霍夫曼编码

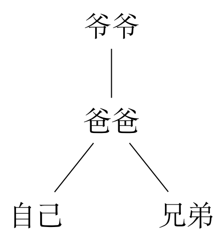
Huffman Coding

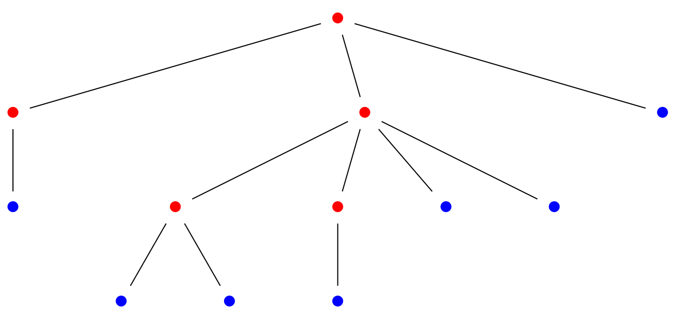
刘卓

1 定义

定义 1: *有根树 (rooted tree) 是指一个顶点 (vertex) 被指定为根的连通有向图 (graph)，它没有入边 (edges)，而其他顶点只有一条入边。*

例 1:





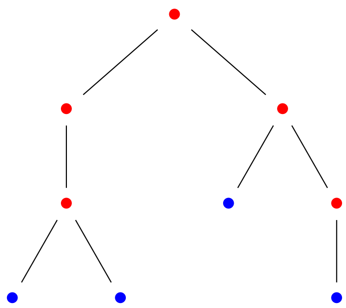
其中 • 为叶子 (leaves)，• 为内部顶点 (internal vertex)。

□

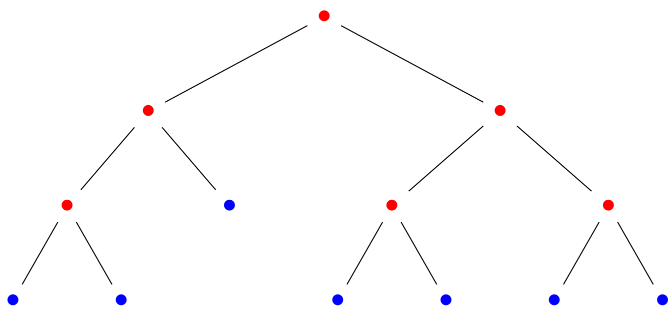
定义 2: *二叉树 (binary tree) 是一棵有根的树，其中每个内部顶点不超过 2 个孩子 (children)。全二叉树是一棵根树每个内部顶点都有两个孩子。*

例 2:

不完全二叉树:



完全二叉树:



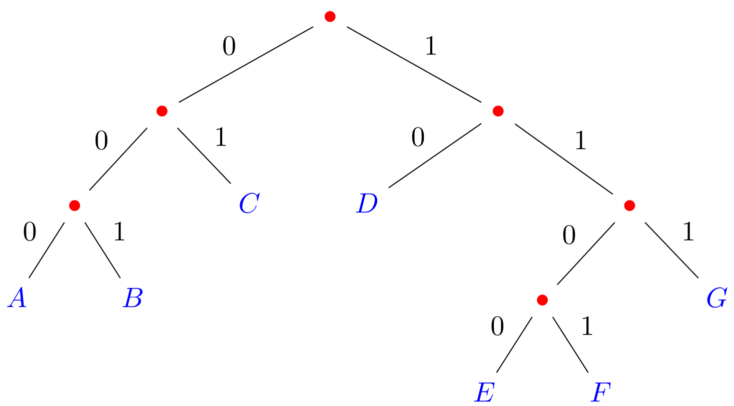
$\square$

定义 *3*: 如果两个码字的连接包含一个有效码字，且两者重叠，则这个二进制码字是无逗号 *(comma-free)*

例 3:

