哈（霍）夫曼（Huffman）编码

引入：
有一段文言文，假设只有“之乎者也”四个字，想要通过二进制哈夫曼编码，则可以这样做。
哈夫曼编码，是根据概率来进行编码的。

假设，这四个字的出现频率如下：

之	出现 700 次			
乎	出现 600 次			
者	出现 300 次			
也	出现 200 次			

假设，出现频率高的用 1 表示，频率低的用 0 表示：

之	出现 700 次	1		
乎	出现 600 次	0	1	
者	出现 300 次	0	0	1
也	出现 200 次	0	0	0

所以，得到的编码如下：
之 1
乎 01
者 001
也 000

提问：
现在收到一段电波，请问是什么意思？
1010100001001
答案：之乎乎也乎者

提问：
这个电波，还有第二种解法吗？
并没有。所以，这种编码是唯一的。这样才是我们希望的结果。

提问：哈夫曼编码的特点？
1. 码的长度不固定
2. 码是唯一的（也称为“即时码”）

推广应用：文件压缩 – 文本

有如下文本：
之乎之乎之乎者乎者者
之之乎乎也乎者之乎也
者之乎者之乎也也之乎
之乎乎者乎者也乎

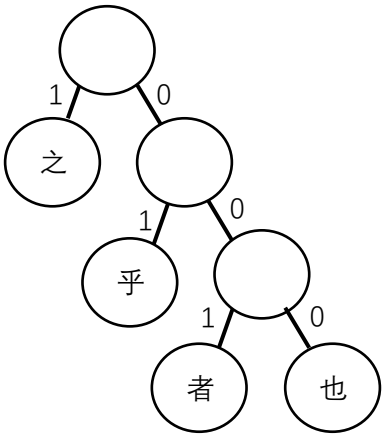
提问：一共 38 个汉字，请问需要多少存储空间？（不考虑回车）
答：38*2=76 Bytes

提问：使用哈夫曼编码，需要多少存储空间？（不考虑回车）
答：
之 10 1 bit
乎 15 2 bits
者 8 3 bits
也 5 3 bits
SUM = 10+30+24+15 = 79 bits
但是，内存一般都是 8 的倍数，所以整合一下，应该是 80bits，10 Bytes
节约了（76-10）/76 = 86.8% 的空间。

哈夫曼树

之	出现 700 次	1		
乎	出现 600 次	0	1	
者	出现 300 次	0	0	1
也	出现 200 次	0	0	0

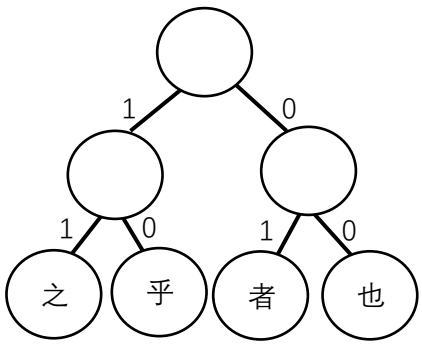
画出上面“之乎者也”的树状结构:



