最优树与哈夫曼算法

王丽杰

Email: ljwang@uestc.edu.cn

电子科技大学 计算机学院

2016-

引子

在计算机及通讯事业中，常用二进制编码来表示符号。

例如，可用 00、01、10、11 分别表示字母 A、B、C、D，这称作等长编码。这在四个字母出现频率基本相等的情况下是非常合理的。

引子

在计算机及通讯事业中，常用二进制编码来表示符号。

例如，可用 00、01、10、11 分别表示字母 A、B、C、D，这称作等长编码。这在四个字母出现频率基本相等的情况下是非常合理的。

但当四个字母出现的频率很不一样，如 A 出现的频率为 50%，B 出现的频率为 25%，C 出现的频率为 20%，D 出现的频率为 5% 时，使用等长编码就不是最优的方式了。

引子

在计算机及通讯事业中，常用二进制编码来表示符号。

例如，可用 00、01、10、11 分别表示字母 A、B、C、D，这称作等长编码。这在四个字母出现频率基本相等的情况下是非常合理的。

但当四个字母出现的频率很不一样，如 A 出现的频率为 50%，B 出现的频率为 25%，C 出现的频率为 20%，D 出现的频率为 5% 时，使用等长编码就不是最优的方式了。

如果此时我们使用不等长编码，如用 000 表示字母 D，用 001 表示字母 C，01 表示 B，1 表示 A。在同样传输 100 个字母的情况下，等长编码需 $2\times100=200$ 个二进制位，而不等长编码仅需 $3\times5+3\times20+2\times25+1\times50=175$ 个二进制位。

引子

在计算机及通讯事业中，常用二进制编码来表示符号。

例如，可用 00、01、10、11 分别表示字母 A、B、C、D，这称作等长编码。这在四个字母出现频率基本相等的情况下是非常合理的。

但当四个字母出现的频率很不一样，如 A 出现的频率为 50%，B 出现的频率为 25%，C 出现的频率为 20%，D 出现的频率为 5% 时，使用等长编码就不是最优的方式了。

如果此时我们使用不等长编码，如用 000 表示字母 D，用 001 表示字母 C，01 表示 B，1 表示 A。在同样传输 100 个字母的情况下，等长编码需 $2\times100=200$ 个二进制位，而不等长编码仅需 $3\times5+3\times20+2\times25+1\times50=175$ 个二进制位。

但不等长编码不能随意定义，否则会引起问题，如当我们用 1 表示 A，用 00 表示 B，用 001 表示 C，用 000 表示 D 时，如果接收到的信息为 001000，则无法辨别它是 CD 还是 BAD。

前缀码

Definition

前缀码

Definition

- 设 $a_1a_2\cdots a_{n-1}a_n$ 为长度为 n 的符号串，称其子串 $a_1, a_1a_2, \cdots, a_1a_2\cdots a_{n-1}$ 分别为 $a_1a_2\cdots a_{n-1}a_n$ 的长度为 $1, 2, \cdots, n-1$ 的前缀。

前缀码

Definition

- 设 $a_1a_2\cdots a_{n-1}a_n$ 为长度为 n 的符号串，称其子串 $a_1, a_1a_2, \cdots, a_1a_2\cdots a_{n-1}$ 分别为 $a_1a_2\cdots a_{n-1}a_n$ 的长度为 $1, 2, \cdots, n-1$ 的前缀。
- 设 $A = \{b_1, b_2, \cdots, b_m\}$ 是一个符号串集合，若对任意 $b_i, b_j \in A$，$b_i \neq b_j$，b_i 不是 b_j 的前缀，b_j 也不是 b_i 的前缀，则称 A 为前缀码。若符号串 $b_i(i = 1, 2, \cdots, m)$ 中，只出现 0 和 1 两个符号，则称 A 为二元前缀码。

前缀码

Definition

- 设 $a_1a_2\cdots a_{n-1}a_n$ 为长度为 n 的符号串，称其子串 $a_1, a_1a_2, \cdots, a_1a_2\cdots a_{n-1}$ 分别为 $a_1a_2\cdots a_{n-1}a_n$ 的长度为 $1, 2, \cdots, n-1$ 的前缀。
- 设 $A = \{b_1, b_2, \cdots, b_m\}$ 是一个符号串集合，若对任意 $b_i, b_j \in A$，$b_i \neq b_j$，b_i 不是 b_j 的前缀，b_j 也不是 b_i 的前缀，则称 A 为前缀码。若符号串 $b_i(i = 1, 2, \cdots, m)$ 中，只出现 0 和 1 两个符号，则称 A 为二元前缀码。