令:

$$A = 000, B = 001, C = 01, D = 10, E = 1100, F = 1101, G = 111$$



$\square$

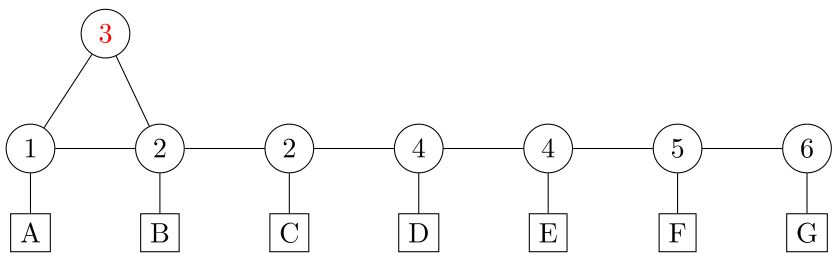
问: 给定一个文件的字母频率，哪棵树需要最少的 bit?
下面的算法给出了最优树:

1. 用一个节点/顶点替换每个字母，并根据每个字母的频率标记这些节点。然后，从左到右读取时，按节点值递增的顺序对节点进行排序。
2. 从左到右，把两个最小的数组合在一起，用它们的和代替它们。
3. 再次根据节点的值对结果节点进行排序。然后重复这些步骤，直到所有节点都连接好。
4. 一旦我们获得了二叉树，用相应的字母替换顶点数。然后我们把分支标记为左边是 0，右边是 1。
5. 最后，我们沿着路径跟踪以获得每个字母的代码。

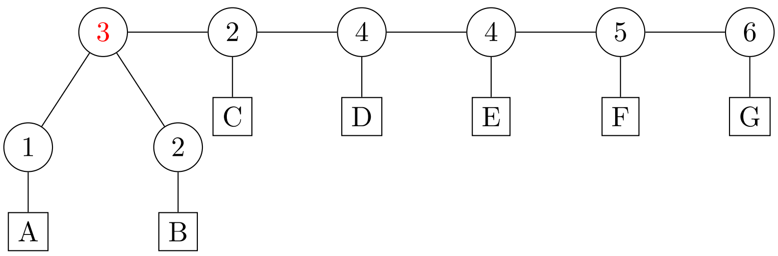
例 4：假设某个文件只包含以下频率的字母

A	B	C	D	E	F	G
1	2	2	4	4	5	6

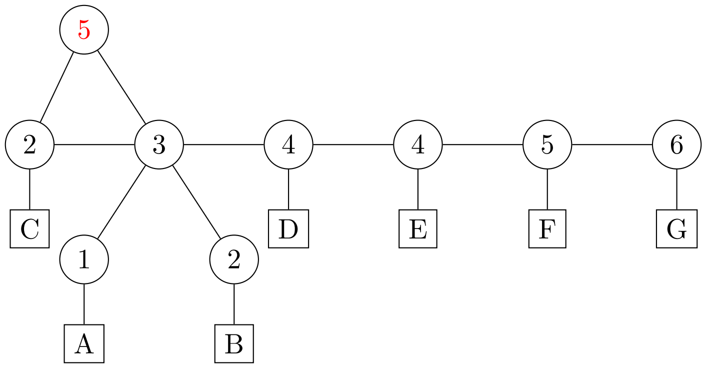
构造使您能够压缩文件的无逗号代码，以便您可以使用最少的位来存储它。
解:
第一步:



