最优二叉树（哈夫曼树）

宁华

最优二叉树（哈夫曼树）

- 也叫最优搜索树、哈夫曼树(Huffman Tree)

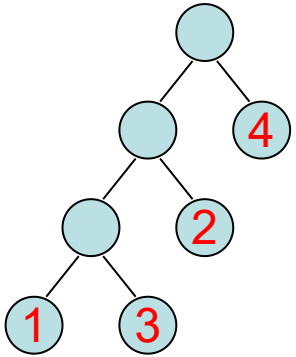
预备知识：二叉树的带权路径长度

- 设二叉树具有n个带权值的叶子结点，那么从根结点到各个叶子结点的路径长度与相应结点权值的乘积的和，叫做二叉树的带权路径长度，记作：

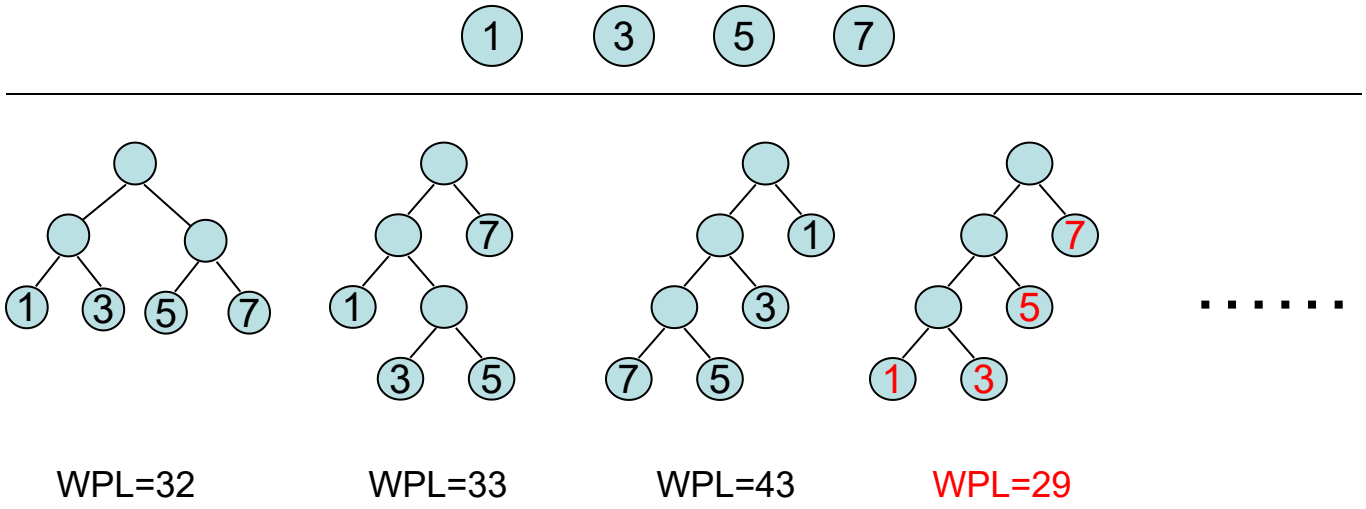
$$WPL = \sum_{i=1}^{n} Wi \times Li$$

- 其中Wi为第i个叶子结点的权值，Li为第i个叶子结点到根的路径长度.