Example

前缀码

Definition

- 设 $a_1 a_2 \cdots a_{n-1} a_n$ 为长度为 n 的符号串，称其子串 $a_1, a_1 a_2, \cdots, a_1 a_2 \cdots a_{n-1}$ 分别为 $a_1 a_2 \cdots a_{n-1} a_n$ 的长度为 $1, 2, \cdots, n-1$ 的前缀。
- 设 $A = \{b_1, b_2, \cdots, b_m\}$ 是一个符号串集合，若对任意 $b_i, b_j \in A$，$b_i \neq b_j$，b_i 不是 b_j 的前缀，b_j 也不是 b_i 的前缀，则称 A 为前缀码。若符号串 $b_i (i = 1, 2, \cdots, m)$ 中，只出现 0 和 1 两个符号，则称 A 为二元前缀码。

Example

- $\{1, 01, 001, 000\}$ 是前缀码；

前缀码

Definition

- 设 $a_1a_2\cdots a_{n-1}a_n$ 为长度为 n 的符号串，称其子串 $a_1, a_1a_2, \cdots, a_1a_2\cdots a_{n-1}$ 分别为 $a_1a_2\cdots a_{n-1}a_n$ 的长度为 $1, 2, \cdots, n-1$ 的**前缀**。
- 设 $A = \{b_1, b_2, \cdots, b_m\}$ 是一个符号串集合，若对任意 $b_i, b_j \in A$，$b_i \neq b_j$，b_i 不是 b_j 的前缀，b_j 也不是 b_i 的前缀，则称 A 为**前缀码**。若符号串 $b_i(i = 1, 2, \cdots, m)$ 中，只出现 0 和 1 两个符号，则称 A 为**二元前缀码**。

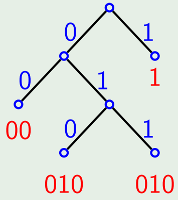
Example

- $\{1, 01, 001, 000\}$ 是前缀码；
- $\{1, 11, 001, 0011\}$ 不是前缀码。

用二元树产生二元前缀码

给定一棵二元树 T，假设它有 t 片树叶。设 v 是 T 任意一个分支点，则 v 至少有一个儿子，至多有两个儿子。若 v 有两个儿子，则在由 v 引出的两条边上，左边的标上 0，右边的标上 1；若 v 只有一个儿子，在 v 引出的边上可标 0 也可标 1。设 w 为 T 的任意一片树叶，从树根到 w 的通路上各边的标号组成的符号串放在 w 处，t 片树叶处的 t 个符号串组成的集合为一个二元前缀码。

Example



此二元树产生的前缀码为 $\{1, 00, 010, 011\}$

最优树

Definition

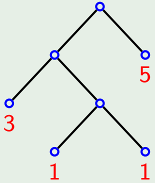
设有一棵二元树 T，若对其所有的 t 片叶赋以权值 $w_1, w_2, \cdots, w_t$，则称之为赋权二元树；若权为 w_i 的叶的层数为 $L(w_i)$，则称 $W(T) = \sum_{i=1}^{t} w_i \times L(w_i)$ 为该赋权二元树的权；而在所有赋权 $w_1, w_2, \cdots, w_t$ 的二元树中，$W(T)$ 最小的二元树称为最优树。

最优树

Definition

设有一棵二元树 T，若对其所有的 t 片叶赋以权值 $w_1, w_2, \cdots, w_t$，则称之为赋权二元树；若权为 w_i 的叶的层数为 $L(w_i)$，则称 $W(T) = \sum_{i=1}^{t} w_i \times L(w_i)$ 为该赋权二元树的权；而在所有赋权 $w_1, w_2, \cdots, w_t$ 的二元树中，$W(T)$ 最小的二元树称为最优树。

