

# [12] 发明专利申请公开说明书

[21] 申请号 98124330.4

[43]公开日 2000年4月5日

[11]公开号 CN 1249473A

[22]申请日 1998.9.30 [21]申请号 98124330.4

[71]申请人 朗讯科技公司

地址 美国新泽西

[72]发明人 赵风光(音译)

[74]专利代理机构 中国国际贸易促进委员会专利商标事

务所

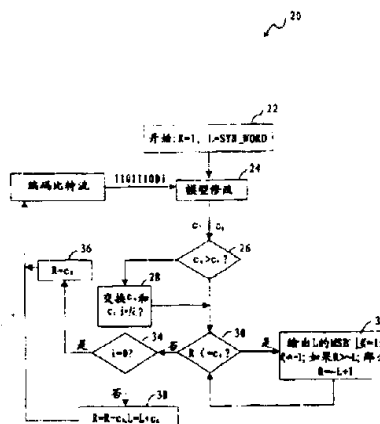
代理人 于静

权利要求书 9 页 说明书 7 页 附图页数 3 页

[54]发明名称 无乘法的算术编码

[57]摘要

用于执行编码和解码的一种无乘法算术编码技术。该编码技术包括从编码的流中接收码元和两个频率计数,找到一个最可能的码元和一个最不可能的码元,使第一寄存器进行数值移位操作,用于输出比特到编码的比特流并且用于近似该编码的字符串中的每个码元的上下文概率,根据该上下文概率编码在该编码字符串中的下一个码元。



ISSN 1008-4274

## 权 利 要 求 书

---

1、一种产生编码的比特流的无乘法算术编码方法，包括以下步骤：  
初始化第一寄存器为 1，并且使第二寄存器是一个任意数；

对于一个输入字符串中的每个码元，重复执行下述步骤 A-E：

A · 接收代表概率为 0 的一个小数值的第一个频率计数，代表概率为 1 的一个小数值的第二个频率计数，以及该码元；

B · 如果第一个频率计数大于第二个频率计数，那么，交换第一个频率计数的值和第二个频率计数的值并且设置该编码的码元的值为它的逻辑补码；

C · 当第一寄存器的值小于或等于第一个频率计数时，重复执行下述步骤 C<sub>1</sub>-C<sub>3</sub>：

C<sub>1</sub>、输出第二寄存器的最高有效位，

C<sub>2</sub>、左移第一寄存器的值和第二寄存器的值，并且

C<sub>3</sub>、如果第一寄存器的值大于第二寄存器的值的二进制补码，那么，设置第一寄存器值为第二寄存器的值的二进制补码加 1；

D · 如果该码元等于零，则设置第一寄存器的值为第一个频率计数，否则从第一寄存器的值中减去第一个频率计数并将第一个频率计数加到第二寄存器的值中；和

E · 选择输入字符串中的下一个码元；并且  
输出第二寄存器的值作为编码的比特流。

2、一种用于执行无乘法算术编码以便产生一个编码的比特流的处理器，包括：

一个初始化处理单元，用于初始化第一寄存器为 1，并使第二寄存器为一个任意数；

一个用于执行于下述操作的处理单元，对于输入字符串中的每个码元，重复执行下列操作 A-E：

A · 接收代表概率为 0 的一个小数值的第一个频率计数，代表概率为

1 的一个小数值的第二频率计数, 以及该码元;

B · 如果第一频率计数大于第二频率计数, 那么, 交换第一频率计数的值和第二频率计数的值并且设置该编码的码元的值为它的逻辑补码;

C · 当第一寄存器的值小于或等于第一频率计数时, 重复执行下述步骤 C<sub>1</sub>-C<sub>3</sub>:

C<sub>1</sub>、输出第二寄存器的最高有效位,

C<sub>2</sub>、左移第一寄存器的值和第二寄存器的值, 并且

C<sub>3</sub>、如果第一寄存器的值大于第二寄存器的值的二进制补码, 那么, 设置第一寄存器的值为第二寄存器的值的二进制补码加 1;

D · 如果该码元等于零, 则设置第一寄存器的值为第一频率计数, 否则从第一寄存器的值中减去第一频率计数并将第一频率计数加到第二寄存器的值中; 和

E · 选择输入字符串中的下一个码元; 和

用于输出第二寄存器的值作为编码的比特流的一个输出处理单元。

3、一种实现在计算机可读的介质上用于执行无乘法算术编码以产生一个编码的比特流的计算机程序, 包括:

一个初始化源代码段, 用于初始化第一寄存器为 1 并且使第二寄存器的为一个任意数;

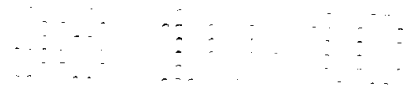
一个处理源代码段, 用于执行下述操作, 对于输入字符串中的每个码元, 重复执行下述操作 A-E:

A · 接收代表概率为 0 的一个小数值的的第一频率计数, 代表概率为 1 的一个小数值的第二频率计数, 以及该码元;

B · 如果第一频率计数大于第二频率计数, 那么交换第一频率计数的值和第二频率计数的值并且设置该编码的码元的值为它的逻辑补码;

C · 当第一寄存器的值小于或等于第一频率计数时, 重复执行下述步骤 C<sub>1</sub>-C<sub>3</sub>:

C<sub>1</sub>、输出第二寄存器的最高有效位,



C2、左移第一寄存器的值和第二寄存器的值，并且

C3、如果第一寄存器的值大于第二寄存器的值的二进制补码，那么设置第一寄存器的值为第二寄存器的二进制补码加1；

D·如果该码元等于零，则设置第一寄存器的值为第一频率计数，否则从第一寄存器的值中减去第一频率计数并将第一频率计数加到第二寄存器的值中；和

E·选择输入字符串中的下一个码元；和

一个输出源代码段，用于输出第二寄存器的值作为编码的比特流。

4、一种用于执行无乘法算术编码，以产生一个编码的比特流的计算机信号，包括：

一个初始化信号段，用于初始化第一寄存器为1并且使第二寄存器为一个任意数；

一个处理信号段，用于执行下列操作，对输入字符串中的每个码元，重复执行下列操作A-E：

A·接收代表概率为0的一个小数值的第二频率计数，代表概率为1的一个小数值的第二频率计数，以及该码元；

B·如果第一频率计数大于第二频率计数，那么交换第一频率计数的值和第二频率计数的值并且设置该编码的码元的值为它的逻辑补码；

C·当第一寄存器的值小于或等于第一频率计数时，重复执行下述步骤C<sub>1</sub>-C<sub>3</sub>：

C<sub>1</sub>、输出第二寄存器的最高有效位，

C<sub>2</sub>、左移第一寄存器的值和第二寄存器的值，并且

C<sub>3</sub>、如果第一寄存器的值大于第二寄存器的值的二进制补码，那么设置第一寄存器的值为第二寄存器的二进制补码加1；

D·如果该码元等于零，则设置第一寄存器的值为第一频率计数，否则从第一寄存器中减去第一频率计数并将第一频率计数加到第二寄存器的值中；和

E·选择输入字符串中的下一个码元；和

一个输出信号段，用于输出第二寄存器的值作为编码的比特流。

5、权利要求4的计算机信号，其中所述的计算机信号实现在一个载波上：

6、一种无乘法算术编码以产生一个编码的比特流的方法，包括步骤：

从一个编码的字符串中接收一个码元和两个频率计数；

找到一个最可能的码元和一个最不可能的码元；

使第一寄存器进行数值移位操作，用于输出比特到编码的比特流并且用于近似该编码的字符串中的每个码元的上上文概率；和

根据上下文概率，编码在编码字符串中的下一个码元。

7、权利要求6的方法，其中不需要数值或比特填充步骤。

8、权利要求6的方法，其中所述的编码步骤在所述的进行步骤之后执行。

9、一种用于执行无乘法算术编码以产生一个编码的比特流的处理器，包括：

一个接收单元，用于从一个编码的字符串中接收一个码元和两个频率计数；和一个处理单元，用于找出一个最可能的码元和一个最不可能的码元，使第一寄存器进行数值移位操作，用于输出比特到编码的比特流和用于近似编码字符串中的每个码元的上下文概率并且根据该上下文概率编码在编码的字符串中的下一个码元。

10、权利要求9的处理器，其中不需要数值或比特填充操作。

11、权利要求9的处理器，其中在数值移位操作以后所述处理器执行编码。

12、一种实现在计算机可读的介质上，用于执行无乘法算术编码以产生一个编码的比特流的计算机程序，包括：

一个接收源程序代码段，用于从一个编码的字符串中接收一个码元和两个频率计数；和

一个处理源程序代码段，用于找出一个最可能的码元和一个最不可能的码元，使第一寄存器进行数值移位操作，用于输出比特到编码的比

特流并且用于近似在编码的字符串中的每个码元的上下文概率并且根据该上下文概率编码在编码的字符串中的下一个码元。

13、权利要求 12 的计算机程序，其中不需要数值或比特填充。

14、权利要求 12 的计算机程序，其中所述处理源程序代码段在数值移位操作以后执行编码。

15、一种用于执行无乘法算术编码以产生一个编码的比特流的计算机信号，包括：

一个接收信号段，用于从一个编码的字符串中接收一个码元和两个频率计数；和

一个处理信号段，用于找出一个最可能的码元和一个最不可能的码元，使第一寄存器进行数值移位操作，用于输出比特到编码的比特流并且用于近似在编码的字符串中的每个码元的上下文概率并且根据该上下文概率编码在编码的字符串中的下一个码元。

16、权利要求 15 的计算机信号，其中不需要数值或比特填充信号段。

17、权利要求 15 的计算机信号，其中所述处理源代码段在数值移位操作以后执行编码。

18、权利要求 15 的计算机信号，所述的计算机信号实现在一个载波上。

19、一种产生解码的字符串的无乘法算术编码的方法，包括步骤：

初始化第一寄存器为 1，并且使第二寄存器是一个任意数；

接收代表概率为 0 的一个小数值的的第一频率计数，代表概率为 1 的一个小数值的第二频率计数，以及一个输出比特；

如果第一频率计数大于第二频率计数，那么，交换第一频率计数和第二频率计数的值并用设置输出比特的值为 1，否则设置输出比特的值为 0；

当第一寄存器的值小于或等于第一频率计数时，重复执行下述步骤 A-C：

A·左移第一寄存器的值，第二寄存器的值，和包括解码字符串的

第三寄存器的值,

B · 提供来自比特流的下一比特到第三寄存器, 和

C · 如果第一寄存器的值大于第二寄存器的值的二进制补码, 那么设置第一寄存器的值为第二寄存器的二进制补码加 1; 和

如果第一频率计数小于第三寄存器的值, 则设置第一寄存器的值为第一频率计数并且输出该输出比特作为解码字符串的下一比特, 否则从第一寄存器的值中减去第一频率计数并且将第一频率计数加到第二寄存器的值并且输出该输出比特的逻辑补码作为解码字符串的下一比特。

20、一种用于执行无乘法算术编码以便产生一个解码的字符串的处理器, 包括:

一个初始化处理单元, 用于初始化第一寄存器为 1 并且使第二寄存器为一个任意数;

一个输入处理单元, 用于接收表示概率为 0 的一个小数值的频率计数, 表示概率为 1 的一个小数值的第二频率计数, 和一个输出比特; 和

一个处理单元, 用于执行下列操作, 如果第一频率计数大于第二频率计数, 那么交换第一频率计数的值和第二频率计数的值并且设置输出比特的值为 1, 否则设置输出比特的值为 0;

当第一寄存器的值小于或等于第一频率计数时, 重复执行下述步骤 A-C:

A · 左移第一寄存器的值, 第二寄存器的值, 和包括解码字符串的三寄存器的值,

B · 提供来自比特流的下一比特到第三寄存器, 和

C · 如果第一寄存器的值大于第二寄存器的值的二进制补码, 那么设置第一寄存器的值为第二寄存器的二进制补码加 1;

如果第一频率计数小于第三寄存器的值, 则设置第一寄存器的值为第一频率计数并且输出该输出比特作为解码字符串的下一比特, 否则从第一寄存器的值中减去第一频率计数并且将第一频率计数加到第二寄



寄存器的值并且输出该输出比特的逻辑补码作为解码字符串的下一比特。

21、一种实现在计算机可读的介质上用于执行无乘法算术编码以产生一个解码的字符串的计算机程序，包括

一个初始化源代码段，用于初始化第一寄存器为 1 并且使第二寄存器为一个任意数；

一个输入源代码段，用于接收表示概率为 0 的一个小数值的第一个频率计数，表示概率为 1 的一个小数值的第二个频率计数，和一个输出比特；和

一个处理源代码段，用于执行下列操作，

如果第一频率计数大于第二频率计数，那么交换第一频率计数值和第二频率计数的值并且设置输出比特的值为 1，否则设置输出比特的值为 0；

当第一寄存器的值小于或等于第一频率计数时，重复执行下述步骤 A-C：

A · 左移第一寄存器的值，第二寄存器的值，和包括解码字符串的三寄存器的值，

B · 提供来自比特流的下一比特到第三寄存器，和

C · 如果第一寄存器的值大于第二寄存器的值的二进制补码，那么设置第一寄存器的值为第二寄存器的二进制补码加 1；和

如果第一频率计数小于第三寄存器的值，则设置第一寄存器的值为第一频率计数并且输出该输出比特作为解码字符串的下一比特，否则从第一寄存器的值中减去第一频率计数并且将第一频率计数加到第二寄存器的值并且输出该输出比特的逻辑补码作为解码字符串的下一比特。

22、一种用于执行无乘法算术编码以产生一个解码的字符串的计算机信号，包括：

一个初始化信号段，用于初始化第一寄存器为 1 并且使第二寄存器为一个任意数；



一个接收信号段，用于接收表示概率为 0 的一个小数值的频率计数，表示概率为 1 的一个小数值的频率计数，和一个输出比特；和

一个处理信号段，用于执行下列操作，

如果第一频率计数大于第二频率计数，那么，交换第一频率计数的值和第二频率计数的值并用设置输出比特的值为 1，否则设置输出比特的值为 0；

当第一寄存器的值小于或等于第一频率计数时，重复执行下述步骤 A-C：

A 左移第一寄存器的值，第二寄存器的值，和包括解码字符串的三寄存器的值，

B 提供来自比特流的下一比特到第三寄存器，和

C 如果第一寄存器的值大于第二寄存器的值的二进制补码，那么设置第一寄存器的值为第二寄存器的二进制补码加 1；并且

如果第一频率计数小于第三寄存器的值，则设置第一寄存器的值为第一频率计数并且输出该输出比特作为解码字符串的下一比特，否则从第一寄存器的值中减去第一频率计数并且将第一频率计数加到第二寄存器的值并且输出该输出比特的逻辑补码作为解码字符串的下一比特。

23、权利要求 22 的计算机信号，其中所述的计算机信号实现在一个载波上。

24、一种无乘法算术编码以产生解码的字符串的方法，包括步骤：从一个解码的字符串中接收一个码元和两个频率计数；

找到一个最可能的码元和一个最不可能的码元；

使第一寄存器进行数值移位操作，用于输出比特到解码的比特流并且用于近似该解码的字符串中的每个码元的上下文概率；和

根据上下文概率，解码在解码字符串中的下一个码元。

25、一种用于执行无乘法算术编码以产生一个解码的字符串的处理器，包括：

一个接收单元，用于从一个编码的字符串中接收一个码元和两个频率计数；和

一个处理单元，用于找出一个最可能的码元和一个最不可能的码元，使第一寄存器进行数值移位操作，用于输出比特到编码的比特流并且用于近似在编码的字符串中的每个码元的上下文概率并且根据该上下文概率编码的字符串中的下一个码元。

26、一个实现在计算机可读者的介质上，用于执行无乘法算术编码以产生一个解码的字符串的计算机程序，包括：

一个接收源代码段，用于从一个编码的字符串中接收一个码元和两个频率计数；和

一个处理源代码段，用于找出一个最可能的码元和一个最不可能的码元，使第一寄存器进行数值移位操作，用于输出比特到编码的比特流并且用于近似在编码的字符串中的每个码元的上下文概率并且根据该上下文概率编码在编码的字符串中的下一个码元。

27、一种用于执行无乘法算信编码，以产生一个解码的字符串的计算机信号，包括：

一个接收信号段，用于从一个编码的字符串中接收一个码元和两个频率计数；和

一个处理信号段，用于找出一个最可能的码元和一个最好不可能的码元，使第一寄存器进行数值移位操作，用于输出比特到编码的比特流并且用于近似在编码的字符串中的每个码元的上下文概率并且根据该上下文概率编码在编码的字符串中的下一个码元。

28、权利要求 27 的计算机信号，其中所述的计算机信号实现在一个载波上。

# 说 明 书

---

## 无乘法的算术编码

本发明涉及一种算术编码技术，并具体涉及具有低的计算复杂度和低成本的硬件配置的一种算术编码技术。

经常使用将数据压缩为较少比特，以便实现期望的传输数据速率或者在有限的存储器空间存储数据。数据压缩后的一段时间，通过解压原始压缩的数据恢复原始的数据。

算术编码是实现数据压缩和解压的一种技术。在算术编码中，编码另一次判定之后的一个判定，以便确定沿着编号行的连续较小的，较少包括的间隔。算术编码提供每个判定具有多个可能的排它的结果或事件。每个输出或事件由一个码元表示。

根据现有技术算术编码技术，概率行具有沿所定义的一个当前间隔。第一当前间隔是 0 到 1。当前间隔被分成段，其中每段对应下一个判定的一个可能的结果。此外对于每个判定只有两个可能的结果，当前的间隔被分为两段。每一段的长度是基于它的各自相关的概率。各自的概率可以保持固定或者适应输入的判定数据。

算术编码的效力在于它们执行的压缩并且算术编码的灵活性在于它们同样容易地编码由静态和非静态源建模的字符串的能力。算术编码允许在无需字母扩展的情况下编码二进制字符串，并且还利用从包含多于两个字符的字母表中抽出的码元编码字符串。

算术编码通过乘法  $P(S) * P(i/s)$  修改一个目前处理的源字符串  $S$  的概率  $P(s)$ ，此处  $P(i/s)$  是码元  $i$  给出  $S$  的条件概率。增加概率所需的乘法相对昂贵并且慢，即使在概率由最多具有固定个数的有效数字的二进制数表示的情况下。

美国专利 4467317 实现了递归技术，它通过利用二分之一整数次方，近似最不可能信号 (LPS) 的概率，简化了二进制码元流的编码



操作。但是，这种技术不能容易地推广到非二进制的字母表，因为  $n$  进制字母表码元概率不能准确地近似为二分之一幂次方。这种限制就当今处理器在数据结构和操作的并行增长而言是严重的。另外，递归技术依赖能够计算和提供源统计特性的复杂的模型单元的存在。源统计特性的计算的绝对是对模型单元的偏离数计算操作的缺陷，这种操作增加了整体数据压缩问题的复杂性。

美国专利 4652856 公开了一种算术编码技术，它能够编码多个字符字母表，没有乘法或除法以便产生一个二进制代码流，响应简化的数据源统计特性，该特性保持代码流和内部代码变量二者同时修改的非常希望的特性。美国专利 4652856 是根据以二进制形式接受码元出现计算的编码算法，使那些计算进行简单的数值移动操作以便近似码元流中的每个码元的上下文概率，增加编码的流，并且然后同时定位代码流的  $W$  个最低有效位并调整下一个编码子间隔为内部编码变量行列式，响应一个移动的出现计数。

算术编码是一种统计压缩方法并且因此利用估计模型源中的每个可能码元的概率的一种模型。该模型可以是固定的，半固定的或自适应的。固定的模型具有分配概率  $P_1, P_2, \dots, P_n$ 、此处  $1, 2, \dots, n$  是模型源中的码元。

算术编码根据间隔变换工作，它将一个间隔  $[l_k, h_k]$  变换为另一个间隔  $[l_{k+1}, h_{k+1}]$ ，假设码元  $i$  被编码，其中

$$l_{k+1} = l_k + r_k \sum_{j=0}^{i-1} P_j \quad (1)$$

$$h_{k+1} = l_k + r_k \sum_{j=0}^i P_j \quad (2)$$

$$r_{k+1} = h_{k+1} - l_{k+1} \quad (3)$$

$r_k$  表示间隔  $[l_k, h_k]$  的范围。初始，编码器以单位间隔  $[0, 1)$

开始。这意味  $l_0=0$ ,  $h_0=1$ ,  $r_0=1$ 。

算术编码不能在有限精度的计算机上实现,因为在等式(1)-(3)中涉及的所有操作是浮点运算。然而,结果是使用标准16位和32位整数操作可以最好地实现算术编码,浮点运算既不需要也没有帮助。所需要的是增量传输方案。

作为一个简单的例子,考虑  $n=2$  的情况,即采用0序 Markov 模型的二进制源以便估计  $P_0$  和  $P_1$  的概率。在这个例子中,等式(1),(2)和(3)可以简化为

$$r_{k+1}=r_k P_0, \text{ 对于 } i=0 \quad (4)$$

$$l_{k+1}=l_k+r_k P_0, \quad r_{k+1}=r_k P_1, \text{ 对于 } i=1 \quad (5)$$

其中  $i$  是目前编码的码元。

关于等式(4)和(5)的实际实现,以频率计数  $C_0$  和  $C_1$  替换  $P_0$  和  $P_1$ ,  $C_0$  和  $C_1$  可以在编码过程中自适应调节。

$$P_0=C_0/(C_0+C_1), \quad P_1=C_1/(C_0+C_1) \quad (6)$$

在算术编码的常规设计中,编码器使用大小为16比特的两个寄存器,  $L$  和  $R$ , 以便分别存储  $l_k$  和  $r_k$  的小数值。组合(4),(5)和(6)产生

$$L := L, \quad R := RC_0/(C_0+C_1), \quad i=0 \quad (7)$$

$$L := L+RC_0/(C_0+C_1), \quad R := RC_1/(C_0+C_1), \quad i=1 \quad (8)$$

此处涉及  $m$  比特整数操作,并且使用  $m$  比特整数舍入成整数(此处  $m$  是一个典型的硬件寄存器大小,例如8,16或是32)。现在可用的大多数算法是根据等式(7)和(8)或它们的变化。注意等式(7)和(8)包括整数乘法和除法,并且因此不适合于硬件设计,例如数字信号处理器(DSP)和灵巧卡实现,此处没有除法指令是可用的。

本发明提供用于比特级算术编码的一种算术编码实现。本发明执行算术编码,无需乘法和除法。特别是,本发明执行的编码和解码接收频率计数,也就是,表示概率为0的小数值的一个频率计数,表示概率为1的小数值的一个频率计数和目前编码的码元,以便编/解码随后的

码元。

本发明还消除了对常规进位技术例如比特填充的需求，幅值确定式是不必要的并且在编码比特之前调整寄存器的值，而在需要幅值确定的常规设计中，在调整寄存器的值之前编码该比特。

在本发明中，两个寄存器的初始值被置为与  $2^{m-1}$  或  $2^m-1$  和 0 对照的 1 和随机数。因此，输出流中的第一字可以用于指定实时传输应用的同步字。

本发明还提供了一种低计算复杂度和低成本硬件配置，然而仍然获得了与通过基于乘法技术获得的压缩比率可比的压缩比率。

图 1 ( a ) 和图 1 ( b ) 说明在一个实施例中的本发明的方框图；

图 2 说明在本发明的一个实施例中编码的流程图；和

图 3 说明在本发明的一个实施例中解码的流程图。

本发明将参照图 1-图 3 全面描述。图 1 ( a ) 说明在本发明的一个实施例中的硬件图。通过图 1 ( a ) 所示的系统 1 执行无乘法的算术编码。系统 1 至少包括一个处理器 10 和一个制造项目 12。制成品 12 还包括一个存储介质 14 和一个可执行的计算机程序 16。可执行的计算机程序 16 包括执行无乘法的算术编码的指令。在另一个实施例中，如图 1 ( b ) 所示，提供计算机可执行的程序 16 作为外部提供的传播信号 18。

### 编码

首先，将叙述编码操作。初始地，两个寄存器 R 和 L 分别置为 1 和一个任意数。处理器 10 具有三个输入，表示概率为 0 的一个小数值的第一个频率计数  $C_0$ ，表示概率为 1 的一个小数值的第二个频率计数  $C_1$  和一个目前的编码码元 i ( 可以是 0 或 1 )。

由处理器 10 执行的编码步骤可以总结为伪码如下：

1 If  $C_0 > C_1$ ，则交换  $C_0$  和  $C_1$  的值，并且令  $i = !i$ 。

2 While  $R \leq C_1$  时， do。

输出最高有效位 L

$L = L \ll 1$ ，  $R = R \ll 1$

If  $R > \sim L$ ， then  $R = \sim L + 1$

3. If  $i=0$ , then  $R=C_0$ ; else  $R=R-C_0$ ,  $L=L+C_0$ . 输出 L

注意到在上面的伪码中采用某些 C 语言注释。! 表法逻辑补码, ~ 表示二进制补码, 并且  $\ll$  表示算术左移。从上面的描述中, 本发明工作在下面假设: 对于每次迭代,

$R \approx C_0+C_1$ ; 于是等式 (7) 和 (8) 被简化为:

$L:=L, R:=C_0, i=0$  (9)

$L:=L+C_0, R:=R-C_0, i=1$  (10)

在本发明中, 分别初始化两个寄存器 R 和 L 为 1 和一个任意数, 让输出流中的第一字指定用于实时传输应用的一个同步字。另外, 步骤 1 一般称作一个交换步骤, 步骤 2 称调整步骤, 而步骤 3 称作编码步骤。在常规的无乘法算术编码技术中需要的量值步骤在本发明中不需要。在本发明中, 调整步骤在编码步骤前执行。在调整步骤中, 当寄存器 R 的值小于或等于第二频率计数的值, 执行“While”循环, 并且如果寄存器 R 的值大于寄存器 R 的值的二进制补码设置寄存器 R 的值等于寄存器 L 的值的二进制补码加 1, 这样就不需要随后的比特填充步骤。

总而言之, 本发明的无乘法算术编码方法通过从一个编码字符串和两个频率计数中接收一个码元, 找到一个最可能的码元和一个最不可能的码元; 第一寄存器须经幅值移位操作, 用于输出比特到编码的比特流并且用于近似在编码的字符串中的每个码元的一个上下文概率, 并且根据该上下文概率在该编码的字符串中编码下一个码元。

图 2 包括在编码处理 20 中由处理器 10 执行的详细步骤。特别是, 在步骤 22, 分别初始化寄存器 R 和 L 为 1 和同步字。在这个例子中, 在步骤 24 编码的比特流, 11011100i 与寄存器 R 和 L 的初始值一起输入到 0 阶 Markov 模型以便产生频率计数  $C_0$  和  $C_1$ 。在步骤 26, 比较  $C_0$  和  $C_1$  并且如果  $C_0$  大于  $C_1$ , 则交换  $C_0$  和  $C_1$  并且在步骤 28 设置 i 为它的逻辑补码。但是, 如果  $C_0$  不大于  $C_1$ , 则处理进入步骤 30, 此处确定寄存器 R 中的值是否大于或等于  $C_1$ 。如果是大于等于  $C_1$ , 则处理进入步骤 32, 此处输出 L 寄存器的最高有效位, L 和 R 被算术左移, 并且如果 R 大于 L 的二进制补码, 那么 R 被置为 L 的二进制补码加 1,

并且处理返回步骤 30。如果寄存器 R 的值不大于等于  $C_1$ ，那么处理继续到步骤 34。在步骤 34，确定 i 是否等于 0。如果 i 等于 0，那么在步骤 36 寄存器 R 的值被置为等于  $C_0$  并且如果 i 不等于 0，那么在步骤 38 寄存器 R 被置为 R 的以前值减  $C_0$  并且 L 被置为 L 的以前值加  $C_0$ ，因此编码比特流中的下一个比特。然后通过步骤 24 输入下一比特到 Markov 模型修改，重复该处理。继续该处理直到输入比特流的所有比特被编码。然后，输出寄存器 L 的值作为编码的比特流。

尽管使用零阶模型描述了本发明，但是对本领域的普通技术人公知的任意一种模型都可以使用。

### 解码

为解码，再次使 R 和 L 寄存器初始化并且使用第三寄存器 V 以便存储解码比特流的部分，并且 i 指示输出比特。如果 S 是解码比特流，由上述的编码算法产生，则由处理器 10 执行的解码步骤以伪码归纳如下：

- 1 · If  $C_0 > C_1$ ，交换  $C_0$  和  $C_1$  的值并且令  $i=1$ ； else  $i=0$ 。
- 2 · While  $R \leq C_1$ ， do
  - L=L << 1， R=R << 1， V=V << 1
  - V=V|来自 S 的下一比特
  - If  $R > \sim L$ ， then  $R=\sim L+1$
- 3 · If  $C_0 < V$ ， then  $R=C_0$ ； else  $R=R-C_0$ ，  $L=L+C_0$ ， and  $i=!i$

总之，产生解码的字符串的无乘法算术编码方法从解码的流和两个频率计数中接收比特，找出一个最可能的码元和一个最不可能的码元，使第一寄存器经受幅值移位操作，用于从解码的比特流中输入比特并且用于近似在解码的字符串中的每个码元的上下文概率并且根据该上下文概率解码下一个码元到解码的流中。

图 3 包括由解码处理 40 中的处理器 10 执行的具体步骤。特别是，在步骤 42，初始化寄存器 R， L 和 V。在步骤 44，输入寄存器 R， L 和 V 的值和解码的字符串到 0-Markov 模型以便产生频率计数  $C_0$  和  $C_1$ 。在步骤 46，比较  $C_0$  和  $C_1$  并且如果  $C_0$  大于  $C_1$ ，则在步骤 48 交换  $C_0$  和  $C_1$  并且设置 i 为它的逻辑补码。但是，如果  $C_0$  不大于  $C_1$ ，则处



理进入步骤 50，此处确定寄存器 R 的值是否大于或等于  $C_1$ 。如果是，处理进入步骤 52，此处将寄存器 R，L 和 V 都算术左移，来自解码比特流 S 的下个比特加到寄存器 V，并且如果 R 大于 L 的二进制补码，则 R 被置为 L 的二进制补码加 1。然后处理返回步骤 50。

如果寄存器 R 的值不大于或等于  $C_1$ ，由处理继续到步骤 54。在步骤 54，确定  $C_0$  是否小于 V。如果  $C_0$  小于 V，那么在步骤 56 设置寄存器 R 的值等于  $C_0$  并且如果  $C_0$  不小于 V，那么，在步骤 58 寄存器 R 被置为 R 的以前值减  $C_0$ ，L 被置为 L 的以前值加  $C_0$ ，并且 i 被置为它的逻辑补码，因此解码比特流 S 中的下一比特。然后通过步骤 44 输入下一比特到 Markov 模型修改，处理重复。继续该处理，直到解码该解码的比特流 S 所有比特。

再次，尽管只使用 0 阶 Markov 模型，但是对于本领域普通技术人员公知的任意模型，都可以使用。

下面阐述的表 1，说明了在执行乘法的一个编码器和在美国专利 4652856 中公开的现有技术和本发明的无乘法算术编码之间的用于各种类型文件的压缩比率比较。

如表 1 所示的，本发明实现了优于现有技术各种类型压缩比率的无乘法算术技术。表 1 还说明由于每个无乘法设计使用一些近似值，而不是实际可能值，所以乘法编码器通常提供最好的压缩，所以在使用无乘法算术技术的压缩比率上通常会有某些退化。但是，如表 1 所示的本发明，提供低的计算复杂度和低的成本硬件配置，还实现了可与基于乘法的技术相比的压缩比率。

表 1

源文件	大小	乘法编码器	本发明	美国专利 4,652,856
C 源程序	27620	37.5%	38.4%	39.9%
中文文件	72596	43.3%	43.8%	44.9%
标度图像	262330	67.9%	68.8%	69.6%
执行文件	54645	74.3%	74.6%	75.6%
混和数据	417192	67.2%	68.0%	68.9%

# 说明书附图

图1(a)

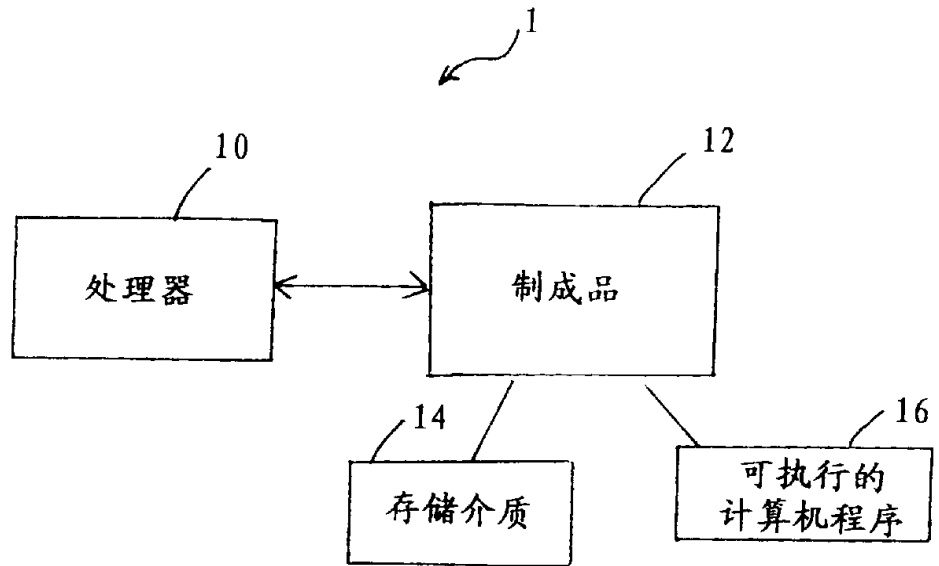


图1(b)

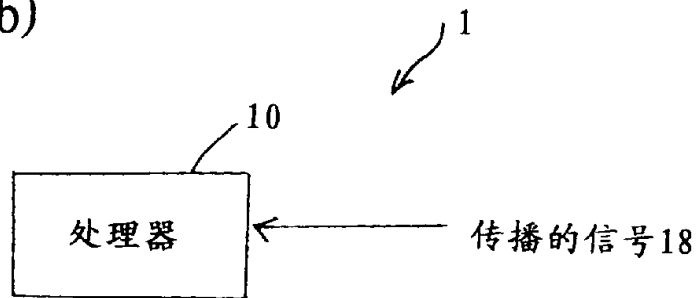


图 2

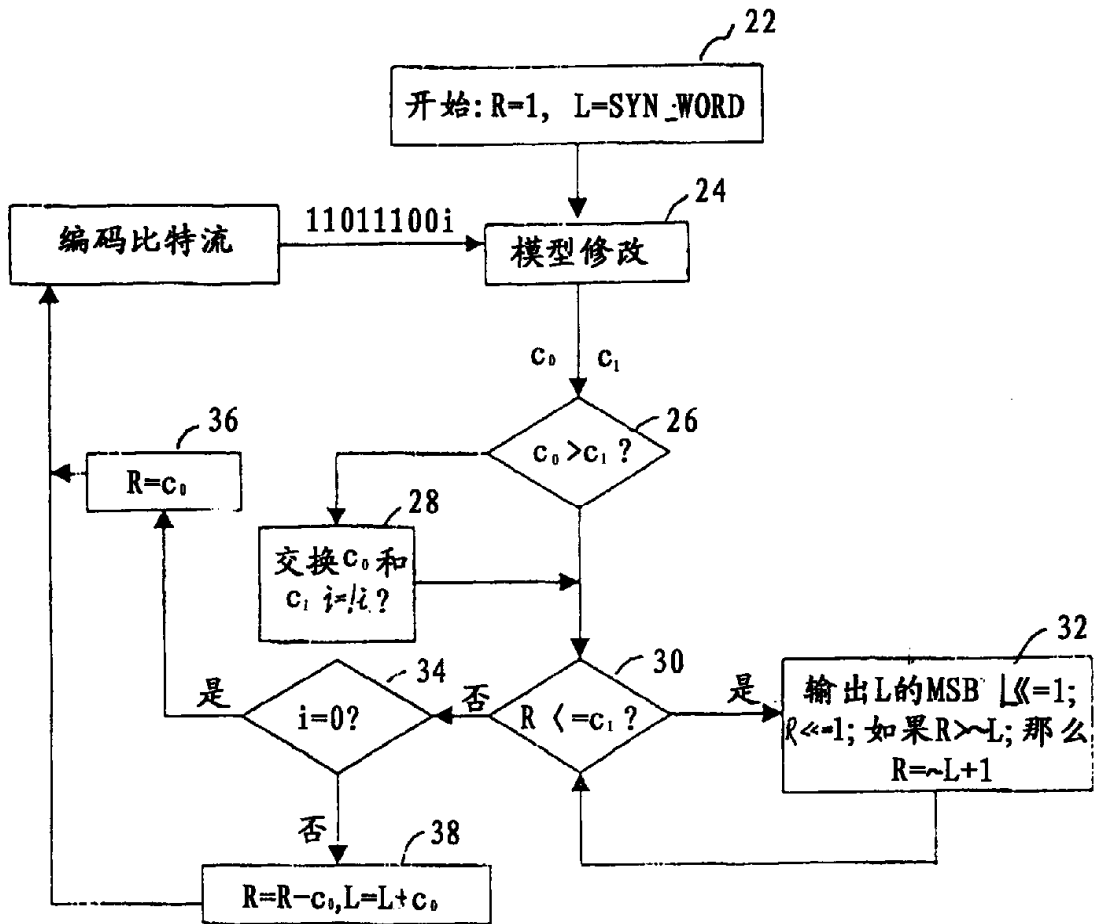


图 3