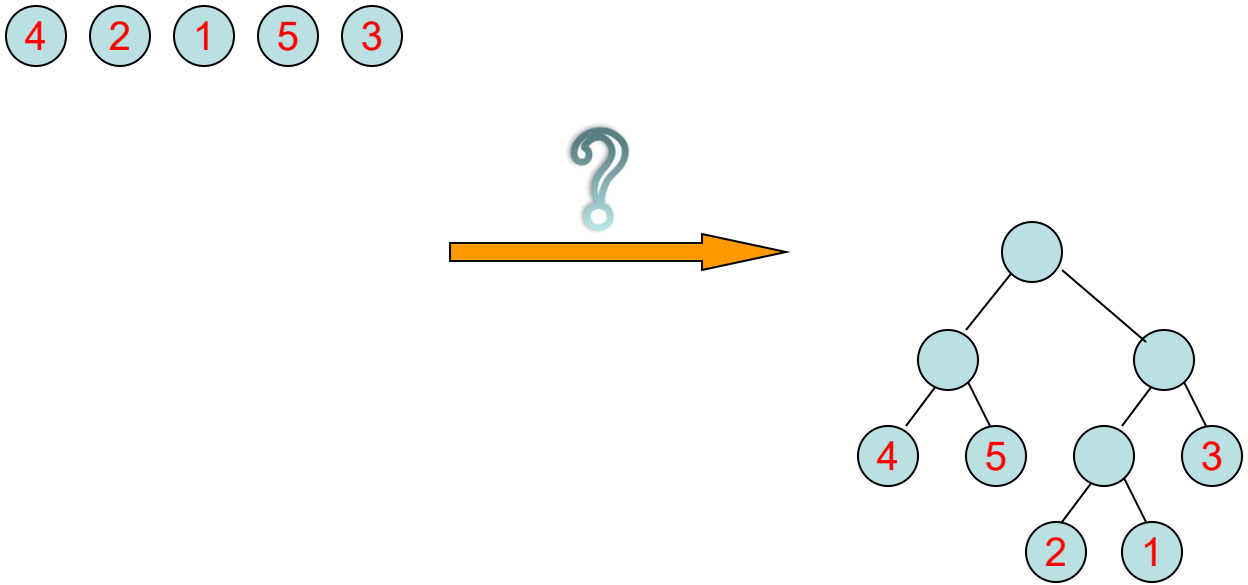
- 例如：
- WPL=1*3+3*3+2*2+4*1=20



- 如果给定一组具有确定权值的叶子结点，可以构造出不同的带权二叉树，它们的带权路径长度并不一定相同.
- 具有最小带权路径长度的二叉树称为**最优二叉树，也叫哈夫曼树**.