Example



则此赋权二元树的权为：

$$5 \times 1 + 3 \times 2 + 1 \times 3 + 1 \times 3 = 17$$

哈夫曼算法

1952 年哈夫曼 (Huffman) 给出了求最优树的方法。

哈夫曼算法：

1. 初始：令 $S = \{w_1, w_2, \cdots, w_t\}$；

哈夫曼算法

1952 年哈夫曼 (Huffman) 给出了求最优树的方法。

哈夫曼算法：

1. 初始：令 $S = \{w_1, w_2, \cdots, w_t\}$；
2. 从 S 中取出两个最小的权 w_i 和 w_j，画结点 v_i 和 v_j，分别带权 w_i 和 w_j。画 v_i 和 v_j 的父亲 v，令 v 带权 $w_i + w_j$；

哈夫曼算法

1952 年哈夫曼 (Huffman) 给出了求最优树的方法。

哈夫曼算法：

1. 初始：令 $S = \{w_1, w_2, \cdots, w_t\}$；
2. 从 S 中取出两个最小的权 w_i 和 w_j，画结点 v_i 和 v_j，分别带权 w_i 和 w_j。画 v_i 和 v_j 的父亲 v，令 v 带权 $w_i + w_j$；
3. 令 $S = (S - \{w_i, w_j\}) \cup \{w_i + w_j\}$；

哈夫曼算法

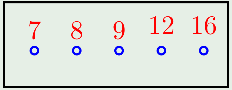
1952 年哈夫曼 (Huffman) 给出了求最优树的方法。

哈夫曼算法：

1. 初始：令 $S = \{w_1, w_2, \cdots, w_t\}$；
2. 从 S 中取出两个最小的权 w_i 和 w_j，画结点 v_i 和 v_j，分别带权 w_i 和 w_j。画 v_i 和 v_j 的父亲 v，令 v 带权 $w_i + w_j$；
3. 令 $S = (S - \{w_i, w_j\}) \cup \{w_i + w_j\}$；
4. 判断 S 是否只含一个元素？若是，则停止，否则转 2。