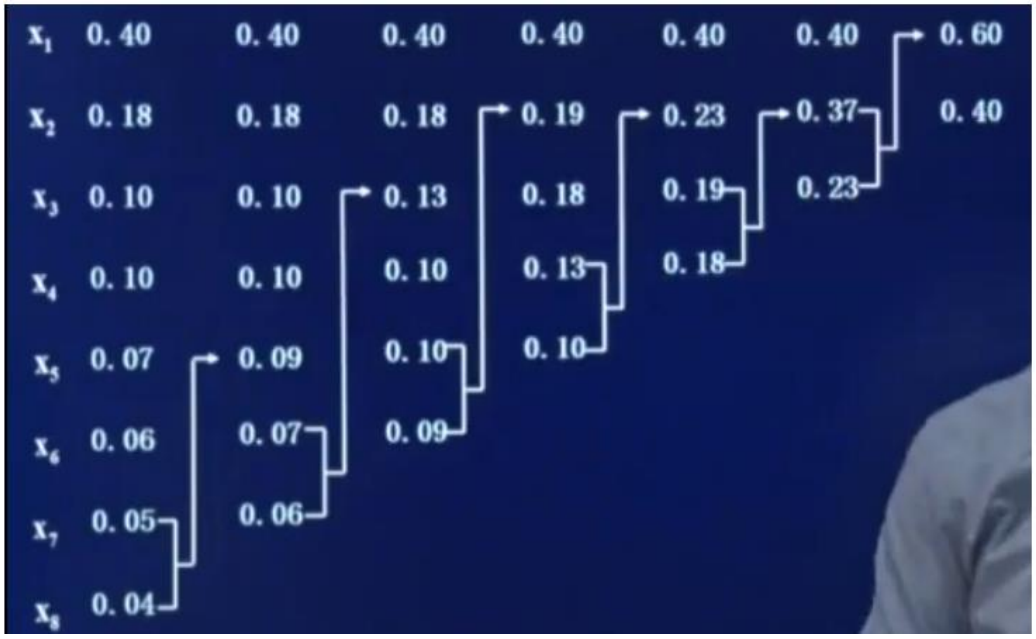
特点:

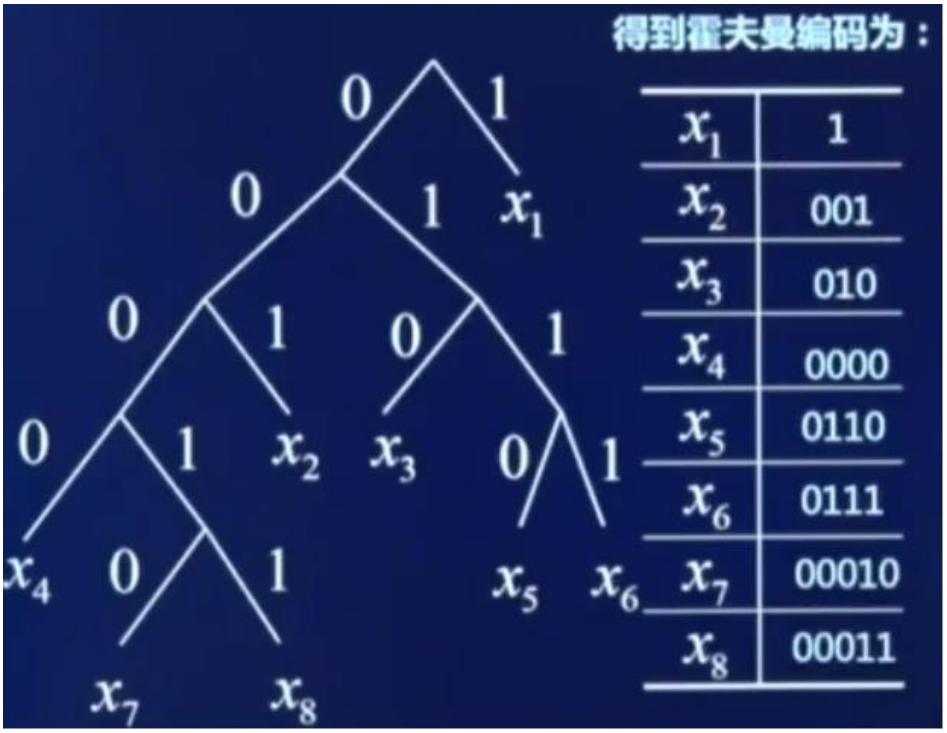
1. 叶节点
2. 深度太大

提问: 然后怎么优化树结构?
把一些概率较小的点整合起来。



字符	x1	x2	x3	x4	x5	x6	x7	x8
概率	0.4	0.18	0.1	0.1	0.07	0.06	0.05	0.04





然后怎么才能生成一个满二叉树呢?
答：按照数量划分