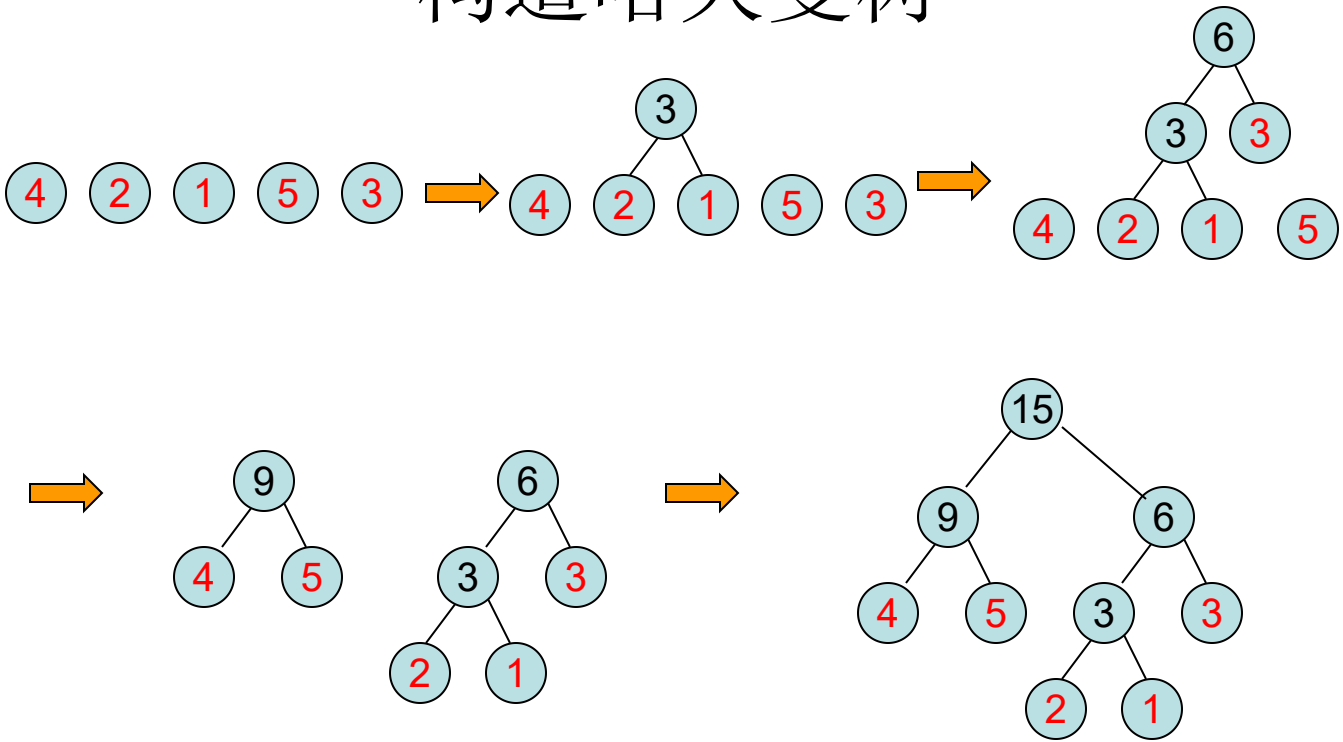
如何构造哈夫曼树



如何构造哈夫曼树

- 贪心

构造哈夫曼树



构造方法

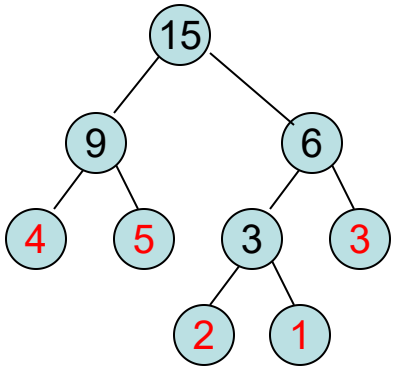
- 假定给出n个结点k_i(i=1..n)，其权值分别为w_i(i=1..n)。要构造以此n个结点为叶结点的最优二叉树，其构造方法如下：
- 首先，将给定的n个结点构成n棵二叉树的集合F={T_1，T_2，……，T_n}。其中每棵二叉树T_i中只有一个权值为w_i的根结点k_i，其左、右子树均为空。然后做以下两步
- ⑴在F中选取根结点权值最小的两棵二叉树作为左右子树，构造一棵新的二叉树，并且置新的二叉树的根结点的权值为其左、右子树根结点的权值之和；
- ⑵在F中删除这两棵二叉树，同时将新得到的二叉树加入F中；
- 重复⑴、⑵，直到在F中只含有一棵二叉树为止。这棵二叉树便是最优二叉树。

- 以上构造最优二叉树的方法称为**哈夫曼**（huffmann）算法。

代码实现

- 给出n个叶结点，构造二叉树，求最小WPL值。

- 样例输入：
- 5
- 4 2 1 5 3
- 样例输出：
- 33



问题

- 如何选择两个权值最小的根结点？
- 每次排序，找两个最小的？效率太低！
- 可借助于二叉堆（或优先队列）

小结

- 1、最优二叉树只含有度为0和度为2的点。
- 2、n个初始点（叶结点），共需要合并n-1次。每次合并得到一个新结点，最终的树含有n+(n-1)=2n-1个点。
- 3、根结点的权值，等于所有叶结点的权值和。
- 4、所有分支结点（非叶结点、度为2的点）的权值和，即为最小的WPL。例如NOIP2004合并果子，答案即是最小的WPL。

编程练习：构造哈夫曼树，输出其广义表

- 给出带权叶子结点，构建哈夫曼树，输出其广义表（括号表示）。（哈夫曼树结构不唯一，故答案不唯一）
- 【输入样例】
- 5
- 16 2 18 16 23
- 【输出样例】
- (75(34(16,18(2,16)),41(18,23)))
- 或
- (75(34(16,18),41(18(2,16),23)))
- 等等。

数据结构

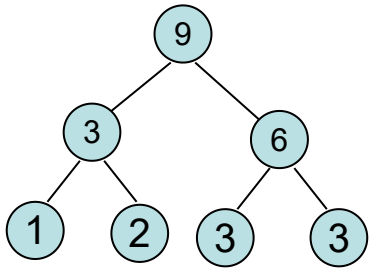
- 在最优二叉树中非叶结点的度均为2，因此采用顺序存储结构为宜。如果带权叶结点数为n个，则最优二叉树的结点数为2*n-1个。由此，最优二叉树的数据类型定义：
- const int N=107;
- struct node{
- 　　int data,fa,left,right;
- };
- node a[N<<1];
- 在最优二叉树的顺序存储结构中前n个结点为叶结点。新产生结点依次存储在a[n+1]….a[2*n-1]中。

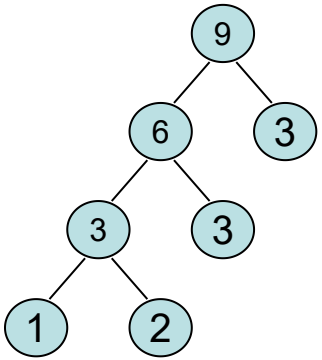
思考

- 哈夫曼树唯一吗？

思考

- 哈夫曼树唯一吗？





例题

- 在有N个叶子结点的哈夫曼树中，其结点总数为（ ）。
- A、不确定
  B、2N-1
  C、2N+1
  D、2N

分析

- 如果这道题目里面的哈夫曼树是指二叉的话，那么答案是B，如果不确定是几叉的话，那么是A。
- 通常提到哈夫曼树，如无特殊说明，我们一般认为是二叉树。

- 无论哈夫曼树是几叉，其特点是一致的（假设为m叉），即树中只存在度为0的结点（即叶结点）和度为m的结点。不妨设度为0的结点个数为x，度为m的结点个数为y，则存在一个等式x+y=m*y+1，x+y是树的总结点个数，m*y+1可以理解为每个结点都由其父结点发出的一条边得到（除了根结点），度为m的结点共y个，所以共发出m*y条边。

- 就这道题来说，假设哈夫曼树是二叉的话，则度为0的结点个数x=N，根据上述等式，x+y=2y+1，则度为2的结点个数y=N-1，则结点总数为2N-1。

哈夫曼树的应用

- 哈夫曼编码

- 了解两个概念：编码和解码

- 数据压缩过程称为编码。即将文件中的每个字符均转换为一个唯一的二进制位串。
- 数据解压过程称为解码。即将二进制位串转换为对应的字符。

引例

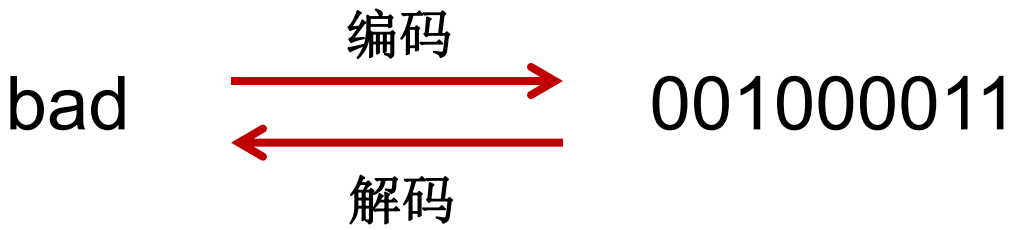
- 一篇文章只包含abcdef六个小写英文字母，现在要把这篇文章以二进制形式保存到计算机中。

- 如何编码？

一种编码方案

- 比如，对字符集{a,b,c,d,e,f}设置如下编码方案：

- a---000
- b---001
- c---010
- d---011
- e---100
- f ---101



等长码

- 等长编码方案：
- 将给定字符集C中每个字符的码长定为[log|C|]，|C|表示字符集的大小。
- 【例】
- 设待压缩的数据文件共有100000个字符，这些字符均取自字符集C={a，b，c，d，e，f}，等长编码需要3位二进制数字来表示六个字符，因此，整个文件的编码长度为300000位。

填空题

- 一篇文章只包含英文字母（大小写均有）和逗号、句号、顿号、问号、感叹号、空格，不包含其他字符。现在要给每个字符编码，采用等长编码方案，则每个字符的码长至少为_____。

思考

- 等长码

- 优点：
- 缺点：

变长码

- 变长编码方案将频度高的字符编码长度设置较短，将频度低的字符编码长度设置较长。

字符出现的频率表

	a	b	c	d	e	f
出现频率(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

- aaabbae
- 定长码：000 000 000 001 001 000 010
- 变长码：0 0 0 101 101 0 1101

- 根据计算公式：
- (45*1+13*3+12*3+16*3+9*4+5*4)*1000=224000
- 整个文件被编码为224000位，比等长编码方式节约了约25%的存储空间。

	优点	缺点
定长码		
变长码		

二义性

- 注意：
- 变长编码可能使解码产生二义性。产生该问题的原因是某些字符的编码可能与其他字符的编码开始部分(称为前缀)相同。
- 【例】
- 设E、T、W分别编码为00、01、0001，则解码时无法确定信息串0001是ET还是W。

思考

- 如何避免二义性？

- 前缀码：对每一个字符规定一个0, 1串作为其编码，并要求任一字符的编码都不是其他字符编码的前缀。

- 注意：等长编码是前缀码。下表中的变长编码也是前缀码。

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

最优前缀码

- 平均码长或文件总长最小的前缀编码称为最优的前缀码。最优前缀码对文件的压缩效果亦最佳。

- 平均码长=∑Pi*Li （1<=i<=n)
- 其中：Pi为第i个字符的概率，Li为码长
- 若将上表所示的文件作为统计的样本，则a至f六个字符的概率分别为0.45，0.13，0.12，0.16，0.09，0.05，对变长编码求得的平均码长为2.24，优于定长编码(平均码长为3)。

问题

- 如何得到最优前缀码编码方案？

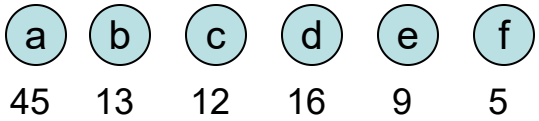
- 【例】
- 有一数据文件共有n（n<=100000）个字符，这些字符均取自字符集C={a，b，c，d，e，f}，等长编码需要三位二进制数字来表示各个字符，因此，整个文件的编码长度为3*n位。为了节省存储空间，现在对文件进行压缩。请给出一种编码方案，使所需存储空间最少。

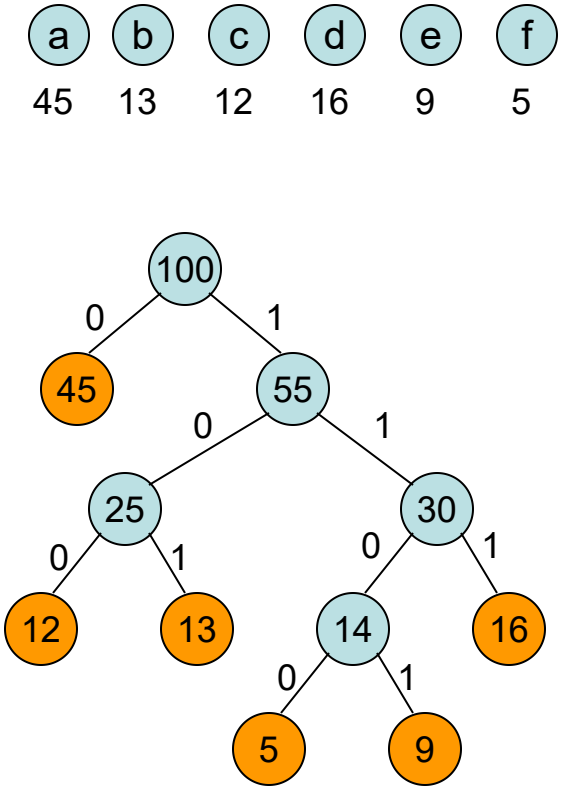
编程题

- 输入：
- 一行，若干个字符，每个字符均取自字符集{a,b,c,d,e,f}
- 输出：
- 采用最优前缀码编码方案对文件压缩后的文件总长度
- 样例输入：
- abbcccddddeeeeeffffff
- 样例输出：
- 51

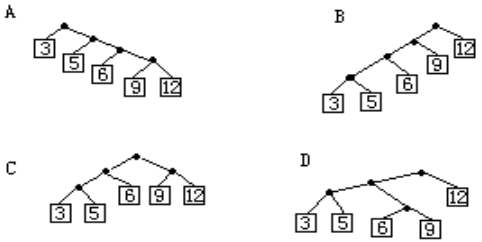
变长码

- 如何编码？

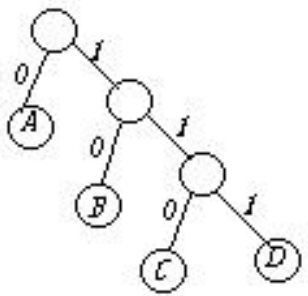



a
b
c
d
e
f
45
13
12
16
9
5
100
0
1
45
55
0
1
25
30
0
1
0
1
12
13
14
16
0
1
5
9

1）.给定一个整数集合{3，5，6，9，12}，下列二叉树哪个是该整数集合对应的霍夫曼（Huffman）树。 （ ）



2）、已知如图所示的哈夫曼树，那么电文CDAA的编码是[ ]



A 110100 B 11011100

C 010110111 D 11111100

3）、若以{4,5,6,7,8}作为叶子结点的权值构造哈夫曼树，则其带权路径长度是[ ]

A 、69 B、 70 C 、 73 D、 68

例：合并果子（NOIP2004提高组第二题）

- 【问题描述】
- 在一个果园里，多多已经将所有的果子打了下来，而且按果子的不同种类分成了不同的堆。多多决定把所有的果子合成一堆。
- 每一次合并，多多可以把两堆果子合并到一起，消耗的体力等于两堆果子的重量之和。可以看出，所有的果子经过n-1次合并之后，就只剩下一堆了。多多在合并果子时总共消耗的体力等于每次合并所耗体力之和。
- 因为还要花大力气把这些果子搬回家，所以多多在合并果子时要尽可能地节省体力。假定每个果子重量都为1，并且已知果子的种类数和每种果子的数目，你的任务是设计出合并的次序方案，使多多耗费的体力最少，并输出这个最小的体力耗费值。
- 例如有3堆果子，数目依次为1，2，9。可以先将第1、2堆合并，新堆数目为3，耗费体力为3。接着，将新堆与原先的第3堆合并，又得到新的堆，数目为12，耗费体力为12。所以多多总共耗费体力=3+12=15。可以证明15为最小的体力耗费值。

- 【输入】
- 第一行：一个整数n(1<＝n<=10000)，表示果子的种类数;
- 第二行：包含n个整数，用空格分隔，第i个整数ai(1<＝ai<=20000)是第i种果子的数目。
- 【输出】
- 一行，只包含一个整数，也就是最小的体力耗费值。输入数据保证这个值小于2^{31}。
- 【样例输入】
- 3
- 1 2 9
- 【样例输出】
- 15
- 【数据规模】
- 对于30％的数据，保证有n<=1000：
- 对于50％的数据，保证有n<=5000；
- 对于全部的数据，保证有n<=10000。

分析

- 经典的Huffman树问题。
- 把给定的n堆果子看作n个结点，每个结点有一个权值W[i]，合并相当于把它们两个合并为树，假设每个结点从根到它的距离是D[i]，使得最终∑(W[i]*D[i])最小。构造方法如下：

- 1.从森林里取两个权和最小的根结点；
- 2.将它们的权和相加，得到新的结点，并且把原来的两个结点作为新结点的儿子结点；
- 3.重复以上操作，直到整个森林中只剩下一棵树。

POJ 3253 Fence Repair

- 题意：
- FJ要把一个木板锯成n块指定长度的小木板，每次锯都会把一个大木板锯成两个小木板，而且每次锯都要收取一定的费用，这个费用就是当前锯的这个大木板的长度。求最小费用。（初始木板的长度恰好等于n块目标小木板的长度总和，且锯割无损失）
- Sample Input
- 3
- 8
- 5
- 8
- Sample Output
- 34

UVA 12676 Inverting Huffman

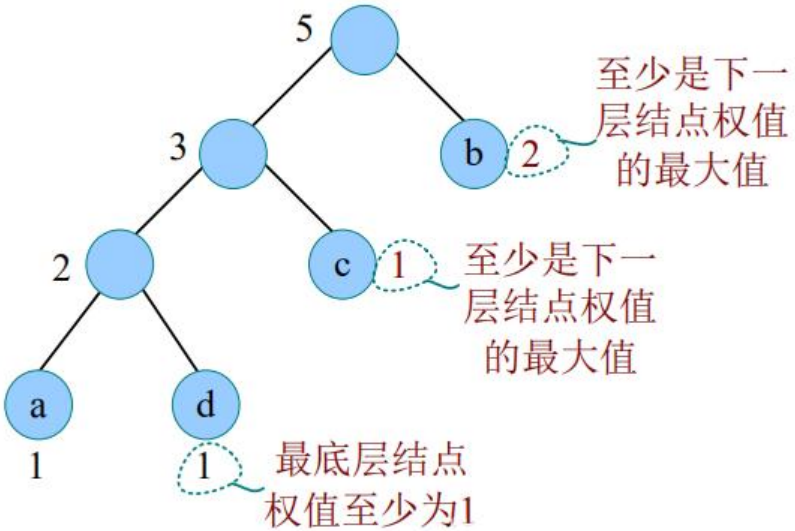
- Description
- 一串文本中包含 N 个不同字符，经过哈夫曼编码后，得到这 N 个字符的相应编码长度，求文本的最短可能长度。多组数据。
- Sample Input
- 2
- 1 1
- 4
- 2 2 2 2
- 10
- 8 2 4 7 5 1 6 9 3 9
- Sample Output
- 2
- 4
- 89

- Sample Input
- 4
- 3 1 2 3

- Sample Output
- 5

- Sample Input
- 4
- 3 1 2 3

- Sample Output
- 5



两个难度稍大的题目

- NOI2015 荷马史诗
- Codeforces 700D Huffman Coding on Segment