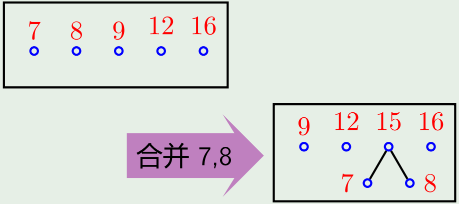
哈夫曼算法

Example



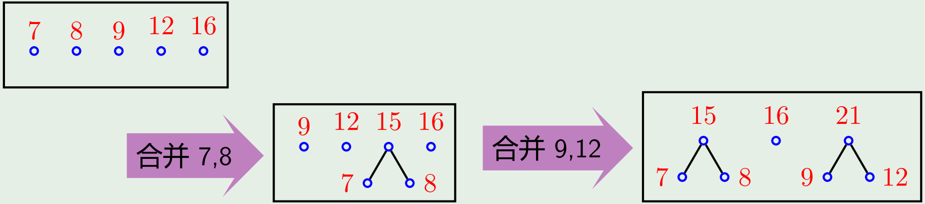
哈夫曼算法

Example



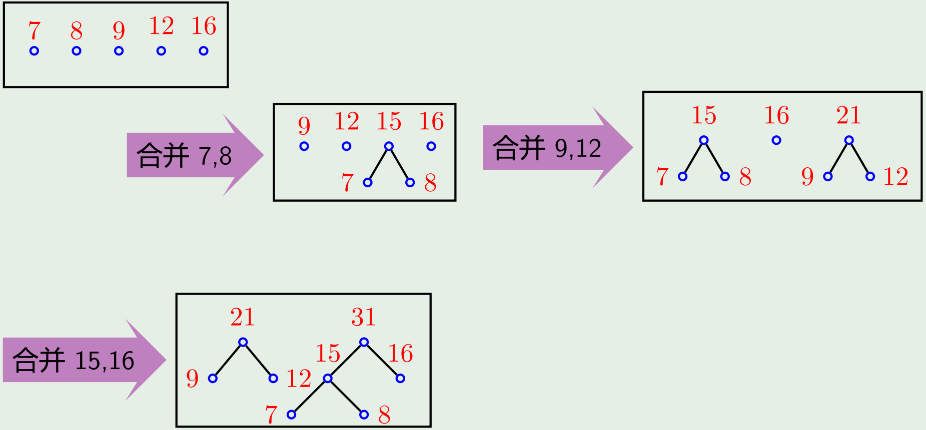
哈夫曼算法

Example



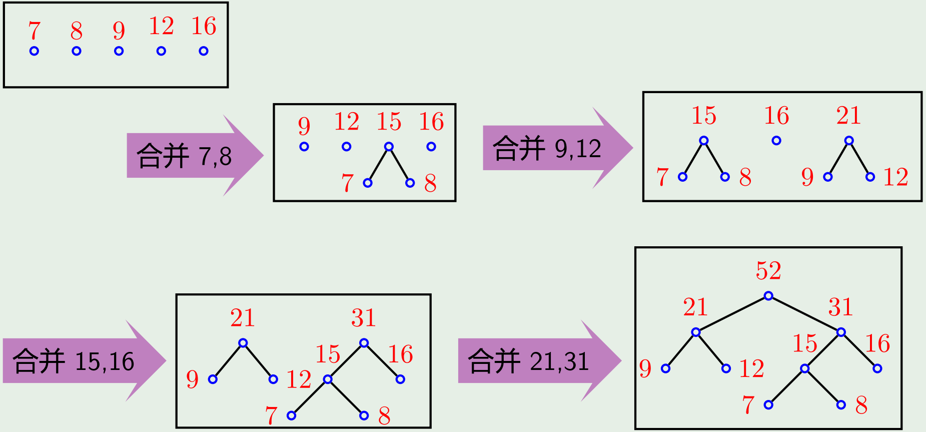
哈夫曼算法

Example



哈夫曼算法

Example



前缀码构造

Example

已知字母 A、B、C、D、E、F 出现的频率如下：

A—30%，B—25%，C—20% D—10%，E—10%，F—5%

构造一个表示 A、B、C、D、E、F 前缀码，使得传输的二进制位最少。

前缀码构造

Example

已知字母 A、B、C、D、E、F 出现的频率如下：

A—30%，B—25%，C—20% D—10%，E—10%，F—5%

构造一个表示 A、B、C、D、E、F 前缀码，使得传输的二进制位最少。

1. 构造带权
   30,25,20,10,10,5 的最优
   二元树 T;

前缀码构造

Example

已知字母 A、B、C、D、E、F 出现的频率如下：

A—30%，B—25%，C—20% D—10%，E—10%，F—5%

构造一个表示 A、B、C、D、E、F 前缀码，使得传输的二进制位最少。

1. 构造带权 30,25,20,10,10,5 的最优二元树 T;
2. 在 T 上构造前缀码;

前缀码构造

Example

已知字母 A、B、C、D、E、F 出现的频率如下：

A—30%，B—25%，C—20% D—10%，E—10%，F—5%

构造一个表示 A、B、C、D、E、F 前缀码，使得传输的二进制位最少。

1. 构造带权 30,25,20,10,10,5 的最优二元树 T;
2. 在 T 上构造前缀码;
3. 将前缀码对应于字母;

前缀码构造

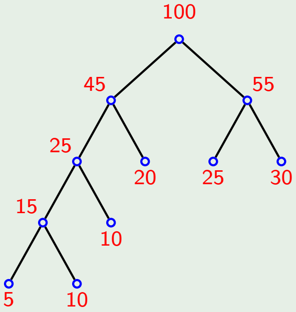
Example

已知字母 A、B、C、D、E、F 出现的频率如下：

A—30%，B—25%，C—20% D—10%，E—10%，F—5%

构造一个表示 A、B、C、D、E、F 前缀码，使得传输的二进制位最少。

1. 构造带权 30,25,20,10,10,5 的最优二元树 T;
2. 在 T 上构造前缀码;
3. 将前缀码对应于字母;



前缀码构造

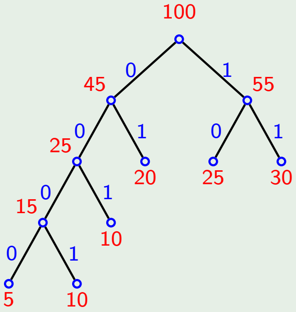
Example

已知字母 A、B、C、D、E、F 出现的频率如下：

A—30%，B—25%，C—20% D—10%，E—10%，F—5%

构造一个表示 A、B、C、D、E、F 前缀码，使得传输的二进制位最少。

1. 构造带权 30,25,20,10,10,5 的最优二元树 T;
2. 在 T 上构造前缀码;
3. 将前缀码对应于字母;