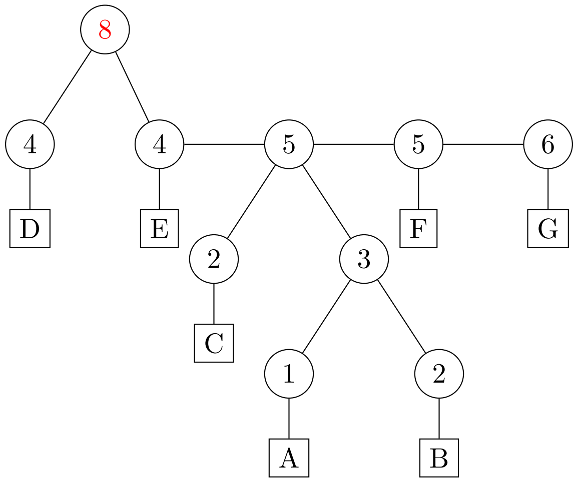
第二步:



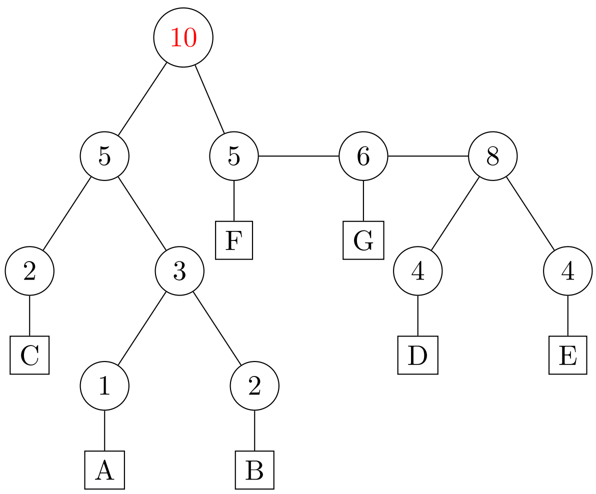
第三步:


