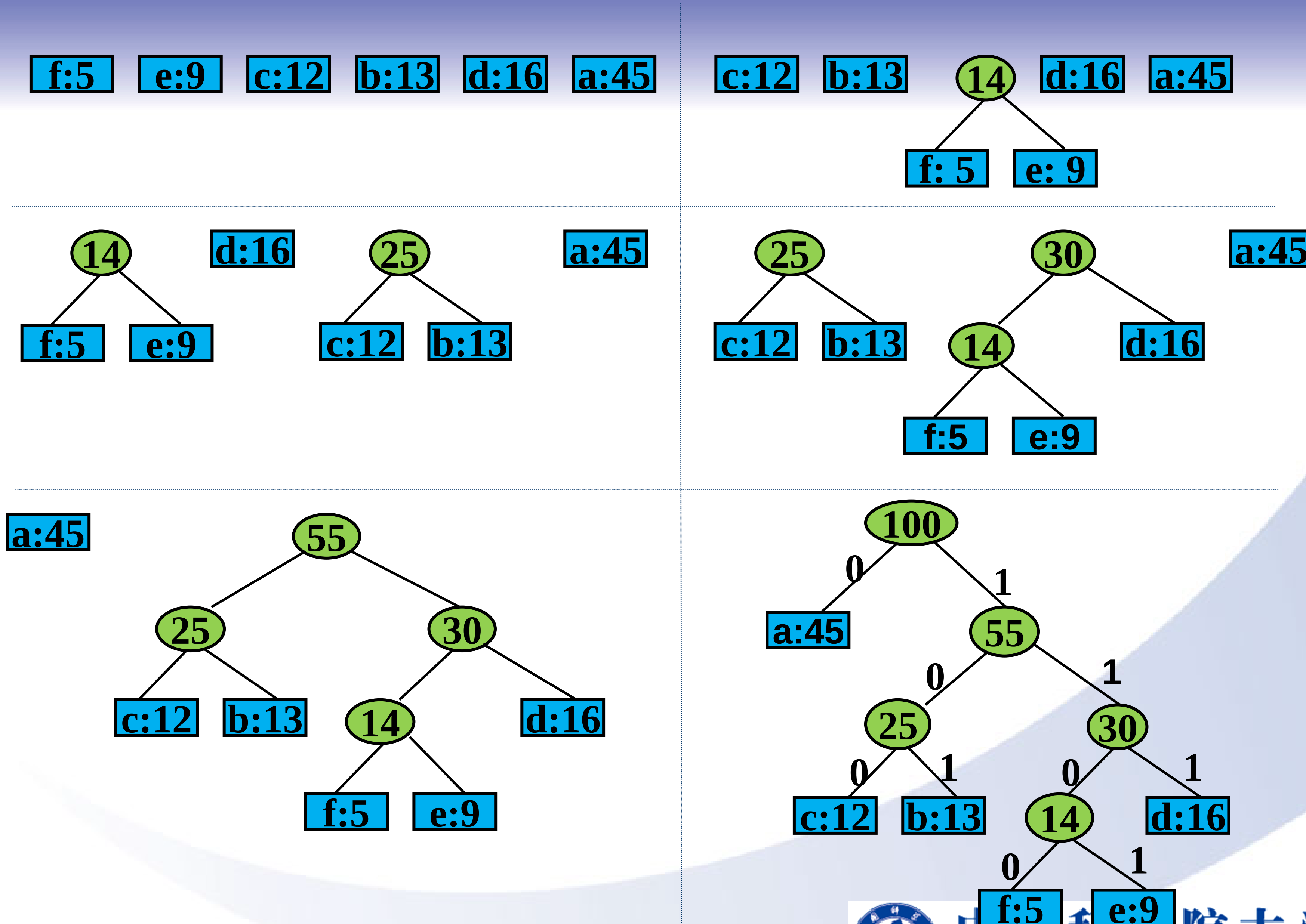
④ 画出由概率1处到每个信源符号的路径，顺序记下沿路径的0和1，所得就是该符号的霍夫曼码字。

4.4 最优归并模式

■ 6. Huffman编码

- 例：长为100000的文件出现六种字符。频率分布如下表。求这六种字符的霍夫曼码和平均码长。

不同的字符	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

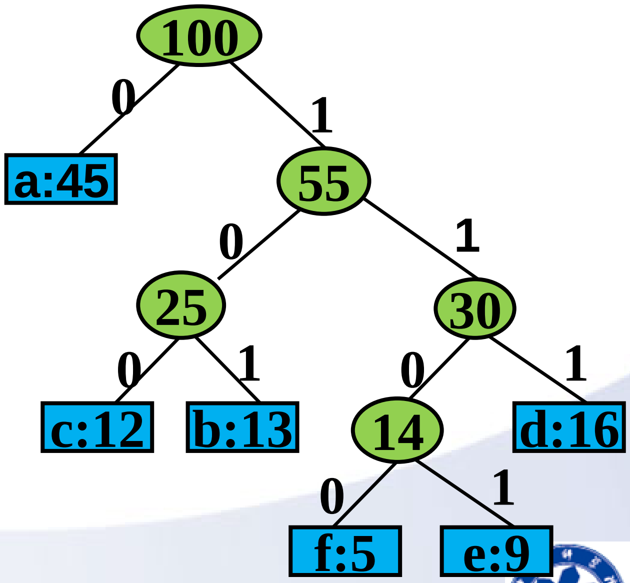


f:5
e:9
c:12
b:13
d:16
a:45
c:12
b:13
14
d:16
a:45
f: 5
e: 9
14
d:16
25
a:45
f:5
e:9
c:12
b:13
25
30
a:45
c:12
b:13
14
d:16
f:5
e:9
a:45
55
25
30
c:12
b:13
14
d:16
f:5
e:9
100
0
1
a:45
55
0
1
25
30
0
1
0
1
c:12
b:13
14
d:16
0
1
f:5
e:9

4.4 最优归并模式

■ 6. Huffman编码

- 例:长为100000的文件出现六种字符。频率分布如下表。求这六种字符的霍夫曼码和平均码长。
  - 平均码长
    =(45*1+13*3+12*3+16*3+9*4+5*4)/100
    = 2.24



作业-课后练习13

- 问题描述
  - 字符a~h的出现频率恰好是前8个斐波那契数。
  - 求他们的Huffman编码。
  - 将结果推广到n个字符的频率恰好是前n个斐波那契数的情况。

End