前缀码构造

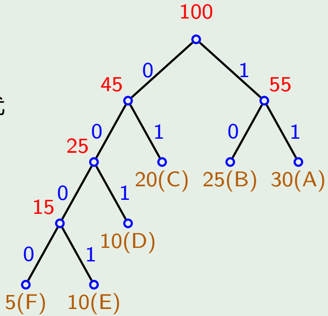
Example

已知字母 A、B、C、D、E、F 出现的频率如下：

A—30%，B—25%，C—20% D—10%，E—10%，F—5%

构造一个表示 A、B、C、D、E、F 前缀码，使得传输的二进制位最少。

1. 构造带权 30,25,20,10,10,5 的最优二元树 T;
2. 在 T 上构造前缀码;
3. 将前缀码对应于字母;



前缀码构造

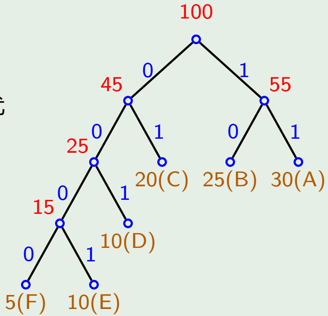
Example

已知字母 A、B、C、D、E、F 出现的频率如下：

A—30%，B—25%，C—20% D—10%，E—10%，F—5%

构造一个表示 A、B、C、D、E、F 前缀码，使得传输的二进制位最少。

1. 构造带权 30,25,20,10,10,5 的最优二元树 T;
2. 在 T 上构造前缀码;
3. 将前缀码对应于字母;



字母	编码
A	11
B	10
C	01
D	001
E	0001
F	0000

决策问题

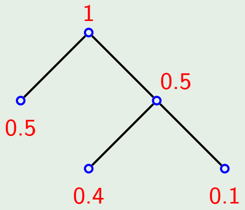
Example

用机器分辨一些币值为 1 分、2 分、5 分的硬币，假设各种硬币出现的概率分别为 0.5、0.4、0.1。问如何设计一个分辨硬币的算法，使所需的时间最少？(假设每作一次判别所用的时间相同，以此为一个时间单位)

决策问题

Example

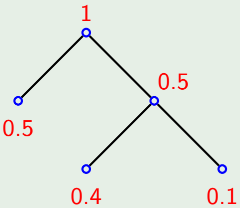
用机器分辨一些币值为 1 分、2 分、5 分的硬币，假设各种硬币出现的概率分别为 0.5、0.4、0.1。问如何设计一个分辨硬币的算法，使所需的时间最少？(假设每作一次判别所用的时间相同，以此为一个时间单位)

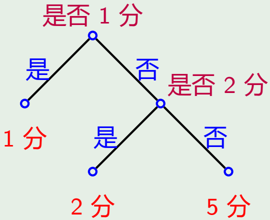


决策问题

Example

用机器分辨一些币值为 1 分、2 分、5 分的硬币，假设各种硬币出现的概率分别为 0.5、0.4、0.1。问如何设计一个分辨硬币的算法，使所需的时间最少？(假设每作一次判别所用的时间相同，以此为一个时间单位)

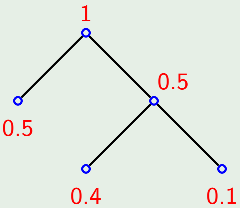


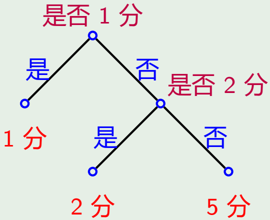


决策问题

Example

用机器分辨一些币值为 1 分、2 分、5 分的硬币，假设各种硬币出现的概率分别为 0.5、0.4、0.1。问如何设计一个分辨硬币的算法，使所需的时间最少？(假设每作一次判别所用的时间相同，以此为一个时间单位)





所需时间：$2 \times 0.1 + 2 \times 0.4 + 1 \times 0.5 = 1.5$(时间单位)。

THE END, THANKS!