5
2
3
4
4
5
6
C
D
E
F
G
1
2
A
B

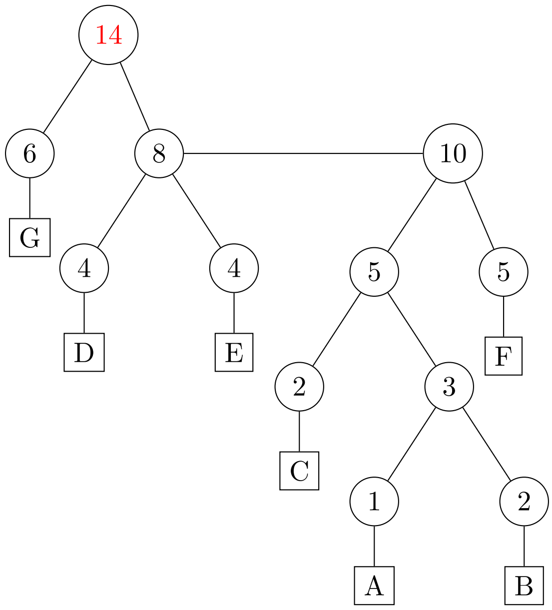
第四步：


8
4
4
5
5
6
D
E
F
G
2
3
C
1
2
A
B

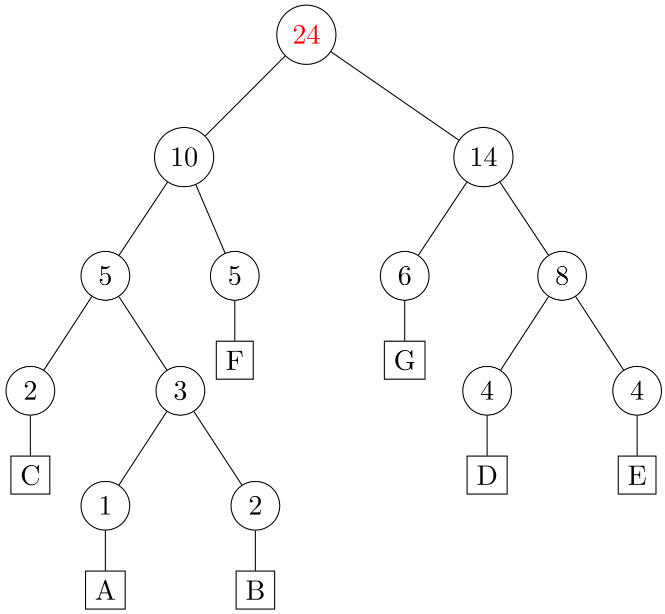
第五步：


10
5
5
6
8
F
G
2
3
4
4
C
D
E
1
2
A
B

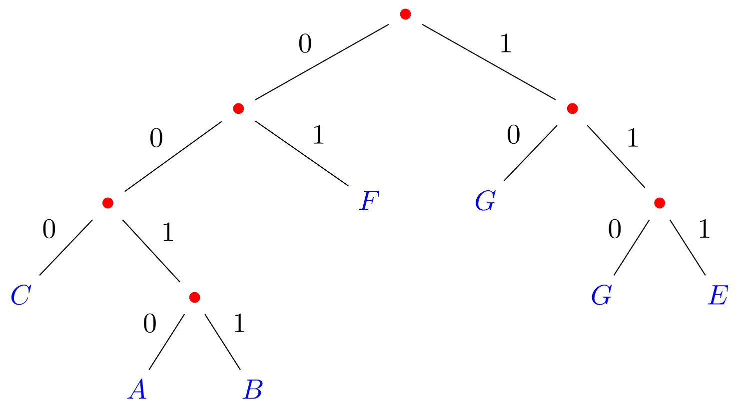
第六步：



第七步:



可转化为:



字母	A	B	C	D	E	F	G
频率	1	2	2	4	4	5	6
Code	0010	0011	000	110	111	01	10
Bits	4	4	3	3	3	2	2

因此加密后的长度为

$$4N_A + 4N_B + 3N_C + 3N_D + 3N_E + 2N_F + 2N_G = 64$$

因此相比未加密前的字符，平均每个字符需要 $\frac{\text{密文长度}}{\text{明文长度}}\frac{64}{24} \approx 2.66$ 个 bit 替代
其熵值为:

$$\begin{aligned}
\sum_{\alpha} \mathbb{P}(\alpha) \log_2 \left(\frac{1}{\mathbb{P}(\alpha)}\right) &= \frac{1}{24}\log_2\left(\frac{1}{1/24}\right) + \frac{2}{24}\log_2\left(\frac{1}{2/24}\right) + \frac{2}{24}\log_2\left(\frac{2}{1/24}\right) \\
&+ \frac{4}{24}\log_2\left(\frac{1}{4/24}\right) + \frac{4}{24}\log_2\left(\frac{1}{4/24}\right) \\
&+ \frac{5}{24}\log_2\left(\frac{1}{5/24}\right) + \frac{6}{24}\log_2\left(\frac{1}{6/24}\right) \\
&\approx 2.62165
\end{aligned}$$

□

定理 *1*: *假设明文中的字母数为* $n_1, n_2, \cdots, n_k$，*且设* $N = n_1 + \cdots + n_k$。*那么最佳代码长度 (以每个字母的比特数表示) 是*

$$H = \sum_{i=1}^{k} p_i \log_2 \left(\frac{1}{p_i}\right)$$

其中 $p_i = n_i/N$, $1 \le i \le k$。

定理 *2*: *由概率* $p_1, p_2, \cdots, p_k$ *产生的预期码长在熵的 1 bit 以内*

$$H = \sum_{i=1}^{k} p_i \log_2 \left(\frac{1}{p_i}\right)$$

其中 $p_i = n_i/N$, $1 \le i \le k$。

定理 *3:* 仅针对密文攻击

$$H(K|C) = H(K) + H(M) - H(C)$$

定理 *4:* 仅针对明文攻击

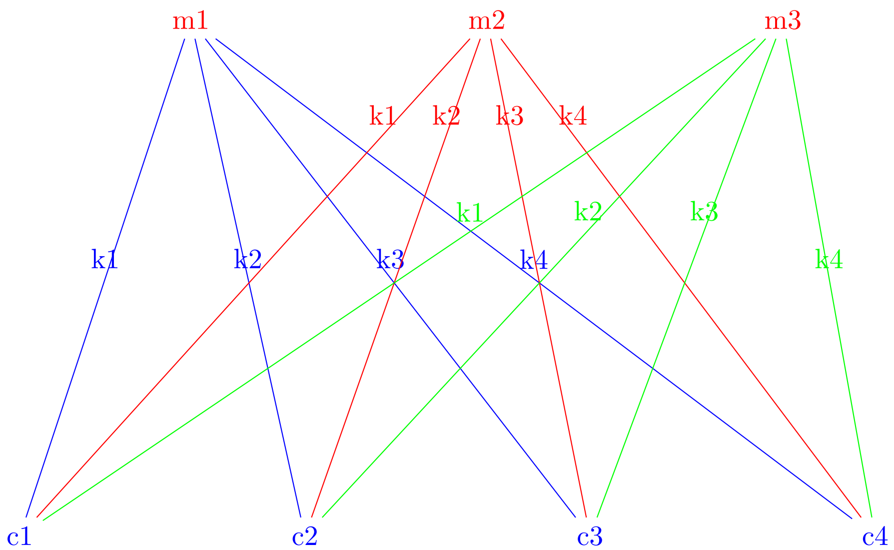
$$H(K|C,M) = H(K) + H(C|M)$$

2 随机密码系统

设计：

- 明文 $M = m_1, m_2, \cdots, m_N$
- 密钥 $K = k_1, k_2, \cdots, k_S$
- 密文 $C = c_1, c_2, \cdots, c_Q$
- 使用密钥 k 的加密公式：$c = E_k(m)$
- 使用密钥 k 的解密公式：$m = D_k(c)$

例 5
明文：m_1, m_2, m_3
密钥：k_1, k_2, k_3, k_4
密文：c_1, c_2, c_3, c_4



由此可知，一个密文，可能对应三种明文，破解难度加大。

□

3 完善保密性 Perfect Secrecy

定义 *4*: 如果密文不提供关于明文的信息，就说密码系统达到完全保密。即 M、C 为随机变量，即

$$\mathbb{P}\left(M = m_i \cap C = c_j\right) = \mathbb{P}\left(M = m_i\right) \cdot \mathbb{P}\left(C = c_j\right)$$

其中 $m_j \in M$, $c_j \in C$。
因此在一个完善的保密系统中:

- 密钥的数量至少必须与密文的数量一样大。
- 对于固定密钥: 不同的明文对应不同的密文。因此，密文的数量必须至少与明文的数量相等。

即：密钥数量 $\geq$ 密文数量 $\geq$ 明文数量

定理 *5:* *完全保密应该满足于:*

- *所有密钥概率应该相等*
- *对于加密过程 m_i 到 c_i，都为唯一的密钥 k 与之对应*

那么如果建立的一个完善的保密方法呢? 需要有以下几点要求:

- 密钥数量 = 密文数量 = 明文数量
- 所有密钥概率应该相等
- 加密矩阵是拉丁方图 (Latin square) 或加密图是完全二部图 (complete bipartite graph)。
  - 拉丁方图是一个 $n \times n$ 矩阵，其中整数从 1 到 n 在每一行和每一列中只出现一次。
  - 完全二部图是一个图，它的顶点集合被分解成两个不相交的子集，使得同一个子集中的两个顶点不连通，且每个顶点之间有一条边

例 6
明文：m_1, m_2, m_3
密钥：k_1, k_2, k_3
密文：c_1, c_2, c_3
拉丁方图:

	m_1	m_2	m_3
k_1	c_1	c_2	c_3
k_2	c_2	c_3	c_1
k_3	c_3	c_1	c_2

完全二部图: