



## [12] 发明专利申请公布说明书

[21] 申请号 200780013972.2

[43] 公开日 2009年5月6日

[11] 公开号 CN 101427495A

[22] 申请日 2007.2.16

[21] 申请号 200780013972.2

[30] 优先权

[32] 2006. 2. 21 [33] US [31] 60/775,528

[32] 2007. 2. 13 [33] US [31] 11/674,655

[86] 国际申请 PCT/US2007/062302 2007.2.16

[87] 国际公布 WO2007/098397 英 2007.8.30

[85] 进入国家阶段日期 2008.10.20

[71] 申请人 数字方敦股份有限公司

地址 美国加利福尼亚州

[72] 发明人 M·A·肖克洛拉希 M·G·卢比  
M·沃森 L·明德

[74] 专利代理机构 上海专利商标事务所有限公司

代理人 陈 炜

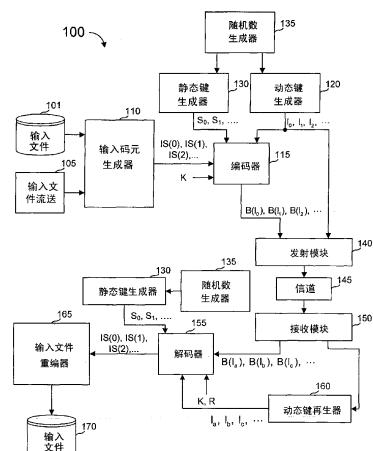
权利要求书4页 说明书72页 附图27页

## [54] 发明名称

用于通信系统的基于多域的码生成器的解码器

## [57] 摘要

提供了一种用于编码数据以便在通信信道上从源向目的地传输的方法。该方法对输入码元的有序集操作并且包括基于线性约束从输入码元生成多个冗余码元。该方法还包括基于线性组合从包括输入码元和冗余码元的码元组合集生成多个输出码元，其中该线性约束或组合中的至少一个是在第一有限域上而该线性约束或组合中的至少另一个是在一不同的第二有限域上，并使得可从任意预定数目的输出码元再生输入码元的有序集至所需的精确度。



1. 一种用于编码数据以便在预期至少部分地如删除信道一样执行的通信信道上从源传输至目的地的方法，所述方法包括：

获得表示所述要被编码的数据的输入码元的有序集；

选择多个域数组值，其中每个域数组是从有限域数组导出的并且至少两个不同的有限域数组被表示；

生成表示系数矩阵——其表示所述多个域数组中的至少两个——的数据结构，其中这些域数组中的至少两个是从彼此不同的有限域数组导出的；

生成作为输入码元的线性组合的输出码元，其中特定组合是依据于表示所述系数矩阵的所述数据结构；以及

使用所生成的输出码元并对所述数据编码。

2. 如权利要求 1 所述的方法，其特征在于，所述表示系数矩阵的数据结构是单元值的二维数组，每个单元值表示在一个输出码元的生成中一个输入码元的系数以使得当系数为非零或以某一基数为模取余不为零时，该相应输出码元的值依赖于该相应输入码元的值。

3. 如权利要求 1 所述的方法，其特征在于，所述表示系数矩阵的数据结构是指定系数值的规则集，且其中规则指示系数不为零或以某一基数为模取余不为零，则相应输出码元的值依赖于该相应输入码元的值。

4. 如权利要求 1 所述的方法，其特征在于，针对所述输入码元的固定值的任意集合能够从所述输入码元集合生成的唯一性输出码元的数目独立于所述域数组大小。

5. 如权利要求 1 所述的方法，其特征在于，表示系数矩阵——其表示所述多个域数组中的至少两个——的数据结构的所述生成是使用从第一有限域数组导出的第一域数组和从第二有限域数组导出的第二域数组的生成，其中这些域数组中的至少两个是从彼此不同的有限域数组导出的，其中所述第一有限域数组和所述第二有限域数组是不同的，且此外所述第一有限域和所述第二有限域各自是从包含 GF(2)、GF(4)、GF(16)、GF(256)的域集合中选出的。

6. 如权利要求 5 所述的方法，其特征在于，所述第一有限域数组是 GF(2)

而所述第二有限域数组为 GF(256)。

7. 如权利要求 5 所述的方法，其特征在于，所述第一有限域数组是 GF(2)而所述第二有限域数组为 GF(4)。

8. 如权利要求 5 所述的方法，其特征在于，所述第一有限域数组是 GF(4)而所述第二有限域数组为 GF(16)。

9. 如权利要求 5 所述的方法，其特征在于，所述第一有限域数组是 GF(16)而所述第二有限域数组为 GF(256)。

10. 如权利要求 5 所述的方法，其特征在于，所述第一有限域数组小于所述第二有限域数组。

11. 如权利要求 5 所述的方法，其特征在于，所述第一有限域数组大于所述第二有限域数组。

12. 一种用于将在目的地通过预期至少部分地如删除信道一样执行的通信信道从源接收到的传输的数据解码的方法，所述方法包括：

接收多个输出码元——从被编码为所述多个输出码元的输入码元的有序集生成——的至少部分，其中每个输出码元是作为具有选自有限域的系数的所述输入码元中一个或多个的线性组合而生成的，其中至少一个系数是第一有限域的成员且至少另一系数是第二有限域的成员而不是所述第一有限域的成员；以及

从任意预定数目的输出码元的接收再生所述输入码元的有序集至所需的精确度。

13. 如权利要求 12 所述的方法，其特征在于，针对所述输入码元的固定值的任意集合可能已从所述输入码元集合生成的唯一性输出码元的数目独立于所述域数组大小。

14. 如权利要求 12 所述的方法，其特征在于，所述有限域是使得第一有限域数组和第二有限域数组是不同的且所述第一有限域和所述第二有限域各自是从包含 GF(2)、GF(4)、GF(16)、GF(256)的域集合中选出的。

15. 如权利要求 14 所述的方法，其特征在于，所述第一有限域数组是 GF(2)而所述第二有限域数组为 GF(256)。

16. 如权利要求 14 所述的方法，其特征在于，所述第一有限域数组是 GF(2)

而所述第二有限域数组为 GF(4)。

17. 如权利要求 14 所述的方法，其特征在于，所述第一有限域数组是 GF(4) 而所述第二有限域数组为 GF(16)。

18. 如权利要求 14 所述的方法，其特征在于，所述第一有限域数组是 GF(16) 而所述第二有限域数组为 GF(256)。

19. 如权利要求 14 所述的方法，其特征在于，所述第一有限域数组小于所述第二有限域数组。

20. 如权利要求 14 所述的方法，其特征在于，所述第一有限域数组大于所述第二有限域数组。

21. 一种用于编码数据以便在预期至少部分地如删除信道一样执行的通信信道上从源传输至目的地的方法，所述方法包括：

获得表示所述要被编码的数据的输入码元的有序集；

选择多个域数组的值，其中每个域数组是从有限域数组导出的并且至少两个不同的有限域数组被表示；

生成表示系数矩阵——其表示所述多个域数组中的至少两个——的数据结构，其中这些域数组中的至少两个是从彼此不同的有限域数组导出的；

从所述输入码元的有序集生成多个冗余码元，其中每个冗余码元是基于具有有限域上的系数的所述输入码元中的一个或多个与其它冗余码元的一组线性约束而生成的；

生成作为输入码元的线性组合的输出码元，其中特定组合是根据表示所述系数矩阵的所述数据结构；

从输入码元和冗余码元的组合集生成多个输出码元，其中每个输出码元是作为具有选自有限域的系数的所述输入码元和冗余码元的组合集中的一或多个的线性组合而生成的；

使用所生成的输出码元并对所述数据编码。

22. 如权利要求 21 所述的方法，其特征在于，针对所述输入码元的固定值的任意集合可从所述输入码元集合生成的冗余码元的数目独立于所述域数组大小。

23. 如权利要求 21 所述的方法，其特征在于，所述有限域是使得第一有

---

限域数组和第二有限域数组是不同的，且所述第一有限域和所述第二有限域各自是从包含 GF(2)、GF(4)、GF(16)、GF(256)的域集合中选出的。

24. 一种用于将在目的地处通过预期至少部分地如删除信道一样执行的通信信道从源接收到的传输的数据解码的方法，所述方法包括：

接收从输入和冗余码元的组合集生成的多个输出码元中的至少部分，其中每个输出码元是作为具有选自有限域的系数的输入码元和冗余码元的组合集中的一个或多个的线性组合而生成的，

其中所述多个冗余码元是从所述输入码元的有序集生成的，其中每个冗余码元是基于具有有限域上的系数的所述输入码元中的一个或多个与其它冗余码元的一组线性约束而生成的，

其中至少一个系数是第一有限域的成员且至少另一系数是第二有限域的成员而不是所述第一有限域的成员；以及

从任意预定数目的输出码元的接收再生所述输入码元的有序集至所需的精确度。

25. 如权利要求 24 所述的方法，其特征在于，针对所述输入码元的固定值的任意集合可能已从所述输入码元集合生成的唯一性输出码元的数目独立于所述域数组大小。

26. 如权利要求 24 所述的方法，其特征在于，所述第一有限域是 GF(2)。

27. 如权利要求 24 所述的方法，其特征在于，所述第二有限域是 GF(256)。

28. 如权利要求 24 所述的方法，其特征在于，所述第二有限域是 GF(4)。

29. 如权利要求 24 所述的方法，其特征在于，所述第一有限域是 GF(4)。

30. 如权利要求 24 所述的方法，其特征在于，所述第一有限域是 GF(16)。

31. 如权利要求 24 所述的方法，其特征在于，所述第二有限域是 GF(16)。

---

用于通信系统的基于多域的码生成器和解码器

交叉引用

本申请要求提交于 2006 年 2 月 21 日的美国临时专利申请 No.60/775,528 的优先权并且是其非临时申请。

以下引用出于所有用途被包括和结合于此。

授予 Luby 的题为 “Information Additive Code Generator and Decoder for Communication Systems (用于通信系统的信息加性码生成器和解码器) ” 的美国专利 No. 6,307,487 (以下称为 “Luby I” ) ;

授予 Luby 等人的题为 “Information Additive Group Code Generator and Decoder for Communication Systems (用于通信系统的信息加性群码生成器和解码器) ” 的美国专利 No. 6,320,520 (以下称为 “Luby II” ) ;

授予 Shokrollahi 等人的题为 “Multi-Stage Code Generator and Decoder for Communication Systems (用于通信系统的多级码生成器和解码器) ” 的美国专利 No. 7,068,729 (以下称为 “Shokrollahi I” ) ;

授予 Shokrollahi 等人的题为 “Systematic Encoding and Decoding of Chain Reaction Codes (连锁反应码的系统编码和解码) ” 的美国专利 No. 6,909,383 (以下称为 “Shokrollahi II” ) ;

授予 Shokrollahi 等人的题为 “System and Processes for Decoding Chain Reaction Codes through Inactivation (通过钝化解码连锁反应码的系统和过程) ” 的美国专利 No. 6,856,263 (以下称为 “Shokrollahi III” ) ; 以及

Shokrollahi 于 2004 年 12 月 1 日提交的题为 “Protection of Data from erasures Using Subsymbol Based Codes (使用子码元基础码以防止数据被删除) ” 的美国专利公开 No. 2005/0219070 A1 (以下称为 “Shokrollahi IV” ) 。

发明领域

本发明涉及通信系统内的编码和解码，尤其涉及对数据进行编码和解

码以考虑传达的数据内的差错和间隙的通信系统。通信是在广泛意义上使用的，包括但并不限于任何形式的数字数据通过空间和/或时间的传输。

### 发明背景

通信信道上发送方和接收者方之间的文件传输是许多文献的主题。较优地，接收方期望能以一定的确信度接收到发送方在信道上传送的数据的精确副本。当信道没有完美的保真度时(这是大多数物理可实现系统的情况)，要考虑的问题是如何处理传输中的数据丢失或误传的数据。丢失的数据(删除)比受损的数据(差错)要更容易处理，因为接收方不能总是分辨何时接收到了错误的受损数据。研发了许多纠错码以纠正删除和/或差错。一般，使用的特定码是基于关于正通过其传送数据的信道的不保真度和正在传送的数据的性质的信息来选择的。例如，在已知信道有较长时间的不保真度时，最适合该情况的是突发差错码。在预期只有较短不频繁的差错时，最好是简单的奇偶校验码。

当发射机和接收机具有用于通信所需的全部计算功率和电功率且发射机与接收机之间的信道足够干净以允许相对无误差的通信时，数据传输是简单的。当信道处于不利环境中或者发射机和/或接收机具有有限能力时，数据传输的问题变得较为困难。

一种方案是使用前向纠错(FEC)技术，其中数据在发射机处被编码以使得接收机可从传输的删除和差错中进行恢复。在可行情况下，从接收机到发射机的反向信道允许接收机针对差错向发射机进行传达，发射机随后可相应地调节其传输过程。然而，常常反向信道是不可用或不可行的，或者仅在有限容量下可用。例如，在发射机正向大量接收机传送的情况下，发射机可能没有能力处理来自所有接收机的反向信道。作为另一示例，通信信道可以是存储介质，由此数据的传输在时间上往前的，从而除非有人发明出回到从前的时间穿梭机，否则该信道的反向信道是不可行的。因此，常需要在没有反向信道或在有限容量的反向信道的情况下设计通信协议，由此，发射机不得不在没有关于变化较大的信道状况的全景了解的情况下处理这些信道状况。

当接收机需要为可能是便携式或移动的低功率小型设备并且需要以高带

宽接收数据时，发射机与接收机之间的数据传输问题变得更为困难。例如，无线网络可能被建立成从固定发射机向大量或不确定数目的便携式或移动接收机递送文件或流送——作为广播或多播，其中接收机受限于其计算能力、存储器大小、可用电力、天线尺寸、设备尺寸和其它设计约束。另一示例是在存储应用中，其中接收机从在原始数据的再现上呈现不保真度的存储介质检索数据。这种接收机常与其存储介质被嵌入在设备中，例如在计算能力和电力上高度受约束的盘驱动器。

在这种系统中，所要解决的是考虑包括具有极小或没有反向信道、有限的存储器、有限的计算周期、功率、移动性和定时。较优地，设计应最小化向潜在的大量接收机递送数据所需的传输时间量，其中个别接收机可能在不可预测的时间被转为开或关，移进和移出范围，遭遇由于链路差错、移动性、拥塞而造成的丢失，从而迫使较低优先级文件或流送分组被临时丢弃等。

在可能丢失分组的信道上的数据传输所用的分组协议的情况下，文件、流送或将在分组网络上传送的其它数据块被划分成相等大小的输入码元，使用 FEC 码从输入码元生成与输入码元相等大小的编码码元，且编码码元被放在分组中发送。不管码元是否实际被分成比特流，码元的“大小”可以用比特测量，其中当码元从  $2^M$  码元的字母表中选出时该码元大小为 M 比特。在这种基于分组的通信系统中，面向分组的删除 FEC 编码方案可能是合适的。如果目标接收方能在网络内即使有删除的情况下也能以及时方式恢复流送的每个部分的精确副本，则称该流送传输是可靠的。文件传输和流送传输两者也可以是一定程度上可靠的，在这个意义上，文件或流送的某些部分是不可恢复的，或者对于流送，如果该流送的某些部分不能以及时方式恢复。由于不定时的拥塞导致路由器中的缓冲机制接近其容量进而迫使丢弃传入的分组，所以经常发生分组丢失。传输期间针对删除进行保护已经是许多研究的主题。

在会损坏比特的有噪信道上的数据传输所用的协议的情况下，将在数据传输信道上传送的数据块划分成相等大小的输入码元，从输入码元生成相同大小的编码码元并且在该信道上发送编码码元。对于这种有噪信道而言，码元的大小通常是 1 比特或几比特，不论码元是否被实际分成比特流。在这种通信系统

中，面向比特流的纠错 FEC 编码方案可能是合适的。如果允许目标接收方能够在有差错（不论在该信道中检测或是未检测到的码元损坏）的情况下恢复原始块的精确副本，则该数据传输被称为是可靠的。该传输也可以是一定程度上可靠的，在这种意义上，块的某些部分在恢复以后仍然损坏。码元经常由于不定时的噪声、周期性噪声、干扰、弱信号、信道堵塞、以及各种其它因素而被损坏。传输期间针对数据损坏进行保护已经成为许多研究的主题。

连锁反应码是允许从文件或流送的固定输入码元生成任意数目的输出码元的 FEC 码。有时，它们被称为喷泉或无率 FEC 码，因为该码没有先验固定的传输率。连锁反应码具有许多用途，包括相比于信息复制方式以信息加性方式生成任意数目的输出码元，其中后者是接收机在能够恢复输入码元之前接收到的输出码元复制已接收到的信息并由此不提供可用于恢复输入码元的信息。用于生成、使用和操作连锁反应码的新颖技术在例如 Luby I、Luby II、Shokrollahi I 和 Shokrollahi II 中示出。

连锁反应码生成的输出码元的一个属性是接收机能够一旦接收到足够的输出码元就能恢复原始文件或原始流送的块。具体而言，为了以较高概率恢复原始的  $K$  个输入码元，接收机需要约  $K+A$  个输出码元。比率  $A/K$  被称为“相对接收开销”。相对接收开销取决于输入码元的数目  $K$  和解码器的可靠性。

还已知可使用多级连锁反应（“MSCR”）码，诸如在 Shokrollahi I 和/或 II 中所描述且由数字方敦公司所研发的商标名为“Raptor”码的码。多级连锁反应码被用于例如从源文件或源流送接收输入码元、从输入码元生成中间码元并使用连锁反应码将中间码元编码的编码器中。特别地，从要被传送的输入码元的有序集生成多个冗余码元。从包括输入码元和冗余码元的组合码元集生成多个输出码元，其中可能输出码元的数目远大于该组合码元集中码元的数目，其中至少一个输出码元是从该组合码元集中一个以上的码元和从该组合码元集中不足所有的码元生成的，并使得输入码元的有序集可从任意数目  $N$  个输出码元被再生至所需的精确度。还已知可使用以上所述的技术来编码和解码系统码，其中输入码元被包括在该码的可能输出码元当中。这可以如在 Shokrollahi II 中所述的通过在上述步骤之前对输入码元应用变换实现，所述增强过程导致该码生成的第一输出码元等于输入码元。如差错或删除编码领域的技术人员所

明了的，Shokrollahi II 的技术可直接应用到本文所述或建议的码。

对于某些应用，码的其它变形可能更为适合或优选。

上述 MSCR 码和连锁反应码在它们的编码和解码复杂度方面是极为有效的。它们这种效率的原因之一是所执行的操作是域 GF(2)上的线性操作，GF(2)即一个比特上的简单域，其中将两个域元素相加的操作是简单的逻辑 XOR（异或）操作，而将两个域元素相乘的操作是简单的逻辑 AND（与）操作。一般而言，这些操作在多个比特上并发地执行，例如一次 32 比特或一次 4 字节，并且这些操作在所有现代 CPU 处理器上是自然地受支持的。另一方面，当用作删除 FEC 码时，由于 GF(2)上的操作，使得对于超出前 K 个码元之外的每个额外码元，接收机能够解码所有输入码元的机会下降至多约一半，其中 K 是原始输入码元的数目。例如，如果  $K+A$  个编码码本被接收，则恢复过程无法恢复这 K 个原始输入码元的机会至少为  $2^{-A}$ 。如果解码失败的机会作为 A 的函数下降得更快那么这将是更有利的行为。

存在其它在较大的域上操作的删除和纠错 FEC 码，例如在 GF(4)上、或在 GF(8)上、或在 GF(256)上、或者更一般地在 GF( $2^L$ )上( $L>1$ )操作的 Reed-Solomon 码，以及还有在较大域上操作的 LDPC 码。这些 FEC 码的优点在于，例如在删除 FEC 码的情况下，解码失败的机会作为 A 的函数比 GF(2)上的 FEC 码下降得更快。另一方面，这些 FEC 码在编码和解码复杂度上通常效率低得多，并且主要原因之一是较大域上的操作更为复杂和/或在现代 CPU 上不是自然地被支持，且复杂度通常随域大小而增长。因此，域在 GF(2)上操作的 FEC 码相比，在较大有限域上操作的 FEC 码常常较慢或不切实际。

因此，所需要的是一种在其编码和解码复杂度方面极具效率同时，又具有解码失败的机会作为超出理想 FEC 码恢复原始输入码元所需的最小数目之外的收到码元的数目的函数非常快速地降低这样的属性的纠删和纠错 FEC 码。

### 发明简要

根据本发明的一个实施例，提供了一种用于编码数据以便在通信信道上从源传输至目的地的方法。该方法对输入码元的有序集操作并且可从输入码元生成零个或多个冗余码元，每个冗余码元等于具有取自一个或多个有限域的系数

的多个输入码元的线性组合，其中所用的有限域在不同输入码元之间和在不同的冗余码元之间可不同。该方法包括从包括输入码元和冗余码元（如果有任何冗余码元）的码元组合集生成多个输出码元，其中每个输出码元可从所组合的输入和冗余码元中的一个或多个生成，其中每个输出码元是作为具有选自一个或多个有限域的系数的多个输入码元和冗余码元的线性组合而生成的，其中所用的有限域在不同输入和冗余码元之间、在不同的输出码元之间以及在输出码元与冗余码元之间可以不同，并使得可从任意预定数目的输出码元再生输入码元的有序集至所需的精确度。

该方法还可被用来生成输出码元，其中可从输入码元的固定集合生成的可能的输出码元的数目可以比输入码元的数目大得多。

根据本发明的另一实施例，该方法包括在目的地接收通过通信信道发送来自源的输出码元中的至少部分，其中信道上的传输可导致部分发送码元的丢失或损坏并且一些收到的码元已知被正确接收且关于码元的损坏程度的信息也可被提供。该方法包括在目的地再生输入码元的有序集至所需的精确度——这取决于接收到多少码元以及收到码元的损坏的知识。

该实施例还可包括在目的地接收输出码元的至少部分，其中可接收到的可能输出码元的数目可以比输入码元的数目大得多。

根据本发明的另一实施例，提供了一种用于编码数据以便在通信信道上从源发送到目的地的方法。该方法对输入码元的有序集操作并包括从输入码元生成多个冗余码元。该方法还包括从包括输入码元和冗余码元的码元组合集生成多个输出码元，其中在生成输出码元时所应用的操作是在较小的有限域（例如， $GF(2)$ ）上，并且使得可从任意预定数目的输出码元再生输入码元的有序集至所需的精确度。这多个冗余码元是从输入码元的有序集生成的，其中生成冗余码元的操作是在并非为  $GF(2)$  的有限域上或者是在一个以上的有限域的混合上（例如，一些操作在  $GF(2)$  上，一些操作在  $GF(256)$  上）。

根据本发明的又一实施例，使用类似技术提供了一种用于接收通过通信信道传送来自源的数据的系统。该系统包括耦合至通信信道的用于接收通过通信信道传送的输出码元的接收模块，其中每个输出码元是从包括输入码元和冗余码元的码元组合集中的至少一个码元生成的，其中在生成输出码元时所应用的

操作是在较小的有限域（例如， $GF(2)$ ）上，并且使得可从任意预定数目的输出码元再生输入码元的有序集至所需的精确度，其中输入码元是来自输入码元的有序集，其中冗余码元是从输入码元生成的，且其中这多个冗余码元是从输入码元的有序集生成的，其中生成冗余码元的操作是在并非为  $GF(2)$  的有限域上（例如， $GF(256)$ ）或者是在一个以上的有限域的混合上（例如，一些操作在  $GF(2)$  上，一些操作在  $GF(256)$  上）。

根据本发明的其它实施例，提供了一种内含在载波中的计算机数据信号。

通过本发明实现了许多益处。例如，在一特定实施例中，用于编码数据以便在信道上传输的计算开销被降低。在另一特定实施例中，解码这些数据的计算成本被降低。在其它特定实施例中，解码器的差错概率被降低，同时保持了较低的编解码计算开销。取决于实施例，可实现这些益处中的一个或多个。这些和其它益处将在本说明书中更具体地提供。

所公开的本发明的性质和优点的进一步理解可参照说明书和附图的剩余部分来了解。

### 附图简述

图 1 是根据本发明的一个实施例的通信系统的框图。

图 2 是根据本发明的一个实施例的编码器的框图。

图 3 是根据本发明的一个实施例生成冗余码元的方法的简易框图。

图 4 是根据本发明的一个实施例的静态编码器的基本操作的简易框图。

图 5 是根据本发明的一个实施例的动态编码器的简易框图。

图 6 是根据本发明的一个实施例的动态编码器的基本操作的简易框图。

图 7 是根据本发明的一个实施例的静态编码器的简易框图。

图 8 是根据本发明的一个实施例的静态编码器的基本操作的简易框图。

图 9 是根据本发明的一个特定实施例用于计算编码参数的方法的简易框图。

图 10 是根据本发明的另一实施例的静态编码器的简易流程图。

图 11 是根据本发明的一个实施例的解码器的简易框图。

图 12 是根据本发明的一个实施例的解码器的操作的简易流程图。

图 13 是根据本发明的另一实施例的解码器的操作的简易流程图。

图 14 是根据本发明的又一实施例的解码器的操作的简易流程图。

图 15 是根据本发明的一个实施例的动态解码器的简易框图。

图 16 是根据本发明的一个实施例的静态解码器的简易框图。

图 17 示出了来自子码元映射的源码元。

图 18 示出了各种文件大小的文件下载参数的可能设置。

图 19 示出了各种源块大小的流送参数的可能设置。

图 20 示出了表示源码元与中间码元之间的关系的矩阵的形式。

图 21 示出了幂度生成器的幂度分配。

图 22 示出了可被用于解码的矩阵 A 的形式。

图 23 示出了可被用于解码的矩阵 A 的块分解。

图 24a 示出了可被用于解码的矩阵 A 的块分解。

图 24b 示出了在解码过程的第一阶段的若干步骤之后矩阵 X 的块分解。

图 25 示出了在某些消去步骤之后矩阵 X 的块分解。

图 26 示出了在进一步消去步骤之后 X 的子矩阵的块分解。

图 27 示出了在消去和删除步骤之后矩阵 A 的块分解。

图 28 示出了在进一步消去和删除步骤之后矩阵 A 的块分解。

图 29 示出了在进一步消去步骤之后矩阵 A 的块分解。

图 30 示出了在又某些其它消去步骤之后矩阵 A 的块分解。

图 31 示出了根据本发明的一个优选实施例构造的(120,100)码的码失败概率的表。

图 32 示出了根据本发明的一个优选实施例构造的(110,100)码的码失败概率的表。

具体描述之后跟有三个附录：附录 A 包含系统索引  $J(K)$  的示例值；附录 B.1 包含表  $V_0$  的示例值；以及附录 B.2 包含表  $V_1$  的示例值。

### 特定实施例的具体描述

本文所描述的发明利用数学运算来基于一个或多个有限域中的运算进行编码和解码。有限域是为其定义了四种算术运算有限代数结构，并且形成关于

这些操作的域。它们的理论和它们的构造是本领域的技术人员所熟知的。

在以下描述中，我们将需要在有限域的元素与表示将被编码或解码的数据或从其导出的码元之间定义乘法过程。在以下描述中考虑三种不同类型的码元：输入码元包括发送方已知的将被传达给接收方的信息，冗余码元包括从输入码元导出的码元，而输出码元包括发送方传送给接收方的码元。定义这种乘法过程的许多可能性当中，我们集中在特定两种：简单变换，以及交织变换。

### 简单变换

在该情形中，乘法过程是在来自有限域  $GF(2^M)$  的元素  $a$  与长度为  $M$  比特的码元  $S$  之间定义的。如在此所使用的，“码元”是指通常小于源块的一段数据。码元的大小通常可用比特来衡量，其中码元具有  $M$  比特的大小且该码元选自  $2^M$  个码元的字母表。在分组网络上可靠传输信息的应用中，例如码元的大小可以等于分组大小，或者可以比分组大小要小，以使得每个分组包含一个或多个码元。

在简单变换的情形中，码元  $S$  被解释为  $GF(2^M)$  的元素，而乘法  $a*S$  被定义为域  $GF(2^M)$  中的标准乘法。对码元执行的操作被称为码元的“简单变换”。作为说明性示例，考虑域  $GF(4)$ 。 $GF(4)$  的元素例如根据其二进制展开可用 2 比特来表示。域  $GF(4)$  具有四个域元素 00、01、10、11，其中加法是比特串的标准异或，而乘法经由下表来定义：

	00	01	10	11
00	00	00	00	00
01	00	10	10	11
10	00	10	11	10
11	00	11	10	10

根据以上乘法表， $10*01$  的结果将是 10，因为 01 是该域中的乘法中性元素（有时被称为单位元素）。

### 交织变换

为了示出交织变换，我们将利用环的数学概念。如本领域的技术人员所公知的，环是其上定义了满足分配律的加法和乘法这两种运算的集合。此外，仅考虑加法的该集合构成了阿贝尔群（abelian group），即加法的结果与被加数的顺序无关，对于加法有一中性元素 0，以及对于每个元素都存在另一元素以使得这些元素的和为 0。其它要求是乘法具有中性元素 1，以使得任何元素与 1

的乘积不会改变该元素的值。对于普通环而言，我们不要求任何非零元素都具有乘法逆元素，也不要求乘法是可交换的。然而当满足了这两个条件时，则我们将该环称为“域”。此概念是代数学中的一个标准概念。

映射是可以硬件、软件、数据存储等来实现的一种逻辑构造，它将相同大小的码元对映射到该大小的另一码元。我们用“ $\oplus$ ”来标示该映射，并用  $S \oplus T$  来标示对码元对(S,T)的这种映射的映像。这种映射的一个示例是逐位的异或(XOR)。

这里使用的另一构造是一特殊类型的集合对码元的“动作”。假定  $A$  是具备了可交换加法运算(“+”)的集合，该加法运算具有中性元素并且对于每一元素都包含其加法逆元素。这样一个集合通常也被称为阿贝尔群。该群对码元集合的“动作”是将由群元素  $r$  和码元  $S$  组成的对映射为另一码元的映射。当该映射在群中遵守加法时，即对于群  $A$  中的每对元素  $a$  和  $b$ ，都有  $(a+b)*S = a*S \oplus b*S$ ，我们用  $r*S$  来标示这种映像。如果  $A$  是环且该动作还在  $A$  中遵守乘法，在乘法运算符在  $A$  中为“.”的情形中即  $(a \cdot b)*S = a*(b*S)$ ，则该动作就是有限域的元素域码元之间所要求的乘法过程。这样我们说域对码元集“操作”。以这种方式对码元执行的操作被称为“交织变换”。

这种乘法过程还有众多示例。以下仅提及了少量示例。该示例列表仅用于说明的目的，而不应被考虑为穷尽性列表，也不应被理解为限制本发明的范围。

由域元素 0 和 1 组成的域 GF(2)，其中加法为异或(XOR)且乘法为逻辑运算 AND(与)，通过定义  $1*S = S$  且  $0*S = \mathbf{0}$  来对码元集合进行运算，其中  $S$  标示任意码元而  $\mathbf{0}$  标示全部由 0 组成的码元。

域 GF(4)可按以下方式对偶数大小的码元操作：对于码元  $S$ ，我们分别用  $S[0]$  和  $S[1]$  来标示其前一半和后一半，从而使得  $S=(S[0], S[1])$  是  $S[0]$  和  $S[1]$  的级联。随后，我们定义

$$\begin{aligned} 00 * S &= \mathbf{0} \\ 01 * S &= S \\ 10 * S &= (S[1], S[0] \oplus S[1]) \\ 11 * S &= (S[0] \oplus S[1], S[0]). \end{aligned}$$

可以很快核实这确实是有效运算。可看出，该域的乘法表描述了与以上在 2 比特码元情形中所定义的运算相一致的运算。

或者，域 GF(4)可按以下方式对偶数大小的码元操作：对于码元  $S$ ，我们

用  $S[0]$  标示  $S$  内偶数位置上比特的级联，类似地我们用  $S[1]$  标示  $S$  内奇数位置上比特的级联（其中各位置是以 0 开始顺序编号的）。对于两个相等长度的比特串  $A$  和  $B$ ，令  $(A \mid B)$  被定义为两倍长度的比特串  $C$ ，其中  $C$  的位置  $2*i$  中的比特是  $A$  的位置  $i$  中的比特，而  $C$  的位置  $2*i+1$  中的比特是  $B$  的位置  $i+1$  中的比特。然后，我们定义

$$\begin{aligned} 00 * S &= 0 \\ 01 * S &= S \\ 10 * S &= (S[1] \mid S[0] \oplus S[1]) \\ 11 * S &= (S[0] \oplus S[1] \mid S[0]). \end{aligned}$$

可以很快核实这确实是有效运算。可看出，以上所定义的所有运算在域 2 比特码元情形中相同。

上述交织变换可被视为交织变换的特例，其中域的元素的二进制长度与该码元按比特计的长度相一致，且域元素对码元的操作与该有限域中的乘法相同。

更一般地，如果  $K$  是幂度（degree）为  $d$  的 GF(2) 的扩展域，则可在其大小可被  $d$  除尽的码元上定义该域的一种运算。这种运算在作为 1995 年伯克利的国际计算机科学协会的技术报告编号 TR-95-048 出版的 Bloemer、Kalfane、Karpinski、Karp、Luby、以及 Zuckerman 的“An XOR-Based Erasure Resilient Coding Scheme（基于 XOR 的删除弹性编码方案）”中进行了描述。这种方案使用了对作为具有二进制项的  $d \times d$  矩阵的域  $K$  的所谓“正规表示（regular representation）”。

对于这些广义化，第一交织变换将长度为  $d*I$  比特的串的  $S$  分割为  $d$  个相等大小的部分，其中第一部分  $S[0]$  是  $S$  的第一  $I$  个比特， $S[1]$  是  $S$  的下一  $I$  个比特， $S[d-1]$  是  $S$  的最后  $I$  个比特。该变换对  $S$  的  $d$  个部分操作并生成被级联在一起以形成操作结果的  $d$  个部分。或者，第二交织变换将  $S$  分割成  $d$  个相等大小的部分，其中第一部分  $S[0]$  是  $S$  中以位置 0 开始的每个第  $d$  个比特的级联，第二部分  $S[1]$  是  $S$  中以位置 1 开始的每个第  $d$  个比特的级联，第  $d$  部分  $S[d-1]$  是  $S$  中以位置  $L-1$  开始的每个第  $d$  个比特的级联。该第二变换对  $S$  的  $d$  个部分操作（与第一变换完全相同）并生成被交织在一起以形成操作结果的  $d$  个部分。

注意，第一交织变换可通过将原始串  $S$  的连续比特一起 XOR 来计算，并且这对于软件实现是一优点，因为通常 CPU 自然地支持这种运算。另一方面，

运算结果中特定位置中比特的值取决于原始串  $S$  的长度，并且如果希望在支持可变长度码元的硬件中实现该运算，这在某种程度上将是一缺点，因为硬件的操作将取决于码元长度而有所不同。注意，第二交织变换涉及原始串当中非连续比特的一起 XOR，并且这对于软件实现而言在某种程度上是一缺点，因为 CPU 并不是将这种 XOR 作为自然操作来支持。不过，对码元的有限域元素直接工作的软件操作可用软件非常高效地实现，因此第二交织变换的软件实现是可能的。此外，对于第二交织变换，运算结果中特定位置中比特的值不依赖于原始串  $S$  的长度，如果想要以支持可变长度码元的硬件实现该操作那么这是一优点，因为硬件的操作可独立于码元长度。因此，第二交织变换相对于第一交织变换确实具有一定总体优势。

### 线性变换

“线性变换”的概念可参照简单或交织变换来定义。对于给定的整数  $m$  和  $n$ ，由该运算导出的线性变换使用具有指定域中的项的矩阵空间将  $n$  个码元的向量映射为  $m$  个码元的向量。域  $F$  上的矩阵是项的 2 维集，其中每项都属于  $F$ 。如果矩阵具有  $m$  行和  $n$  列，则其通常被称为  $m \times n$  矩阵。对  $(m, n)$  被称为该矩阵的“格式”。具有相同格式的多个矩阵可使用底层域或环中的加法和减法而被加或减。如所公知的，格式  $(m, n)$  的矩阵可被乘以格式为  $(n, k)$  的矩阵。

在运算中，如果  $B$  标示格式为  $(m, n)$  的矩阵，且  $B[j, k]$  标示  $B$  中在位置  $(j, k)$  上的项，并且如果  $S$  标示包含码元  $S_1, S_2, \dots, S_n$  的列向量，而  $X$  标示包含码元  $X_1, X_2, \dots, X_m$  的列向量，则该变换可被表达成：

$$X = B \otimes S$$

因此，以下关系成立：

$$\text{对于从 1 到 } m \text{ 的所有 } j, X_j = B[j, 1]*S_1 \oplus B[j, 2]*S_2 \oplus \dots \oplus B[j, n]*S_n$$

其中，“\*”标示简单变换或交织变换。

以上公式描述了在编码器或解码器中称为“简单变换过程”的从  $B$  和  $S$  计算  $X$  的过程，它可由以下步骤执行：

1. 将  $j$  设为 1 以及  $X_j$  设为  $\mathbf{0}$ 。
2. 对于  $k$  的值从 1 到  $n$ ，作  $X_j = X_j \oplus B[j, k]*S_k$ 。
3. 将  $j$  递增 1。如果  $j$  大于  $m$ ，则停止，否则转到步骤 2。

这种线性变换在各种应用中是常见的。例如，在使用线性码来对一段数据或源块进行编码时， $S$  可以是要编码的源块的源码元， $X$  可以是  $S$  的经编码的版本，而  $B$  可以是该码的生成矩阵。例如在所用码是系统性的其它应用中， $X$  可以是  $S$  的编码的冗余码元，而  $B$  可以是描述冗余码元对源码元的相关性的矩阵。

如本领域的技术人员所已知的，通过提供通用处理器内所执行的指令、通过专门设计成执行这些运算的硬件、或这两者的组合来执行上述运算的方法是已知的。在所有情形下，当使用了较大的有限域时，在所需的指令数目、所需的硬件量等方面的运算成本、硬件成本、运算所消耗的电功率和/或执行运算所需的时间一般也较大。特别地，在域 GF(2)的情形中，所需操作等价于在通用处理器内广泛提供的逐位的 AND 和 XOR 运算，并且在需要时可在硬件中简单、快速且廉价地实现。相反，使用大于 GF(2)的有限域的运算很少在通用处理器中直接提供并且要求专用硬件或大量处理器指令和存储器操作来实现。

#### 多域纠删和纠错码

本文参照广义矩阵描述对多域纠删和纠错码的诸多特定实施例进行了描述。本方法仅被采用作为描述性工具而不表示描述本文所述的各实施例的唯一方法，也不应将其理解为限制本发明的范围。在广义描述中，这样来构造矩阵，其元素是取自一个或多个有限域。不同元素可取自不同的有限域，它们具有一种属性，即存在一单个域，可将所有的域嵌入在其中并且选择特定的这种嵌入。如以下进一步示出的，输出码元的部分或所有可与输入或冗余码元的部分相同，或者可与输入和冗余码元不同，这取决于所选择的特定实施例。

在码的输入码元与矩阵的某些列之间作出一一对应关系。在码的冗余码元与矩阵的剩余列之间作出又一个一一对应关系。此外，矩阵当中等于冗余码元数目的这些数目的行被指定为静态行。矩阵当中剩余的行被指定为动态行。在矩阵的动态行与码的输出码元之间作出一一对应关系。在该描述中，静态行表示在输入和冗余码元之间成立所需的约束，且这些静态行完全定义输入码元和冗余码元之间的关系，以使得输入码元和静态行的知识足以构建冗余码元。动态行表示在信道上实际发送的输出码元。在许多码中，输入和/或冗余码元自身被发送，且在本描述中这通过针对要传送的每个输入和冗余码元添加一动态行

来表示，所述动态行在对应该所要求的输入或冗余码元的列中具有非零项而在其余列中具有零项。在某些实施例中，该非零项是单位元素。在其它实施例中，该非零项无需是单位元素。

上述矩阵形式可被用来确定用于将要在通信信道上从源向目的地传输的数据进行编码的方法，该方法包括从输入码元的有序集生成多个冗余码元，其中每个冗余码元是基于具有有限域上的系数的一个或多个输入码元和其它冗余码元的一组线性约束生成的，所述线性约束对应于矩阵描述的静态行，从输入和冗余码元的组合集生成多个输出码元，其中每个输出码元是被生成为该输入和冗余码元的组合集中具有从有限域选出的系数的一个或多个的线性组合，所述线性约束对应于矩阵描述的动态行，以及发送这多个生成的输出码元中的至少部分。

相反，包括以上步骤的方法可用以上所述的这种矩阵形式来描述，其中静态行对应于对一个或多个输入码元和冗余码元的线性约束，而动态行对应于被用来形成输出码元的输入和冗余码元的线性组合。实际上，上述方法的实施例可以不涉及所述矩阵的显式或隐式表示或构造。

如所公知的，在矩阵的所有元素都取自域 GF(2)的情形中，可以用这种方式来描述一大类的公知纠错和纠删码。例如，对于低密度奇偶校验（LDPC）码的情形，包括例如在 V. Roca 和 C. Neumann 在 2004 年 6 月作为 INRIA 研究报告 RR-5225 出版的题为 “Design, Evaluation and Comparison of Four Large Block FEC Codecs, LDPC, LDGM, LDGM Staircase and LDGM Triangle, plus a Reed-Solomon Small Block FEC Codec (LDPC、LDGM、LDGM 阶梯和 LDGM 三角等四种大型块 FEC 编解码器加上 Reed-Solomon 小型块 FEC 编解码器的设计、评估和比较)” 的文献（可在 [www.inrialpes.fr](http://www.inrialpes.fr) 得到）（在此被称为“Roca”）中所述的情形，广义矩阵可通过将奇偶校验矩阵的每一行指定为静态行并如上所述地针对每个输入和冗余码元添加另一动态行来从奇偶校验矩阵构造。另一示例可使用如在 Luby I 和 Luby II 中所述的单级连锁反应码，其中矩阵中静态行的数目为零且动态行包括标准连锁反应矩阵。另一示例是 MSCR 码的使用，在这种情形中，这里的广义描述等价于这些码的标准矩阵表示。

较大域上的其它码也可以这种方式表示。例如，诸如从其中输入码元是源码元的 Vandermonde 矩阵导出的 Reed-Solomon 码，广义矩阵等于 Vandermonde 矩阵且所有行都是动态的，在这种情形下，每项是来自在其乘法群中至少具有所存在的总共的行和列数个元素的域的有限域元素，例如当总共的行和列数小于 256 时的有限域 GF(256)。另一示例是从 Vandermonde 矩阵导出的诸如 GF(256) 的有限域上的系统性 Reed-Solomon 码，在这种情形下，输入码元是源码元，冗余码元是奇偶码元，而矩阵是 Vandermonde 矩阵的系统性形式内对应这些奇偶码元的行，且所有这些行被考虑为静态并且如上所述地针对每个源码元和奇偶码元添加额外动态行，因为这些正好是在信道上发送的码元。

如纠错和纠删码领域的技术人员所公知的，纠错和纠删的共同所需要属性包括低编码复杂度、低解码复杂度、低解码差错概率和低差错基底。码的复杂度是对该码进行编解码所需的计算资源的衡量。低复杂度在其中编码或解码将由诸如移动终端、消费电子设备、存储设备或可同时处理许多编码或解码操作的设备等资源受限设备执行的情况下具有由其具有价值。计算复杂度部分地是用以对该码进行编解码的矩阵的密度以及从其取用矩阵元素的有限域的大小的函数。密集矩阵通常会导致较高复杂度，且这已经导致许多基于稀疏矩阵的码设计，例如低密度奇偶校验码和连锁反应码。较大的有限域也会导致较高复杂度，这已经导致基于小型域——最常见的是 GF(2)——的码设计。

该上下文中指的差错概率是不可能完全成功解码的概率。给定纠错或纠删码的差错概率是信道上所接收到的信息以及用于解码的特定算法的函数。在纠删码的情形中，只要接收少于输入码元数目的码元，则差错概率即为 1。理想删除码具有这样的属性，即只要接收到大于或等于输入码元数目的码元则差错概率为零。其它码在这种情形下具有非零的失败概率。

已知理想删除码可使用密集矩阵来构造，特别是 Reed-Solomon 码。然而，在 Reed-Solomon 码的情形中，所需的域的大小是作为输入码元和冗余码元的数目的和的码大小的函数，并且这一事实与矩阵的密度一起导致通常较高的计算复杂度，特别是在码大小增长时。此外，在低密度码的情形中，已知可使用较大的有限域来降低纠错码（例如在 M.C Davey 和 D.J.C MacKay 的论文“Low Density Parity Check Codes over GF( $q$ )（GF( $q$ )上的低密度奇偶校验码）”中所

示例，该论文出现在 IEEE Communications Letters，第二卷，第六期，165-167 页，1998）和删除码的差错概率。另外，已知在低密度码中引入少量高密度矩阵行或列可改善差错概率，在差错概率与复杂度之间提供折衷（MSCR 码和连锁反应码）。然而，所有这些码的缺点在于在低复杂度与低差错概率之间总是有显著的权衡。

对于许多 FEC 码——即 LDPC 码和连锁反应码和 MSRC 码，随着接收到比输入码元数目更多的输出码元，成功解码的差错概率以一定速率呈指数下降。这种码的差错基底是额外输出码元的接收使得差错概率以比在接收到的输出码元的数目刚开始超过输入码元的数目时慢得多的速率使差错概率下降时的差错概率。已知少量高密度行或列的使用和/或对矩阵使用较大的有限域可以较高的计算复杂度为代价得到较低的差错基底。许多具有低复杂度的已知纠错和纠删码的缺点是差错基底较高而不太理想。

这里，针对解决了上述部分缺点的纠错和纠删码的构造描述新的方法。用于对这些码进行高效编解码的方法是关于本文作为示例描述的特定实施例来呈现的。

如本文所述从一个以上可能的域的集合当中选择用于矩阵元素的域允许这样一种码的设计，即具有较大的域上码的低差错概率和差错基底的情况下保留小型域上码的低计算复杂度，由此表现了相对于现有技术的显著优势。

将在以下具体描述的一个优选实施例中，对于大多数行而言，各项选自 GF(2)，而对于剩余行，各项选自 GF(256)。在另一实施例中，对于每一行，一项选自 GF(256)而剩余元素选自 GF(2)。

与本领域中从同一域中选择所有元素的码相比，有许多使用来自一个以上的域的元素的其它可能实施例，这将导致计算复杂度与差错概率和差错基底之间权衡的改善。

如本文所使用的，术语“文件”是指存储在一个或多个源上并将作为单元递送至一个或多个目的地的任何数据。因此，文档、图像和来自文件服务器或计算机存储设备的文件都是可递送的“文件”的示例。文件可以具有已知大小（诸如存储在硬盘上的一兆字节的图像）或可以具有未知大小（诸如从流送源的输出所取出的文件）。总之，文件是输入码元的序列，其中每个输入码元具

有该文件中的位置和值。

如本文所使用的，术语“流送”是指在一个或多个源上存储或生成并按其生成的次序在每个时间点以特定的速率被递送至一个或多个目的地的任何数据。流送可以是固定速率或可变速率。因此，MPEG 视频流送、AMR 音频流送、以及用以控制远程设备的数据流送都是可递送的“流送”的示例。流送在每一时间点的速率可能是已知的（诸如每秒 4 兆比特）或未知的（诸如每个时间点的速率事先未知的可变速率流送）。总之，流送是输入码元的序列，其中每个输入码元在该流送中具有位置和值。

传输是通过信道将数据从一个或多个发送方传送至一个或多个接收方以递送文件或流送的过程。发送方有时也被称为编码器。如果一个发送方通过完美信道被连接至任意数目的接收方，则收到数据可以是输入文件或流送的精确副本，因为所有数据将被正确接收。这里，我们假定信道是不完美的，这是大多数现实世界信道的情形。在许多信道缺陷中，两种感兴趣的缺陷是数据删除和数据不完整性（可视为数据删除的特例）。数据删除发生在信道丢失或丢弃数据时。数据不完整性在以下情形中发生：接收方直到部分数据已经传过才开始接收数据，接收方在传输结束之前停止接收数据，接收方选择仅接收传送数据的一部分，和/或接收方间歇地停止并再次开始接收数据。作为数据不完整性的示例，移动卫星发送方可能正在传送表示输入文件或流送的数据并且在接收方在范围内之前开始传输。一旦接收方在范围内，就可以接收数据直至卫星移出范围，此时接收方可将其卫星盘式天线重新定向（在此期间不接收数据）以开始接收关于已移进范围的另一卫星所传送的同一输入文件或流送的数据。如从阅读该描述所应明了的，数据不完整性是数据删除的特例，因为接收方可将数据不完整性作为就像该接收方一直都在范围内但信道丢失了直到接收方开始接收数据时所有的数据这种情形来处理（且该接收方具有同样问题）。同时，如通信系统设计中所公知的，可检测的差错可通过简单地丢弃具有可检测差错的所有数据块或码元而被考虑成等价于删除。

在某些通信系统中，接收方接收多个发送方生成、或由一个发送方使用多个连接生成的数据。例如，为了加速下载，接收方可能同时连接至传送关于相同文件的数据的一个以上的发送方。作为另一示例，在多播传输中，可传送多

个多播数据流送以允许各接收方连接至这些流送中的一个或多个以使得总传输速率与将它们连接至发送方的信道的带宽相匹配。在所有这些情形中，所要考虑的是确保所有传送的数据对于接收方独立使用，即多个源数据在各流送之间不是冗余的，即使在传输速率对于不同流送差异极大时以及在有随意的丢失模式时也是如此。

一般而言，通信信道是将发送方与接收方相连以进行数据传输之用。通信信道可能是实时信道，其中该信道在得到数据是将该数据从发送方移到接收方，或者通信信道可能是在从发送方到接收方的传送中存储部分或所有数据的存储信道。后者的示例是盘存储或其它存储设备。在该示例中，产生数据的程序或设备可被视为向存储设备传送数据的发送方。接收方是从该存储设备读取数据的程序或设备。发送方用以将数据放至存储设备上的机制、存储设备本身以及接收方用以从存储设备获得数据的机制一起形成信道。如果这些机制或存储设备有丢失数据的可能，则这将被示为通信信道中的数据删除。

当发送方和接收方相隔其中码元可能被删除的通信信道时，较优地是不传送输入文件或流送的精确副本，而是传送从该输入文件或流送生成的有助于恢复删除的数据（可能包括输入文件或流送本文的全部或部分）。编码器是处理该任务的电路、设备、模块或代码段。一种查看编码器的操作的方法是编码器从输入码元生成输出码元，其中输入码元值的序列代表输入文件或流送的块。每个输入码元将由此在该输入文件或流送块中具有位置以及值。解码器是从接收方接收到的输出码元重构输入码元的电路、设备、模块或代码段。在多级编码中，编码器和解码器被进一步分为各自执行不同任务的子模块。

在多级编码系统的实施例中，编码器和解码器可被分为各自执行不同任务的子模块。例如，在某些实施例中，编码器包括本文所称的静态编码器和动态编码器。如本文所使用的，“静态编码器”是从输入码元集生成多个冗余码元的编码器，其中冗余码元的数目是编码之前确定的。静态编码码的示例包括 Reed-Solomon 码、Tornado 码、Hamming 码、低密度奇偶校验（LDPC）码等。术语“静态解码器”在本文被用于表示可将由静态编码器编码的数据解码的解码器。

如本文所使用的，“动态编码器”是从输入码元集以及可能的冗余码元集

生成输出码元的编码器。在本文所述的一个优选实施例中，可能的输出码元的数目大于输入码元数目的量级，且要生成的输出码元的数目无需固定。这种动态编码器的一个示例是连锁反应编码器，诸如 Luby I 和 Luby II 中所述的编码器。术语“动态解码器”在本文用来表示可将由动态编码器编码的数据解码的解码器。

多域编码的实施例无需限于特定类型的输入码元。通常，输入码元的值是从针对某一正数值  $M$  的  $2^M$  个码元的字母表中选出的。在这种情形下，输入码元可用来自输入文件或流送的数据的  $M$  个比特的序列来表示。 $M$  的值常基于例如应用的使用、通信信道、和/或输出码元的大小来确定。另外，输出码元的大小常基于应用、信道、和/或输入码元的大小来确定。在某些情形下，如果输出码元值和输入码元值具有相同的大小（即可由相同数目的比特来表示或选自同一字母表）则编码过程可以被简化。如果是这种情形，则当输出码元值大小有限时输入码元值大小也是有限的。例如，可能希望将输出码元放在有限大小的分组中。如果关于与输入码元相关联的键有关的某些数据要被传送以在接收机处恢复该键，则输出码元较优地小到足以在一个分组中容纳输出码元值以及关于该键的数据。

作为示例，如果输入文件是若干兆字节的文件，则输入文件可被分裂成数千、数万、或数十万个输入码元，其中每个输入码元编码数千、数百、或仅数个字节。作为另一示例，对于基于分组的因特网信道，具有 1024 字节大小的有效载荷的分组可能是合适的（1 字节为 8 比特）。在该示例中，假定每个分组包含一个输出码元和 8 字节的附属信息，8128 比特  $((1024-8)*8)$  的输出码元大小将是合适的。因此，输入码元大小可被选为  $M=(1024-8)*8$  或 8128 个比特。作为另一示例，某些视频分发系统使用 MPEG 分组标准，其中每个分组的有效载荷包括 188 个字节。在该示例中，假定每个分组包含一个输出码元和 4 字节的附属信息，1472 比特  $((188-4)*8)$  的输出码元大小将是合适的。因此，输入码元大小可被选为  $M=(188-4)*8$  或 1472 个比特。在使用多级编码的通用通信系统中，诸如输入码元大小（即，输入码元所编码的比特的数目  $M$ ）等应用专属参数可以是根据应用所设的变量。

作为另一示例，对于使用可变大小源分组发送的流送，码元大小可被选成

---

相当小，以使得每个源分组可用总大小至多略大于该源分组的整数个输入码元来覆盖。

每个输出码元具有一值。在以下我们所考虑的一优选实施例中，每个输出码元还具有与其相关联的标识符——被称为“键”。较优地，每个输出码元的键可被接收方容易地确定以允许接收方将一个输出码元从其它输出码元当中区别开。较优地，输出码元的键与所有其他输出码元的键不同。在现有技术中讨论了各种形式的键控。例如，Luby I 描述了可在本文所述的实施例中采用的各种形式的键控。

在有数据删除的预料或接收方没有正好在传输开始和结束时开始和结束接收的情形下，多域多级编码尤其有用。后者情形在本文被称为“数据不完整性”。关于删除事件，多级编码享有 Luby I 中所述的连锁反应码的许多益处。特别地，多级码可以是喷泉码、或无率码，在这种情形下，针对固定值的输入码元集可生成比输入码元多许多倍的不同输出码元，并且可使用任意合适数目 的不同输出码元来将输入码元恢复至所需的精确度。当使用了多域多级编码时，这些情形不会有害地影响通信过程，因为用多域多级编码生成的输出码元是信息加性的。例如，如果由于噪声突发而引起一百个分组丢失从而导致数据删除，则可在该突发之后拾取额外的一百个分组来代替所被删除的分组的丢失。如果由于接收机在发射机开始发射时没有调谐至该发射机而使得数千个分组丢失，则接收机可仅从任何其它传输期或甚至从另一发射机拾取这数千个分组。采用多域多级编码，接收机不局限于拾取任何特定的分组集合，所以它能从一个发射机接收某些分组，切换到另一发射机，丢失某些分组，错过给定传输的开始或结束，且仍然恢复输入文件或流送块。在无需接收机-发射机协调的情况下加入和离开传输的能力有助于简化通信过程。

在某些实施例中，使用多域多级编码传送文件或流送可包括从输入文件或流送块生成、形成或提取输入码元，计算冗余码元，将输入和冗余码元编码为一个或多个输出码元，其中每个输出码元是基于键独立于所有其它输出码元而生成的，以及在信道上向一个或多个接收方传送输出码元。另外，在某些实施例中，使用多域多级编码接收（并重构）输入文件或流送块的副本可包括从一个或多个数据流送接收输出码元的某些集或子集，并从接收到的输出码元的值

和键来解码输入码元。

本文所述的合适 FEC 删 除码可被用来克服上述困难并且将在包括多媒体广播和多播系统及服务的众多领域中找到用途。以下称为“多域多级连锁反应码”的 FEC 删 除码具有满足这些系统和服务的许多当前和将来要求的属性。

多域多级连锁反应码的某些基本属性是对于任何分组丢失情况和对于任何相关大小的源文件或任何相关速率的流送的递送：(a)每个个体接收机设备（“RD”）的接收开销被最小化；(b)将源文件递送至任意数目的 RD 所需的总传输时间可被最小化；(c)在选择合适的传输调度的情况下对于相对于输入码元的数目所发送的输出码元的数目能够使得递送至任意数目的 RD 的流送的质量被最大化。RD 可能是手持设备，被嵌入到车辆中、便携式的（即，可移动但在使用时通常不移动）或者被固定到一位置上。

解码所需的工作存储器的量较低且仍然能够提供以上属性，且编解码所需的计算量最小。在该文档中，我们为多域多级连锁反应码的某些变形提供了简易实现的描述。

多域多级连锁反应码是喷泉码，即可在过程中根据需要生成编码分组，其中每个包含对于恢复源文件或流送块同样有用的唯一编码码元。相比于其它类型的 FEC 码，使用喷泉码有许多优点。一个优点是不论分组丢失情况和 RD 可用性如何，喷泉码将每个 RD 用以重构源文件或流送块所需要接收的编码分组的数目最小化。这即使在较差的分组丢失情况下和当例如移动 RD 在较长的文件下载会话期间仅间歇地开启或可用时也是如此。

另一优点是正好生成所需要的数目个编码分组的能力，从而在传输正在进行时关于在过程中生成多少编码分组作出决策。这在例如有来自 RD 的指示它们是否接收到足以恢复源文件或流送块的反馈的情况下是有用的。当分组丢失情况比预期的较轻时，传输可以及早终止。当分组丢失情况比预期更严重或 RD 比预期更频繁地不可用时，传输可被无缝地延长。

另一优点是逆多路复用的能力。逆多路复用是在 RD 能够组合所接收到的在独立发送方处生成的编码分组以重构源文件或流送块时。逆多路复用的一个实际用途将在以下参照从不同发送方接收编码分组来进行描述。

在将来的分组丢失、RD 可用性以及应用状况难以预测的情况下，选择尽

可能灵活的 FEC 方案以在不可预测的状况下良好工作是非常重要的。多级连锁反应码提供了其它类型的 FEC 码所无法媲美的灵活性程度。

多域多级码的又一优点是该码的差错概率和差错基底比具有等价计算复杂度的先前已知的码低得多。同样，多域多级连锁反应码的计算复杂度比具有等价差错概率和/或差错基底的先前已知的码低得多。

多域多级连锁反应码的另一优点是诸如码元大小和域大小等参数可以灵活选择，以在计算复杂度与差错概率和/或差错基底之间实现任何所需的平衡。

### 系统概述

图 1 是使用多级编码的通信系统 100 的框图。在通信系统 100 内，向输入码元生成器 110 提供输入文件 101 或输入流 105。输入码元生成器 110 从输入文件或流生成一个或多个输入码元的序列( $IS(0), IS(1), IS(2), \dots$ )，每个输入码元有值和位置(在图 1 内用括号内的整数表示)。如上所述，对于输入码元的可能值，即其字母表一般是  $2^M$  个码元，使得每个输入码元编码为输入文件的 M 个比特。值 M 一般通过使用通信系统 100 确定，但通用系统可以包括输入码元生成器 110 的码元大小输入，使得 M 可以随每次使用而改变。输入码元生成器 110 的输出可以被提供给编码器 115。

静态键生成器 130 生成静态键  $S_0, S_1 \dots$  的流送。生成的静态键数一般是有有限的，且取决于编码器 115 的特定实施例。静态键的生成在以下将接着更详细地描述。动态键生成器 120 为每个要由编码器 115 生成的输出码元生成一动态键。每个动态键被生成使得同一输入文件或流送块的动态键的很大部分是唯一的。例如，Luby I 描述可以使用的键生成器的实施例。动态键生成器 120 和静态键生成器 130 的输出被提供给编码器 115。

从动态键生成器 120 提供的每个键 I，编码器 115 从输入码元生成器提供的输入码元生成输出码元，其值为  $B(I)$ 。编码器 115 的操作在以下将详述。每个输出码元的值是基于其键、一个或多个输入码元的某些函数以及可能从输入码元计算出的一个或多个冗余码元被生成。产生特定输出码元的输入码元和冗余码元的集合在此被称为输出码元的“关联码元”或就是其“关联”。函数(“值函数”)的选择以及相关是根据以下更详细描述的过程完成的。一般，但不总是，M 对于输入和输出码元是相同的，即它们都编码为

相同数目的比特。

在某些实施例中，输入码元数 K 被编码器 115 用于选择关联。如果 K 事先未知，诸如输入是流文件，则 K 可以只是一个估计。值 K 还可以被编码器 115 用于为输入码元和任何由编码器 115 生成的中间码元分配存储。

编码器 115 提供输出码元给发射模块 140。发射模块 140 还被提供了来自动态键生成器 120 的每个该种输出码元的键。发射模块 140 发送输出码元，且取决于使用的键方法，发送模块 140 可能在信道 145 上将某些关于发送的输出码元的键的某些数据发送到接收模块 150。信道 145 被假设为删除信道，但这不是通信系统 100 的合适操作的需要。模块 140、145 和 150 还可以是任何合适的硬件组件、软件组件、物理介质或其任何组合，只要发射模块 140 用于将输出码元和任何关于键需要的数据发送到信道 145，且接收模块 150 用于从信道 145 接收码元，以及潜在的某些关于其键的数据。K 的值如果被用于确定关联，则可以在信道 145 上被发送，或可以事先按照编码器 115 和解码器 155 的同意而被设定。

如上解释的，信道 145 可以是实时信道，诸如通过因特网的路径或从电视发射机到电视接收机或从一点到另一点的电话连接的广播链路，或信道 145 可以是存储信道，诸如 CD-ROM、盘驱动、万维网站等。信道 145 甚至可以是实时信道和存储信道的组合，诸如当个人将输入文件在电话线上从个人计算机发送到因特网服务提供商(ISP)时形成的信道，输入文件被存储在万维网服务器上，接着通过因特网被发送到接收者。

由于信道 145 被假设是删除信道，通信系统 100 不假设从接收模块 150 出来的输出码元和进入发射模块 140 的输出码元间的一对一对应。实际上，当信道 145 包括分组网络时，通信系统 100 可能甚至不能假设在通过信道 145 时保留了任何两个或多个分组的相对顺序。因此，使用一个或多个上述的键方案的输出码元的键被确定，且不需要按输出码元离开接收模块 150 的顺序确定。

接收模块 150 将输出码元提供给解码器 155，且任何数据接收模块 150 接收这些输出码元的键，这些键被提供给动态键再生器 160。动态键再生器 160 再生接收到的输出码元的动态键并将这些动态键提供给解码器 155。静

态键生成器 163 再生静态键  $S_0, S_1, \dots$  并将其提供给解码器 155。静态键生成器访问在编码和解码过程中都使用的随机数生成器 135。如果随机数就在该种设备上生成，则这可以是以对该相同物理设备访问的形式，或是以该相同算法访问的形式来用于生成随机数以获得相同的行为。解码器 155 使用动态键生成器 160 和静态键生成器 163 提供的键连同对应的输出码元以恢复输入码元(同样 IS(0),IS(1),IS(2),...)。解码器 155 将恢复的输入码元提供给输入文件重编器 165，该重编器生成输入文件 101 或输入流 105 的副本 170。

连锁反应编码器生成的输出码元的一个属性是接收机能够一旦已经接收到足够的输出码元就能够恢复原始文件或原始流送的块。特别地，为了以较高概率恢复原始 K 个输入码元，接收机需要近似  $K+A$  个输出码元。比率  $A/K$  被称为“相对接收开销”。相对接收开销取决于输入码元的数目、以及解码器的可靠性。Luby I、Luby II 以及 Shokrollahi I 提供了可在特定实施例中使用的系统和方法的教学。然而应该理解的是，这些系统和方法并不是本发明所必需的，也可使用许多其它变形、修改、或替换方案。

### 编码器

图 2 是图 1 内示出的编码器 115 的一个特定实施例框图。编码器 115 包括静态编码器 210、动态编码器 220 以及冗余计算器 230。静态编码器 210 接收以下输入：a)由输入码元生成器 110 提供并存储在输入码元缓冲器 205 内的原始输入码元( $IS(0), IS(1), \dots, IS(K-1)$ )； b)原始输入码元数 K； c)由静态键生成器 130 提供的静态键  $S_0, S_1, \dots$ ；以及 d)冗余码元数 R。在接收到这输入后，静态编码器 205 计算 R 个冗余码元  $RE(0), RE(1), \dots, RE(R-1)$ ，如下描述。一般但不总是冗余码元与输入码元有相同大小。在一特定实施例中，静态编码器 210 生成的冗余码元被存储在输入码元缓冲器 205 内。输入码元缓冲器 205 可以只是逻辑的，即文件或流送块可以物理地被存储在一个地方，而码元缓冲器 205 内的输入码元的位置可能只是原始文件或流送块内的这些码元位置的重命名。

动态编码器接收输入码元和冗余码元，且如以下详述地生成输出码元。在一实施例中，其中冗余码元被存储到输入码元缓冲器 205 内，动态编码

器 220 从输入码元缓冲器 205 接收输入码元和冗余码元。

冗余计算器 230 从输入码元的数目 K 计算冗余码元的数目 R。该计算在以下详述。

### 静态编码器概览

静态编码器 210 的一般操作参考图 3 和 4 示出。图 3 是说明静态编码方法的一实施例的简化流程图。在步骤 305，变量 j 跟踪已生成多少冗余码元，该值被设定为零。然后，在步骤 310，第一冗余码元 RE(0)作为输入码元 IS(0),...,IS(K-1)的至少某些的函数  $F_0$  被计算。然后在步骤 315，变量 j 递增。接着，在步骤 320，测试是否所有的冗余码元均被生成了(即 j 是否大于 R-1?)。如果是，则流程结束。否则，流程进行到步骤 325。在步骤 325，RE(j) 作为输入码元 IS(0),...,IS(K-1) 和先前生成的冗余码元 RE(0),...,RE(j-1) 的函数  $F_j$  被计算，其中  $F_j$  不需要是取决于每个输入码元或每个冗余码元的函数。步骤 315、320 和 325 经重复直到已计算了 R 个冗余码元。

再次回到图 1 和 2，在某些实施例中，静态编码器 210 从静态键生成器 130 接收一个或多个静态键  $S_0, S_1, \dots$ 。在这些实施例中，静态编码器 210 使用静态键以确定某些或所有的函数  $F_0, F_1, \dots, F_{R-1}$ 。例如，静态  $S_0$  可以用于确定函数  $F_0$ ，静态键  $S_1$  可以被用于确定函数  $F_1$  等。或一个或多个静态键  $S_0, S_1, \dots$  可以被用于确定函数  $F_0$ ，一个或多个静态键  $S_0, S_1, \dots$  可以用于确定函数  $F_1$  等。在其他实施例中，不需要静态键，因此不需要静态键生成器 130。

回到图 2 和 3，在某些实施例中，静态编码器 210 生成的冗余码元可以被存储在输入码元缓冲器 205 中。图 4 是静态编码器 210 的一实施例的操作的简化说明。尤其是，静态编码器 210 用从输入码元缓冲器 205 接收到的输入码元 IS(0),...,IS(K-1), RE(0),...,RE(j-1) 的函数  $F_j$  生成冗余码元 RE(j)，并将其存储回输入码元缓冲器 205。函数  $F_0, F_1, \dots, F_{R-1}$  的准确形式取决于特定应用。一般但不总是，函数  $F_0, F_1, \dots, F_{R-1}$  包括某些或所有它们对应的参变量的异或。如上描述，这些函数可以实际上采用或可不采用图 1 的静态键生成器 130 生成的静态键。例如，在以下描述的一个特定实施例，开始一些函数实现 Hamming 码且不使用任何静态键  $S_0, S_1, \dots$ ，而剩余函数实现低

密度奇偶校验编码并显式使用静态键。

### 多级编码器概览

再次参照图 2, 动态编码器 220 接收输入码元  $IS(0), \dots, IS(K-1)$  以及冗余码元  $RE(0), \dots, RE(R-1)$  以及要生成的每个输出码元的键  $I$ 。该集合包括原始输入码元及冗余码元, 此后会被称为“动态输入码元”集合。图 5 是动态编码器的一个实施例的简易框图, 包括权重选择器 510、关联器 515、值函数选择器 520 以及计算器 525。如图 5 所示,  $K+R$  个动态输入码元被存储在动态码元缓冲器 505 中。实际上, 动态编码器 500 执行图 6 中所示的动作, 即作为所选输入码元的某一值函数生成输出码元值  $B(I)$ 。

图 7 是静态编码器的一个特定实施例的简易框图。静态编码器 600 包括参数计算器 605、低密度奇偶校验 (LDPC) 编码器 610、以及高密度奇偶校验 (HDPC) 编码器 620。LDPC 编码器 610 被耦合用以从输入码元缓冲器 625 接收输入码元  $IS(0), \dots, IS(K-1)$ , 输入码元的数目  $K$ 、以及参数  $E$ 。作为响应, LDPC 编码器 610 根据 LDPC 码生成  $E$  个冗余码元  $LD(0), \dots, LD(E-1)$ 。接着, HDPC 编码器 620 被耦合以接收  $K+E$  个码元  $IS(0), \dots, IS(K-1), LD(0), \dots, LD(E-1)$  中的多个以及参数  $D$  来根据 HDPC 码生成  $D$  个冗余码元  $HA(0), HA(1), \dots, HA(D-1)$ 。

图 8 示出了采用图 7 中所示的静态编码器的一个实施例的操作。

图 9 是示出了诸如图 7 的参数计算器 605 的参数计算器的一个实施例的简易流程图, 当 HDPC 码为 Hamming 码时该参数计算器计算如上所述的参数  $D$  和  $E$ 。首先, 在步骤 705, 参数  $D$  被初始化为 1。然后, 在步骤 710, 确定  $2^D-D-1$  是否小于  $K$ 。如果为否, 则流程行进到步骤 730。如果为是, 则流程行进到步骤 720, 其中参数  $D$  递增。然后, 流程回到步骤 710。一旦  $D$  已经确定, 则在步骤 730 将参数  $E$  计算为  $R-D-1$ 。

图 10 是根据本发明的一个实施例的编码器的简易流程图, 现在将对其进行描述。首先在步骤 805, 变量  $i$  被初始化为 0。变量  $i$  跟踪已经生成的冗余码元的数目。在步骤 810, 数目  $t$  被计算为大于或等于  $K/2$  的最小奇数整数。在步骤 815, 基于  $K, t$ 、以及静态键  $S_i$  生成值  $P_1, P_2, \dots, P_t$ 。值  $P_1, P_2, \dots, P_t$  指示将被用来生成冗余码元的输入码元的位置。在一个特定实施例中, 诸

如图 5 的关联器 515 的关联器被用来生成  $P_1, P_2, \dots, P_t$ 。特别地，值  $t$  可被提供作为  $W(I)$  输入，值  $K$  可被提供作为  $K+R$  输入，而静态键  $S_i$  可被提供作为键  $I$  输入。应该注意，许多不同的  $t$  值将产生类似的编码效果，因此该特定选择仅是示例。在步骤 820， $RE(i)$  的值被计算为值  $IS(P_1), IS(P_2), \dots, IS(P_t)$  的 XOR。在步骤 825，变量  $i$  被递增 1 以准备下一冗余码元的计算，而在步骤 830，确定是否已生成所有的冗余码元。如果为否，则流程返回步骤 815。

图 11 是示出了根据本发明的解码器的一个实施例的简易框图。解码器 900 可被用于例如实现图 1 的解码器 155。

解码器 900 包括动态解码器 905 和静态解码器 910。动态解码器 905 恢复的输入码元和冗余码元被存储在重构缓冲器 915 中。一旦完成动态解码，静态解码器 910 尝试恢复动态解码器 905 所未恢复的任何输入码元（若有）。特别地，静态解码器 910 从重构缓冲器 915 接收输入码元和冗余码元。

图 12 是示出了根据本发明的用于解码的方法的一个实施例的简易流程图。在步骤 1005，解码器接收到  $Q$  个输出码元。 $Q$  的值可取决于输入码元的数目和所使用的该具体的动态编码器。 $Q$  的值还取决于解码器能够恢复输入码元所需达到的精确度。例如，如果希望解码器能够以较高概率恢复所有输入码元，则  $Q$  应被选成大于输入码元的数目。特别地，在特定实现中，当输入码元的数目较大时， $Q$  可以比原始输入码元的数目大不超过 3%。在其它应用中，当输入码元的数目较小时， $Q$  可以比输入码元的数目大至少 10%。特别地， $Q$  可被选为输入码元的数目  $K$  加上数目  $A$ ，其中  $A$  被选成确保解码器能够以较高概率再生所有输入码元。数目  $A$  的确定在以下具体描述。如果解码器不能解码所有输入码元（或者有时或者总是）是可接受的，则  $Q$  可以小于  $K+A$ 、等于  $K$ 、或甚至小于  $K$ 。显然，总体编码系统的一个目标将是尽可能地减小数目  $Q$ ，同时保持解码过程关于所需精确度的成功有较佳的概率保证。

在步骤 1010，动态解码器 905 从  $Q$  个接收到的输出码元再生输入码元和冗余码元。应该理解的是，步骤 1005 和 1010 可基本同时执行。例如，

动态解码器 905 可在解码器接收 Q 个输出码元之前再生输入码元和冗余码元。

在动态解码器 905 已经处理 Q 个输出码元之后，则确定输入码元是否已被恢复到所需的精确度。所需精确度可以是例如所有输入码元、或者某一数目或百分比地少于所有输入码元。如果是，则流程结束。如果否，则流程行进到步骤 1020。在步骤 1020，静态解码器 910 尝试恢复动态解码器未能恢复的任何输入码元。在静态编码器 910 已经处理动态编码器 905 所恢复的输入码元和冗余码元之后，则流程结束。

图 13 是根据本发明的用于解码的方法的另一实施例的简易流程图。该实施例类似于关于图 11 所述的实施例，并且包括相同步骤 1005、1010、1015、和 1025。但在步骤 1025 之后，流程行进到步骤 1030，其中确定输入码元是否已经被恢复到所需的精确度。如果是，则流程结束。如果否，则流程行进到步骤 1035。在步骤 1035，接收到一个或多个额外输入码元。然后，流程行进返回步骤 1010，从而使得动态解码器 905 和/或静态解码器 910 可尝试恢复剩余未恢复的输入码元。

图 14 是示出根据本发明的用于解码的又一实施例的简易流程图。在步骤 1055 中，解码器接收到输出码元，并且在步骤 1060，动态解码器 905 从接收到的输出码元再生输入码元和冗余码元。然后，在步骤 1065，确定动态解码是否应结束。该确定可基于经处理的输出码元的数目、所恢复的输入码元的数目、额外输入码元正被恢复的当前速率、处理输出码元所费的时间等中的一个或多个。

在步骤 1065，如果确定将不停止动态解码，则流程行进返回到步骤 1055。但如果在步骤 1065 确定要结束动态解码，则流程行进到步骤 1070。在步骤 1070，确定输入码元是否已被恢复到所需的精确度。如果是，则流程结束。如果否，则流程行进到步骤 1075。在步骤 1075，静态解码器 910 尝试恢复动态解码器 905 未能恢复的任何输入码元。在静态编码器 910 已经处理动态编码器 905 所恢复的输入码元和冗余码元之后，流程结束。

图 15 示出了根据本发明的动态解码器的一个实施例。动态解码器 1100 包括如图 5 中所示的动态编码器 500 的相似组件。解码器 1100 类似于 Luby

I 和 Luby II 中所述连锁反应解码器的实施例。动态解码器 1100 包括权重选择器 510、关联器 515、值函数选择器 520、输出码元缓冲器 1105、减缩器 1115、重构器 1120 以及重构缓冲器 1125。

图 16 是示出了静态解码器的一个实施例的简易框图。该实施例可在数据用诸如参照图 7 所述的静态编码器编码时使用。静态解码器 1200 包括 LDPC 解码器 1205 和 Hamming 解码器 1210。LDPC 解码器 1205 从重构缓冲器 1215 接收输入码元和冗余码元，并重试恢复重构缓冲器 1215 当中在动态解码器的解码步骤之后未能恢复的那些码元。在某些实施例中，重构缓冲器 1215 是重构缓冲器 1125（图 15）。

LDPC 解码器和 HDPC 解码器的许多变形是本领域的技术人员已知的，并且可在根据本发明的各种实施例中采用。在一个特定实施例中，HDPC 解码器是使用高斯消去算法实现的。高斯消去算法的许多变形是本领域的技术人员所公知的，并且可在根据本发明的各种实施例中采用。

#### HDPC 编码的变形

现在描述另一类型的 HDPC 编码。在该 HDPC 编码的实施例中，用于从给定数据集创建冗余码元的数学运算是基于有限域中的运算。

在该 HDPC 编码的实施例中，有限域的元素被用来获得冗余码元  $HD[0], \dots, HD[D-1]$ 。这些码元是通过如上所述地在码元  $IS[0], \dots, IS[K-1], LD[0], \dots, LD[E-1]$  与有限域的元素之间定义乘法过程获得的。

#### HDPC 编码

当使用 HDPC 编码时，该码可通过有限域  $GF(2^M)$  上的生成矩阵来描述。在码是系统性的时，在优选实施例中即这种情形，生成矩阵可以仅使用  $K+E$  个输入码元  $IS[0], \dots, IS[K-1], LD[0], \dots, LD[E-1]$  与冗余码元  $HD[0], \dots, HD[D-1]$  之间的关系来描述。这一被称为  $G$  的矩阵其形式为  $Dx(K+E)$ 。如果  $X$  标示包括码元  $HD[0], \dots, HD[D-1]$  的列向量且  $S$  标示包括码元  $IS[0], \dots, IS[K-1], LD[0], \dots, LD[E-1]$  的列向量，则得到  $X = G \otimes S$ 。以下描述矩阵  $G$  的更具体的实施例以及用于高效计算码元的各种方法。

#### 变形

上述多级连锁反应码不是系统性码，即源块的所有原始源码元不一定要在发送的编码码元当中。然而，系统性 FEC 码对于文件下载系统或服务是有用的，且对于流送系统或服务非常重要。如以下实现中所示的，一种经过修改的码可被制成系统性的且仍然保持喷泉码和其它所述属性。

为何容易使用多级码构造各种补充服务的一个原因在于它可以组合从多个发送方接收到的编码码元以重构源文件或流送而无需各发送方之间的协调。唯一要求是发送方使用不同的键集来生成它们在编码分组中向该码发送的编码码元。实现这一点的方法包括指定不同范围的键空间以供每个发送方使用，或在每个发送方处随机地生成键。

作为这种能力使用的示例，考虑向文件下载服务提供一种补充服务，以允许没有接收到足够的编码分组的多级连锁反应码，可从文件下载会话请求例如经由 HTTP 会话从制作发送方发送附加的编码分组，以重构源文件。制作发送方从源文件生成编码码元并例如使用 HTTP 发送它们，且所有这些编码码元可与从文件下载会话接收的编码码元相组合以恢复源文件。使用该方法允许不同的发送方提供递增的源文件递送服务而无需各发送方之间的协调，并确保每个个体接收机仅需要接收恢复每个源文件所要的最少数目的编码分组。

在源码元的数目较小时——例如在数百到几千个源码元的量级时，如上所述的多级连锁反应码的解码可要求相对较大的开销。在这种情形下，一种不同的解码器是较优的，诸如 Shokrollahi III 中所公开的解码器。如以下实现所示，可以为本文所公开的这类码——使用 Shokrollahi III 中公开的码特征和概念并在维持解码效率的同时为非常小数目的源码元提供了低解码误差概率——设计了一种经修改的解码算法。

### 多域多级码的各级的实现

#### FEC 方案定义

使用这些技术的分组可用诸如包括源块编号 (SBN)（该分组内的编码码元相关的源块的 16 比特整数标识符）和编码码元 ID (ESI)（该分组内的编码码元的 16 比特整数标识符）的四个八位位组的 FEC 有效载荷 ID 等报头信息来表示。源块编号和编码码元标识符的一种合适解释在以下 B 节中定义。FEC 对象传输信息可包括 FEC 编码 ID、传送长度 ( $F$ ) 以及以下定义的参数  $T$ 、 $Z$ 、

$N$  和  $A$ 。参数  $T$  和  $Z$  是 16 比特无符号整数,  $N$  和  $A$  是 8 比特无符号整数。根据需要, 可使用其它整数。

前向纠错的 FEC 编码方案在以下章节定义。其定义了两种不同的 FEC 有效载荷 ID 格式, 一种用于 FEC 源分组以及另一种用于 FEC 修复分组, 但非系统性码的变形也是可能的。

源 FEC 有效载荷 ID 可包括源块编号 (SBN) (该分组内的编码码元相关的源块的 16 比特整数标识符) 和编码码元 ID (ESI) (该分组内的编码码元的 16 比特整数标识符), 而修复 FEC 有效载荷 ID 可包括源块编号 (SBN) (该分组内的修复码元相关的源块的 16 比特整数标识符)、编码码元 ID (ESI) (该分组内的修复码元的 16 比特整数标识符)、以及源块长度 (SBL) (16 比特, 表示源块中源码元数目)。源块编号、编码码元标识符以及源块长度的解释在以下定义。

FEC 对象传输信息可包括 FEC 编码 ID、以码元计的最大源块长度、和以字节计的码元大小。码元大小和最大源块长度可包括四个八位位组——码元大小 ( $T$ ) (16 比特, 表示以字节计的编码码元的大小的) 和最大源块长度 (16 比特, 表示以码元计的源块最大长度的)。

以下章节规定了系统性多域 MSCR 前向纠错码。多域 MSCR 码是喷泉码, 即编码器可在过程中从块的源码元生成如所需要的数目的编码码元。解码器能够从仅略多于源码元数目的编码码元集恢复出源块。本文档中描述的码是系统性码, 即原始源码元未经修改的与一定数目的修复码元被从发送方发送到接收方。

## B.1 定义、码元和缩写

### B.1.1 定义

出于描述的目的, 应用以下术语和定义。

**源块:** 被一起考虑以作 MSCR 编码之用的  $K$  个源码元的块。

**源码元:** 在编码过程中使用的最小数据单元。源块内的所有源码元具有相同大小。

**编码码元:** 包括在数据分组中的码元。编码码元包括源码元和修复码元。从源块生成的修复码元具有与源块的源码元相同的大小。

**系统性码：**其中源码元被包括作为针对源块所发送的编码码元的一部分的码。

**修复码元：**针对源块所发送的编码码元当中不是源码元的那些码元。修复码元是基于源码元生成的。

**中间码元：**使用逆编码过程从源码元生成的码元。随后直接从中间码元生成修复码元。编码码元不包括中间码元，即中间码元不包括在数据分组中。

**码元：**数据单元。以字节计的码元的大小被称为码元大小。

**编码码元组：**一起发送的一组编码码元，即同一分组内其与源码元的关系可从单个编码码元 ID 导出的一组编码码元。

**编码码元 ID：**定义编码码元组的各码元与源码元之间的关系的信息。

**编码分组：**包含编码码元的数据分组

**子块：**源块有时被分裂为数个子块，其中每一个子块小到足以在工作存储器中被解码。对于包括  $K$  个源码元的源块，每个子块包括  $K$  个子码元，源块的每个码元由来自每个子块的一个子码元构成。

**子码元：**码元的一部分。源块中有多少个子块，每个源码元就由多少个子码元构成。

**源分组：**包含源码元的数据分组。

**修复分组：**包含修复码元的数据分组。

### B.1.2. 码元

$i, j, x, h, a, b, d, v, m$	表示正整数
$\text{ceil}(x)$	标示大于或等于 $x$ 的最小正整数
$\text{choose}(i, j)$	标示可从 $i$ 个对象当中选出 $j$ 个对象而不重复的方法的数目
$\text{floor}(x)$	标示小于或等于 $x$ 的最大正整数
$i \% j$	标示 $i \bmod j$
$X \wedge Y$	标示，对于等长的比特串 $X$ 和 $Y$ ， $X$ 和 $Y$ 的逐位异或
$A$	标示码元对准参数。码元和子码元大小限于 $A$ 的倍数
$\mathbf{A}^T$	标示矩阵 $\mathbf{A}$ 的转置
$\mathbf{A}^{-1}$	标示矩阵 $\mathbf{A}$ 的逆矩阵
$K$	标示单个源块中的码元数目

$K_{MAX}$	标示能够在单个源块中的最大码元数目。设为 8192。注意可使用其它值。
$L$	标示单个源块的预编码码元的数目
$S$	标示单个源块的 LDPC 码元的数目
$H$	标示单个源块的一半码元的数目
$\mathbf{C}$	标示中间码元数组 $C[0], C[1], C[2], \dots, C[L-1]$
$\mathbf{C}'$	标示源码元数组 $C'[0], C'[1], C'[2], \dots, C'[K-1]$
$X$	非负整数值
$V_0, V_1$	两个 4 字节整数数组 $V_0[0], V_0[1], \dots, V_0[255]; V_1[0], V_1[1], \dots, V_1[255]$
$\text{Rand}[X, i, m]$	伪随机数生成器
$\text{Deg}[v]$	幂度生成器
$\text{LTEnc}[\mathbf{K}, \mathbf{C}, (d, a, b)]$	LT 编码码元生成器
$\text{Trip}[K, X]$	三元组生成器函数
$G$	编码码元组内码元的数目
$N$	源块内子块的数目
$T$	按比特计的码元大小。如果源块被分割为子块，则 $T=T' \cdot N$
$T'$	按比特计的子码元大小。如果源块未被分割为子块，则 $T'$ 是无关的。
$F$	按比特计的文件下载的大小
$I$	按比特计的子块的大小
$P$	对于文件下载，按比特计的每个分组的有效载荷大小，被用在文件下载传输参数的一个较优推导中。对于流送，按比特计的每个修复分组的大小，被用在流送传输参数的一个较优推导中。
$Q$	$Q = 65521$ , 即 $Q$ 小于 $2^{16}$ 的最大质数。注意也可使用其它值代替 $2^{16}$ 。
$Z$	对于文件下载，源块的数目
$J(K)$	与 $K$ 相关联的系统性索引
$\mathbf{G}$	标示任何生成器矩阵
$\mathbf{I}_S$	标示 $S \times S$ 单位矩阵

$\mathbf{0}_{S \times H}$	标示 $S \times H$ 零矩阵
---------------------------	---------------------

### B.1.3 缩写

出于本文献的目的，应用以下缩写：

ESI：编码码元 ID

LDPC：低密度奇偶校验

LT：Luby 变换

SBN：源块编号

SBL：源块长度（以码元为单位）

### B.2. 概览

MSCR 前向纠错码可应用于文件递送和流送应用两者。专属于这些应用的每一个的 MSCR 码特征在本文的 B.3 和 B.4 节中讨论。

系统性 MSCR 码的组件是 B.5 节中描述的基本编码器。首先，描述如何从原始源码元导出中间码元集的值以使得中间码元的知识足以重构源码元。其次，编码器产生各自为数个中间码元的异或的修复码元。编码码元是源码元和修复码元的组合。修复码元是以使得中间码元以及由此源码元能够从任何充分大的编码码元集恢复的方式产生的。

本文定义了系统性 MSCR 码编码器。众多可能的解码算法是可能的。B.6 节提供了高效解码算法。

中间码元和修复码元的构造部分地基于 B.5 中所述的伪随机数生成器。该生成器是基于发送方和接收方都可得到的 512 个随机数的固定集合。附录 B.1 和 B.2 中提供的是这些数的示例集合。

最后，从源码元构造中间码元是由“系统性索引”掌控的。在附录 A 中针对从 4 个源码元到  $K_{MAX}=8192$  个源码元的源块大小示出了系统性索引的值的示例集合。

### B.3. 文件下载

#### B.3.1. 源块构造

##### B.3.1.1. 概要

为了对源文件应用 MSCR 编码器，文件可被分裂为  $Z \geq 1$  个块，被称为源块。MSCR 编码器独立于每个源块被应用。每个源块由唯一的整数源块编号

(SBN) 来标识，其中第一源块具有 SBN 为零，第二源块具有 SBN 为一，以此类推。每个源块被分为各自大小为  $T$  个字节的  $K$  个源码元。每个源码元由唯一的整数编码码元标识符 (ESI) 标识，其中源块的第一源码元具有 ESI 零，第二源码元具有 ESI 一，以此类推。

具有  $K$  个源码元的每个源块被分为小到足以在工作存储器中被解码的  $N \geq 1$  个子块。每个子块被分为大小为  $T'$  的  $K$  个子码元。

注意  $K$  的值不一定对于文件的每个源块都相同，且  $T'$  的值对于源块的每个子块并不一定相同。然而，码元大小  $T$  对于文件的所有源块是相同的，且码元的数目  $K$  对于源块的每个子块是相同的。以下 B.3.1.2 节描述了将文件精确分割成源块和子块。

图 17 示出了放在二维数组中的示例源块，其中每项是  $T'$  字节的子码元，每行是子块且每列是源码元。在该示例中， $T'$  的值对于每个子块相同。每个子码元项中所示的数字指示它们在源块内的原始次序。例如，编号为  $K$  的子码元包含源块的字节  $T' \cdot K$  到  $T' \cdot (K+1)-1$ 。然后，源码元  $i$  是来自子块中每一个的第  $i$  个子码元的级联，这些子码元对应于源块当中编号为  $i, K+i, 2 \cdot K+i, \dots, (N-1) \cdot K+i$ 。

### B.3.1.2 源块和子块分割

源块和子块的构造是基于 5 个输入参数  $F$ 、 $A$ 、 $T$ 、 $Z$  和  $N$  以及函数 Partition (分割) [] 确定的。这五个输入参数定义如下：

- $F$  按比特计的文件大小
- $A$  按比特计的码元对准参数
- $T$  按比特计的码元大小——优选地为  $A$  的倍数
- $Z$  源块的数目
- $N$  每个源块中子块的数目

这些参数可被设成使得  $\text{ceil}(\text{ceil}(F/T)/Z) \leq K_{MAX}$ 。这些参数的一些适当推导的示例在 B.3.4 节中给出。

函数 Partition[] 取用一对整数  $(I, J)$  作为输入并导出四个整数  $(I_L, I_S, J_L, J_S)$  作为输出。具体而言，Partition[ $I, J$ ] 的值是四个整数  $(I_L, I_S, J_L, J_S)$  的序列，其中  $I_L = \text{ceil}(I/J)$ ， $I_S = \text{floor}(I/J)$ ， $J_L = I - I_S \cdot J$ ，以及  $J_S = J - J_L$ 。Partition[] 导出用于将大小为  $I$  的块分割成  $J$  个近似相等大小的块。具体而言，长度为  $I_L$  的  $J_L$  个块和

长度为  $I_S$  的  $J_S$  个块。

源文件可被如下分割成源块和子块：

令，  
 $K_t = \text{ceil}(F/T)$

$(K_L, K_S, Z_L, Z_S) = \text{Partition}[K_t, Z]$

$(T_L, T_S, N_L, N_S) = \text{Partition}[T/A, N]$

则，文件可被分割成  $Z = Z_L + Z_S$  个连续源块，前  $Z_L$  个源块各自具有长度  $K_L \cdot T$  个字节，剩余  $Z_S$  个源块各自具有  $K_S \cdot T$  个字节。

如果  $K_t \cdot T > F$ ，则出于编码目的，最后的码元可在末尾处用  $K_t \cdot T - F$  个零字节填充。

接着，每个源块可被划分为  $N = N_L + N_S$  个连续子块，前  $N_L$  个子块各自包括大小为  $T_L \cdot A$  的  $K$  个连续子码元，而剩余的  $N_S$  个子块各自包括大小为  $T_S \cdot A$  的  $K$  个连续子码元。码元对准参数  $A$  确保各子码元总是为  $A$  个字节的倍数。

最后，源块的第  $m$  个码元包括来自  $N$  个子块中每一个的第  $m$  个子码元的级联。

### B.3.2. 编码分组构造

#### B.3.2.1. 概要

每个编码分组包含源块编号（SBN）、编码码元 ID（ESI）和编码码元。每个源块相互间独立地进行编码。各源块从零连续编号。从 0 到  $K-1$  的编码码元 ID 值标识各源码元。从  $K$  向前的编码码元 ID 标识修复码元。

#### B.3.3.2. 编码分组构造

每个编码分组较优地或者包含源码元（源分组）或包含修复码元（修复分组）。分组可包含来自同一源块的任意数目的码元。在分组中的最后一个码元包括出于 FEC 编码目的而添加的填充字节的情况下，这些字节无需被包括在该分组中。否则，可包括仅所有码元。

每个源分组中所携带的编码码元 ID， $X$ ，是该分组中所携带的第一个源码元的编码码元 ID。分组中的后续源码元顺序地具有编码码元 ID， $X+1$  到  $X+G-1$ ，其中  $G$  是该分组中码元的数目。

类似地，被置于修复分组中的编码码元 ID， $X$ ，是该修复分组中第一个修复码元的编码码元 ID，且该分组中后续修复码元顺序地具有编码码元 IDs， $X+1$

到  $X+G-1$ , 其中  $G$  是该分组中码元的数目。

注意接收方无需知道修复码元的总数。被置于具有 ESI  $X$  的修复分组中的修复码元的  $G$  个修复码元三元组  $(d[0], a[0], b[0]), \dots, (d[G-1], a[G-1], b[G-1])$  是使用 B.5.3.4 中定义的三元组生成器如下计算出的：

对于每个  $i = 0, \dots, G-1$

$$(d[i], a[i], b[i]) = \text{Trip}[K, X+i]$$

将被置于具有 ESI  $X$  的修复分组中的  $G$  个修复码元是使用中间码元 **C** 和 LT 编码器  $\text{LTenc}[K, \mathbf{C}, (d[i], a[i], b[i])]$  如 B.5.3 节中所述地基于修复码元三元组计算出的。

### B.3.3. 传输

该节描述 MSCR 编码器/解码器与利用 MSCR 前向纠错进行文件递送的任何传输协议之间的信息交换。

用于文件递送的 MSCR 编码器和解码器要求来自传输协议的以下信息：以字节计的文件大小  $F$ 、码元对准参数  $A$ 、为  $A$  的倍数且以字节计的码元大小、源块数目  $Z$ 、每个源块中子块的数目  $N$ 。用于文件递送的 MSCR 编码器另外要求被编码的  $F$  个字节的文件。

MSCR 编码器向传输协议提供编码分组信息，包括针对每个分组的 SBN、ESI 以及编码码元。传输协议可将该信息透明地传达给 MSCR 解码器。

### B.3.4. 参数具体示例的细节

#### B.3.4.1 参数导出算法

该节提供用于四个传输参数——提供良好结果的  $A$ 、 $T$ 、 $Z$  和  $N$ ——的导出的示例。这些是基于以下输入参数：

$F$  以字节计的文件大小

$W$  以字节计的子块大小的目标指标

$P$  以字节计的最大的分组有效载荷大小——被假定为  $A$  的倍数

$A$  以字节计的码元对准因子

$K_{MAX}$  每源块的最大源码元数目

$K_{MIN}$  每源块码元数目的最小目标指标

$G_{MAX}$  每分组的码元数目的最大目标指标

基于以上输入，传输参数  $T$ 、 $Z$  和  $N$  被计算如下：

令，

$G = \min\{\text{ceil}(P \cdot K_{MIN}/F), P/A, G_{MAX}\}$  – 每分组的近似码元数目

$T = \text{floor}(P/(A \cdot G)) \cdot A$

$K_t = \text{ceil}(F/T)$  – 文件中的码元总数

$Z = \text{ceil}(K_t / K_{MAX})$

$N = \min\{\text{ceil}(\text{ceil}(K_t/Z) \cdot T/W), T/A\}$

以上导出的  $G$  和  $N$  的值应被考虑为下限。这对于将这些值增加至例如最近的 2 的幂是有利的。特别地，以上算法不保证码元大小  $T$  除尽最大分组大小  $P$ ，所以可能不能够使用正好为  $P$  大小的分组。而如果  $G$  被选择为除尽  $P/A$  的值，则码元大小  $T$  将是  $P$  的约数，且可使用大小为  $P$  的分组。

输入参数的合适值可以为  $W=256$  KB、 $A=4$ 、 $K_{MIN}=4$ 、以及  $G_{MAX}=1$ 。

#### B.3.4.2 示例

以上算法导致如图 18 中所示的传输参数，假定上述  $W$ 、 $A$ 、 $K_{MIN}$ 、 $G_{MAX}$  的值与  $P=512$  一起使用。

### B.4. 流送

#### B.4.1. 源块构造

源块是由例如本文中所定义的传输协议来构造的，以利用系统性 MSCR 前向纠错码。将用于源块构造和修复码元构造的码元大小  $T$  是由传输协议提供的。参数  $T$  可被设成使得任何源块中的源码元数目最大为  $K_{MAX}$ 。

工作良好的参数的示例在 B.4.4 中呈现。

#### B.4.2. 编码分组构造

如 B.4.3 中所述，每个修复分组包含 SBN、ESI、SBL 以及修复码元。修复分组内所含的修复码元的数目是从分组长度计算出的。置于各修复分组中的 ESI 值和用以生成修复码元的修复码元三元组是如 B.3.2.2 节中计算出的。

#### B.4.3. 传输

该节描述 MSCR 编码器/解码器与利用 MSCR 前向纠错进行流送的任何传输协议之间的信息交换。用于流送的 MSCR 编码器可使用来自传输协议的关于每个源块的以下信息：以字节计的码元大小  $T$ 、源块中码元的数目  $K$ 、源块编

号 (SBN) 以及要编码的源码元  $K \cdot T$  个字节。MSCR 编码器向传输协议提供编码分组信息，包括针对每个分组的 SBN、ESI、SBL 以及修复码元。传输协议可将该信息透明地传达给 MSCR 解码器。

#### B.4.4. 参数选择

可使用多种参数选择方法。以下具体示出了其中一些。

##### B.4.4.1 参数导出算法

该节解释了基于以下输入参数的传输参数  $T$  的导出：

$B$	以字节计的最大源块大小
$P_{max}$	无填充情况下的最大源分组信息大小
$P_x$	无填充情况下的第 $x$ 个百分点源分组信息大小（即使得分组的 $x\%$ 预期具有源分组信息大小 $n$ 或更少。在一个实施例中， $x$ 的值为 30）。
$A$	以字节计的码元对准因子
$K_{MAX}$	每源块的最大源码元数目
$K_{MIN}$	每源块的码元数目的最小目标指标
$G_{MAX}$	每修复分组的码元数目的最大目标指标

对这些输入的要求是  $\text{ceil}(B/P) \leq K_{MAX}$ 。基于以上输入，传输参数  $T$  被计算如下：

令  $G = \min\{\max\{\text{ceil}(P \cdot K_{MIN}/B), \text{floor}(P_x/P_{max})\}, P/A, G_{MAX}\}$ —每 SPI 的码元数目

$$T = \text{floor}(P/(A \cdot G)) \cdot A$$

以上导出的  $T$  值应被考虑为所使用的真实  $T$  值的向导。这对于确保  $T$  除尽  $P$  是有利的，或者当完整大小的修复码元被用于恢复丢失的源分组末尾处的部分源码元时，将  $T$  的值设置得更小是有利最小化损耗的（只要源块中的最大源码元数目不超过  $K_{MAX}$ ）。此外， $T$  的选择可取决于源分组大小分配，例如如果所有分组为相同大小则这对于选择  $T$  以使得修复分组  $P'$  的实际有效载荷大小（其中  $P'$  是  $T$  的倍数）等于（或大尽可能少的字节）每个源分组在源块中所占的字节数目是有利的。

输入参数的合适值可以为  $A=16$ 、 $K_{MIN}=4$ 、以及  $G_{MAX}=4$ 。

##### B.4.4.2 示例

以上算法导致如图 19 所示的传输参数，假定上述  $A$ 、 $K_{MIN}$ 、和  $G_{MAX}$  的值以及假定  $P=1424$ 。

## B.5. 系统性多域 MSCR 编码器

### B.5.1. 编码概览

系统性 MSCR 编码器被用来从包括  $K$  个源码元的源块生成修复码元。

码元是编解码过程的基本数据单元。对于每个源块（子块），所有码元（子码元）大小相同。针对编解码对码元（子码元）执行的基本操作是异或操作。

令  $C[0], \dots, C[K-1]$  标示  $K$  个源码元。

令  $C[0], \dots, C[L-1]$  标示  $L$  个中间码元。

编码的第一步是从这  $K$  个源码元生成  $L>K$  个中间码元。在该步骤中， $K$  个源三元组  $(d[0], a[0], b[0]), \dots, (d[K-1], a[K-1], b[K-1])$  是使用如 B.5.4.4 节中所述的  $Trip[]$  生成器来生成的。这  $K$  个源三元组与这  $K$  个源码元相关联并随后被用来使用逆编码过程从源码元确定  $L$  个中间码元  $C[0], \dots, C[L-1]$ 。该过程可通过 MSCR 解码过程来实现。

特定的“预编码关系”较优地在这  $L$  个中间码元内成立。B.5.2 节描述了这些关系以及中间码元是如何从源码元生成的。

一旦已经生成中间码元，则修复码元被生成并且一个或多个修复码元作为一组被置于单个数据分组中。每个修复码元组与编码码元 ID (ESI) 和  $G$  个编码码元相关联。ESI 被用来再次使用 B.5.4.4 节中所述的  $Trip[]$  生成器来为每个修复码元生成三个整数的三元组  $(d, a, b)$ 。这是使用 B.5.4 节中所述的生成器如 B.3 和 B.4 节中所述而实现的。然后，每个  $(d, a, b)$  三元组被用来使用 B.5.4.3 中所述的  $LTEnc[K, C[0], \dots, C[L-1], (d, a, b)]$  生成器从中间码元生成相应的修复码元。

### B.5.2. 第一编码步骤：中间码元生成

#### B.5.2.1 概要

第一编码步骤是从源码元  $C[0], \dots, C[K-1]$  生成  $L$  个中间码元  $C[0], \dots, C[L-1]$  的预编码步骤。中间码元是由两组约束唯一地定义的：

1. 中间码元通过一组源码元三元组与源码元相关。源码元三元组的生成是在 B.5.2.2 节中使用 B.5.4.4 节中所述的  $Trip[]$  生成器来定义的。
2. 一组预编码关系在中间码元自身内成立。

这些是在 B.5.2.3 节中定义的。 $L$  个中间码元的生成是随后在 5.2.4 节中定义的。

### B.5.2.2 源码元三元组

这  $K$  个源码元中的每一个与三元组  $(d[i], a[i], b[i])$  相关联，其中  $0 \leq i \leq K$ 。源码元三元组是使用 B.5.4.4 节中定义的三元组生成器如下确定的：

对于每个  $i$ ,  $0 \leq i < K$

$$(d[i], a[i], b[i]) = \text{Trip}[K, i]$$

### B.5.2.3 预编码关系

这  $L$  个中间码元之间的预编码关系是通过根据前  $K$  个中间码元来表示后  $L-K$  个中间码元定义的。

后  $L-K$  个中间码元  $C[K], \dots, C[L-1]$  包括  $S$  个 LDPC 码元以及  $H$  个 HDPC 码元。 $S$  和  $H$  的值是如下所述地从  $K$  确定的。则  $L=K+S+H$ 。

令

$X$  使得  $X \cdot (X-1) \geq 2 \cdot K$  的最小正整数。

$S$  使得  $S \geq \lceil 0.01 \cdot K \rceil + X$  的最小质数

$H$  使得  $\text{choose}(H, \lceil H/2 \rceil) \geq K + S$  的最小整数

$H' = \lceil H/2 \rceil$

$L = K + S + H$

$C[0], \dots, C[K-1]$  标示前  $K$  个中间码元

$C[K], \dots, C[K+S-1]$  标示  $S$  个 LDPC 码元，初始化为零

$C[K+S], \dots, C[L-1]$  标示  $H$  个 HDPC 码元，初始化为零

该  $S$  个 LDPC 码元被定义为以下过程结束时  $C[K], \dots, C[K+S-1]$  的值：

For  $i=0, \dots, K-1$  do

$a = 1 + (\text{floor}(i/S) \% (S-1))$

$b = i \% S$

$C[K+b] = C[K+b] \wedge C[i]$

$b = (b + a) \% S$

$C[K+b] = C[K+b] \wedge C[i]$

$b = (b + a) \% S$

$C[K+b] = C[K+b] \wedge C[i]$

对于  $H$  个 HDPC 码元的构造，系统使用域 GF(256)。该域可用关于域 GF(2) 上的不可约多项式  $f = x^8 + x^4 + x^3 + x^2 + 1$  来表示。令  $\alpha$  标示元素  $x$  对  $f$  取模。如本领域的普通技术人员所公知的，元素是基本的，即  $\alpha$  的 255 个一次方与 255 个 GF(256) 的 255 个非零元素一致。在一个实施例中，系统选择  $K+S$  个整数  $a[0], \dots, a[K+S-1]$ ，且用  $\beta[0], \dots, \beta[K+S-1]$  标示元素  $\alpha^{a[0]}, \dots, \alpha^{a[K+S-1]}$ 。此外，选择另外  $H$  个整数  $b[0], \dots, b[H-1]$  并用  $\Gamma[0], \dots, \Gamma[H-1]$  标示元素  $\alpha^{b[0]}, \dots, \alpha^{b[H-1]}$ 。本发明的其它优选实施例将指定这些整数的特定选择。然而，应该注意这些整数有许多等价选择。令对于所有正整数  $i$ ， $g[i] = i \wedge (\text{floor}(i/2))$ 。注意  $g[i]$  是 Gray 序列，其中每个元素与前一个元素在单个比特位置上有所不同。此外，令  $g[j,k]$  标示其元素在二进制表示中正好有  $k$  个非零比特的序列  $g[i]$  的第  $j$  个元素， $j=0,1,2,\dots$ 。如本领域的技术人员所公知的，序列  $g[j,k]$  具有这样的属性，即  $g[j,k]$  和  $g[j+1,k]$  的二进制表示在精确地两个位置中有所不同。我们用  $p[j,k,1]$  和  $p[j,k,2]$  来标示这些位置。

HDPC 码元的值被定义为在以下过程之后  $C[K+S], \dots, C[L-1]$  的值。

将码元  $U$  初始化为 0。该码元的大小与源码元、LDPC 码元、以及 HDPC 码元的公共大小相同。

接着，对于从 0 到  $K+S-2$  的变量  $h$ ，执行以下：变量  $U$  被更新为  $U = U * \beta[h] \wedge C[h]$ 。同时，设定  $C[K+S+p[j,H',1]] = C[K+S+p[j,H',1]] \wedge U$ ，以及  $C[K+S+p[j,H',2]] = C[K+S+p[j,H',2]] \wedge U$ 。

在另一步骤中，将  $U$  变形为  $U * \beta[K+S-1] \wedge C[K+S-1]$ 。

接着对于从 0 到  $H-1$  的变量  $h$ ，更新  $C[K+S+h] = C[K+S+h] \wedge \Gamma[h] * U$ 。这完成了 HDPC 编码过程的描述。

在优选实施例中，系统选择以下整数  $a[0], \dots, a[K+S-1]$  和  $b[0], \dots, b[H-1]$ ：  
 $a[0]=a[1]=\dots=a[K+S-1]=1$  以及  $b[0]=1, b[1]=2, \dots, b[i]=i+1$  等。有利地，在该优选实施例中，HDPC 码元的构造可仅使用基本元素  $\alpha$  的动作以及伴随码元之间的逐位异或操作来执行。以上给出的不可约多项式的选择允许  $\alpha$  的动作的高效实现，由此降低 HDPC 构造算法的计算复杂度。如本领域的技术人员所显而易见的，上述构造算法可容易地适用于执行多级码解码器内的所需解码操作，由此在解码器处也实现上述计算复杂度的降低。

### B.5.2.4 中间码元

#### B.5.2.4.1 定义

给定  $K$  个源码元  $C[0], C[1], \dots, C[K-1]$ ,  $L$  个中间码元  $C[0], C[1], \dots, C[L-1]$  是满足下列条件的唯一定义的码元值:

1 .  $K$  个源码元  $C[0], C[1], \dots, C[K-1]$  满足  $K$  个约束  
 $C[i] = \text{LTEnc}[K, (C[0], \dots, C[L-1]), (d[i], a[i], b[i])]$ , 对于  $0 \leq i < K$  的所有  $i$ 。

2.  $L$  个中间码元  $C[0], C[1], \dots, C[L-1]$  满足 B.5.2.3 中定义的预编码关系。

#### B.5.2.4.2 中间码元的计算

该小节描述了用于计算满足 B.5.2.4.1 中的约束的  $L$  个中间码元  $C[0], C[1], \dots, C[L-1]$  的可能的方法。

从  $K$  个输入码元生成  $N$  个输出码元的码生成器矩阵  $G$  是 GF(2) 上的  $N \times K$  矩阵, 其中每行对应输出码元中的一个而每列对应于输入码元中的一个, 且其中第  $i$  个输出码元等于第  $i$  行中其列包含非零项的那些输入码元的和。

然后,  $L$  个中间码元可计算如下:

令

**C** 标示  $L$  个中间码元的列向量  $C[0], C[1], \dots, C[L-1]$ 。

**D** 标示包含其后跟随  $K$  个源码元  $C[0], C[1], \dots, C[K-1]$  的  $S+H$  个零码元的列向量

然后, 以上约束定义 GF(2) 上的  $L \times L$  矩阵 **A**, 使得:

$$\mathbf{A} \cdot \mathbf{C} = \mathbf{D}$$

矩阵 **A** 可如下构造:

令:

**G<sub>LDPC</sub>** 为 LDPC 码元的  $S \times K$  生成器矩阵。所以,

$$\mathbf{G}_{\text{LDPC}} \cdot (C[0], \dots, C[K-1])^T = (C[K], \dots, C[K+S-1])^T$$

**G<sub>HDPC</sub>** 为半数码元的  $H \times (K+S)$  生成器矩阵, 所以,

$$\mathbf{G}_{\text{HDPC}} \otimes (C[0], \dots, C[S+K-1])^T = (C[K+S], \dots, C[K+S+H-1])^T$$

**I<sub>s</sub>** 为  $S \times S$  单位矩阵

**I<sub>H</sub>** 为  $H \times H$  单位矩阵

**0<sub>SxH</sub>** 为  $S \times H$  零矩阵

$\mathbf{G}_{\text{LT}}$  为 LT 编码器生成的编码码元的  $K \times L$  生成器矩阵。所以，

$$\mathbf{G}_{\text{LT}} \cdot (C[0], \dots, C[L-1])^T = (C[0], C[1], \dots, C[K-1])^T$$

即，在且仅在  $C[i]$  被包括在被异或以产生  $\text{LTEnc}[K, (C[0], \dots, C[L-1]), (d[i], a[i], b[i])]$  的码元中的情况下， $\mathbf{G}_{\text{LT}i,j}=1$ 。

则：

$\mathbf{A}$  的前  $S$  行等于  $\mathbf{G}_{\text{LDPC}} \mid \mathbf{I}_S \mid \mathbf{Z}_{S \times H}$ 。

$\mathbf{A}$  的接下来  $H$  行等于  $\mathbf{G}_{\text{HDPC}} \mid \mathbf{I}_H$ 。

$\mathbf{A}$  的剩余  $K$  行等于  $\mathbf{G}_{\text{LT}}$ 。

矩阵  $\mathbf{A}$  在图 20 中绘出。中间码元随后可被计算为“

$$\mathbf{C} = \mathbf{A}^{-1} \cdot \mathbf{D}$$

源三元组被生成使得对于任意  $K$ ，矩阵  $\mathbf{A}$  具有满秩并因此不可逆。该计算可通过向  $K$  个源码元  $C[0], C[1], \dots, C[K-1]$  应用 MSCR 解码过程来实现以产生  $L$  个中间码元  $C[0], C[1], \dots, C[L-1]$ 。

为了从源码元高效地生成中间码元，可使用诸如 B.6 节中所述的高效解码器实现。源码元三元组被设计成有助于使用该算法对源码元进行高效解码。

### B.5.3. 第二编码步骤：连锁反应编码

在第二编码步骤中，通过使用根据 B.3.2.2 和 B.4.2 节生成的三元组  $(d, a, b) = \text{Trip}[K, X]$  向  $L$  个中间码元  $C[0], C[1], \dots, C[L-1]$  应用 B.5.4 节中定义的生成器矩阵  $\text{LTEnc}[K, (C[0], C[1], \dots, C[L-1]), (d, a, b)]$  来生成具有 ESI  $X$  的修复码元。

### B.5.4. 生成器

#### B.5.4.1 随机生成器

随机数生成器  $\text{Rand}[X, i, m]$  被定义如下，其中  $X$  是非负整数， $i$  是非负整数，而  $m$  是正整数，且生成的值是 0 与  $m-1$  之间的整数。令  $V_0$  和  $V_1$  是 256 项的数组，其中每项是 4 字节无符号整数。合适的随机数数组在附录 B.1 和 B.2 中作为示例提供，这些示例不应被理解为限制本发明的范围。给定这些假设， $\text{Rand}[X, i, m] = (V_0[(X+i) \% 256] \wedge V_1[(\text{floor}(X/256)+i) \% 256]) \% m$ 。如本文所使用的，除非另有指示，否则“随机”应被假定为包括“伪随机”和“基本随机”。

#### B.5.4.2 幂度生成器

幂度生成器  $\text{Deg}[v]$  被定义如下，其中  $v$  是至少为 0 且小于  $2^{20}=1048576$  的整数。

在图 21 中，找到使得  $f[j-1] \leq v < f[j]$  的索引  $j$

$\text{Deg}[v]=d[j]$

#### B.5.4.3 连锁反应编码码元生成器

编码码元生成器  $\text{LTEnc}[K, (C[0], C[1], \dots, C[L-1]), (d, a, b)]$  取以下输入：

$K$  是源块（子块）的源码元（或子码元）的数目。令  $L$  如 B.5.2 节中所述地从  $K$  导出，且令  $L'$  为大于或等于  $L$  的最小质数整数。

$(C[0], C[1], \dots, C[L-1])$  是如 B.5.2 节中所述地生成的  $L$  个中间码元（子码元）的数组。

$(d, a, b)$  是使用 B.5.3.4 节中定义的三元组生成器确定的源三元组，由此  $d$  是标示编码码元幂度的整数， $a$  是 1 与  $L'-1$ （内包含）之间的整数，而  $b$  是 0 与  $L'-1$ （内包含）之间的整数。

编码码元生成器根据以下算法生成单个编码码元作为输出：

```

While ( $b \geq L$ ) do  $b = (b + a) \% L'$ 
     $\text{LTEnc}[K, (C[0], C[1], \dots, C[L-1]), (d, a, b)] = C[b]$ .
For  $j = 1, \dots, \min(d-1, L-1)$  do
     $b = (b + a) \% L'$ 
    While ( $b \geq L$ ) do  $b = (b + a) \% L'$ 
         $\text{LTEnc}[K, (C[0], C[1], \dots, C[L-1]), (d, a, b)] = \text{LTEnc}[K, (C[0], C[1], \dots, C[L-1]), (d, a, b)] \wedge C[b]$ 

```

#### B.5.4.4 三元组生成器

三元组生成器  $\text{Trip}[K, X]$  取以下输入：

$K$  源码元的数目

$X$  编码码元 ID

令

$L$  如 B.5.2 节中所述地从  $K$  确定

$L'$  为大于或等于  $L$  的最小质数

$Q=65521$ ，小于  $2^{16}$  的最大质数

$J(K)$  为与  $K$  相关联的系统性索引。系统性索引是被选成使得以下过程连同用于构造本文所述的矩阵 **A** 的剩余过程得到不可逆的矩阵 **B** 的数。合适的系统性索引在附录 A 中仅作为示例被提供，这些示例不应被理解为限制本发明的

范围。

三元组生成器的输出是如下确定的三元组( $d, a, b$ ):

1.  $A = (53591 + J(K) \cdot 997) \% Q$
2.  $B = 10267 \cdot (J(K)+1) \% Q$
3.  $Y = (B + X \cdot A) \% Q$
4.  $v = \text{Rand}[Y, 0, 2^{20}]$
5.  $d = \text{Deg}[v]$
6.  $a = 1 + \text{Rand}[Y, 1, L'-1]$
7.  $b = \text{Rand}[Y, 2, L']$

## B.6 FEC 解码器实现

### B.6.1 概要

该节描述了用于本说明书中所述的 MSCR 码的高效解码算法。注意，每个接收到的编码码元可被考虑为中间码元之间的方程的值。从这些联立方程以及中间码元之间已知的预编码关系，用于求解联立方程的任何算法都可成功地解码中间码元以及由此解码出源码元。然而，所选的算法对解码的计算效率有很大影响。

### B.6.2 解码源块

#### B.6.2.1 概要

假定解码器知道其要解码的源块的结构，包括码元大小  $T$ 、以及源块中码元的数目  $K$ 。

从 B.5 节中所述的算法，MSCR 解码器可计算预编码码元的总数  $L=K+S+H$  并确定它们是如何从要解码的源块生成的。在该描述中，假定要解码的源块的收到编码码元已被传递给解码器。此外，对于每个这种编码码元假定其异或等于编码码元的中间码元的数目和集合已被传递给解码器。在源码元的情形中，如 B.5.2.2 节中所述的源码元三元组指示共计给出每个源码元的中间码元的数目和集合。

令  $N \geq K$  为源块的收到编码码元的数目并令  $M=S+H+N$ 。以下  $M \times L$  矩阵 **A** 可从针对要解码的源块被传递给解码器的信息导出。令 **C** 为  $L$  个中间码元的列向量，并令 **D** 为具有接收机已知的值的  $M$  个码元的列向量，其中这  $M$  个码元当中的最后  $S+H$  个是对应于 LDPC 和 HDPC 码元的零值码元（这些是 LDPC

和 HDPC 码元的校验码元而不是 LDPC 和 HDPC 码元本身）， $M$  个码元当中剩余的  $N$  个是源块的接收到的编码码元。然后，**A** 是满足  $\mathbf{A} \cdot \mathbf{C} = \mathbf{D}$  的矩阵，其中这里标示  $\text{G}(256)$  上的矩阵乘法。矩阵 **A** 具有如图 23 所示的块结构。该块结构包括  $N$  行和  $L$  列的矩阵 **F**， $S$  行和  $L-S-H$  列的矩阵 **E**， $S \times S$  单位矩阵 **I**， $S$  行和  $H$  列全零的矩阵 **0**， $H$  行和  $L-H$  列的矩阵 **B**，以及  $H \times H$  的单位矩阵 **J**。子矩阵 **B** 具有在域  $\text{GF}(256)$  上定义的项，而矩阵 **E** 和 **F** 具有 0/1 项，即在域  $\text{GF}(2)$  中的项。矩阵 **F** 定义动态编码过程，矩阵 **E** 定义如上所述的 LDPC 编码过程，以及矩阵 **B** 定义 HDPC 编码过程。特别地，如果对应索引  $j$  的中间码元在编码中被异或成为对应索引  $i$  的编码码元，则  $\mathbf{F}[i,j] = 1$ 。对于所有的其它  $i$  和  $j$ ， $\mathbf{F}[i,j] = 0$ 。类似地，如果对应索引  $j$  的中间码元被异或成对应索引  $i$  的 LDPC 码元，则  $\mathbf{E}[i,j] = 1$ 。最后，如果对应索引  $j$  的中间码元的动作  $\beta$  的结果被异或成对应索引  $i$  的 HDPC 码元，则  $\mathbf{B}[i,j] = \beta$ 。

解码源块等价于从已知的 **A** 和 **D** 解码 **C**。显然在且仅在  $\text{GF}(256)$  上 **A** 的秩为  $L$  的情况下 **C** 是可解码的。一旦 **C** 已被解码，丢失的源码元可通过使用源码元三元组确定中间码元的数目和集合——其被异或以获得每个丢失的源码元——来获得。

解码 **C** 中的第一步骤是形成解码调度。在该步骤中 **A** 被使用高斯消元（使用行操作以及行与列的重排序）并在丢弃  $M-L$  行之后被转换成  $L \times L$  单位矩阵。解码调度包括高斯消元过程中行操作的序列以及行与列重排序，并且仅取决于 **A** 而不取决于 **D**。从 **D** 解码 **C** 可与解码调度的形成同时进行，或者解码过程可基于解码调度在之后进行。

**C** 的解码调度与解码之间的一一对应关系如下。首先令  $c[0] = 0, c[1] = 1, \dots, c[L-1] = L-1$  且  $d[0] = 0, d[1] = 1, \dots, d[M-1] = M-1$ 。

每次 **A** 的行  $i$  在解码调度中被异或成为行  $i'$ ，然后在解码过程中码元  $D[d[i]]$  被异或成为码元  $D[d[i']]$ 。我们将该操作称为  $\text{GF}(2)$ -行操作。

每次 **A** 的行  $i$  的  $\alpha$  倍 ( $\text{GF}(256)$  中的某一  $\alpha$ ) 在解码调度中被异或成  $i'$ ，然后在解码过程中码元  $\alpha * D[d[i]]$  被异或成码元  $D[d[i']]$ 。我们将该操作称为  $\text{GF}(256)$ -行操作。注意  $\text{GF}(2)$ -行操作是其中元素  $\alpha$  为 1 的  $\text{GF}(256)$ -行操作的特例。

每次行  $i$  在解码调度中与行  $i'$  交换，然后在解码过程中  $d[i]$  的值与  $d[i']$  的值交换。

每次列  $j$  在解码调度中与列  $j'$  交换，然后在解码过程中  $c[j]$  的值与  $c[j']$  的值交换。

从该对应关系可以看出，源块的解码中码元异或的总数与高斯消元中行操作（不是交换）的数目有关。由于  $\mathbf{A}$  在高斯消元及丢弃最后  $M-L$  行之后是  $L \times L$  单位矩阵，显然在成功解码结束时， $L$  个码元  $D[d[0]], D[d[1]], \dots, D[d[L-1]]$  是  $L$  个码元  $C[c[0]], C[c[1]], \dots, C[c[L-1]]$  的值。

执行高斯消元以形成解码调度的次序对该解码是否成功没有关系。然而，解码的速度很大地依赖于执行高斯消元的次序。（此外，维持  $\mathbf{A}$  的稀疏表示是至关重要的，尽管这里并未描述）。执行 GF(2)-行操作比执行 GF(256)-行操作更有效率也是显然的。因此，在执行高斯消元时，以矩阵  $\mathbf{A}$  当中具有取自域 GF(2) 的元素的行作为中心进行操作是较佳的。将矩阵当中对应于 HDPC 码元的行的消元留到高斯消元过程的末尾也是有利的。该节的剩余部分描述了可相对高效地执行高斯消元的次序。

### B.6.2.2 第一阶段

参照图 23，用  $\mathbf{X}$  表示包括如图 24a 所示的  $\mathbf{F}$ 、 $\mathbf{E}$ 、 $\mathbf{I}$  和  $\mathbf{0}$  的矩阵。

高斯消元的第一阶段，矩阵  $\mathbf{X}$  在概念上被分割成多个子矩阵。这些子矩阵大小通过被初始化为 0 的非负整数  $i$  和  $u$  来参数化。 $\mathbf{X}$  的子矩阵为：

(1) 由前  $i$  行与前  $i$  列的交集限定的子矩阵。这在该阶段中每一步骤的结束时是单位矩阵。

(2) 由前  $i$  行与除前  $i$  列和最后  $u$  列的所有列的交集限定的子矩阵。该子矩阵的所有项为零。

(3) 由前  $i$  列与除前  $i$  行之外的所有行的交集限定的子矩阵，该子矩阵的所有项为零。

(4) 由所有行与最后  $u$  列的交集限定的子矩阵  $\mathbf{U}$ 。

(5) 由除前  $i$  列和最后  $u$  列之外的所有列与除前  $i$  行之外的所有行的交集形成的子矩阵  $\mathbf{V}$ 。

图 22 示出了  $\mathbf{X}$  的子矩阵。在第一阶段的开始， $\mathbf{V}=\mathbf{X}$ 。在每一步骤，选择

$\mathbf{X}$  的一行。由  $\mathbf{V}$  的结构限定的以下图形被用来确定选择  $\mathbf{X}$  的哪一行。与  $\mathbf{V}$  相交的列是该图形中的节点，而在  $\mathbf{V}$  中正好具有两个 1 的行是该图形中连接这两个 1 的位置上的两列（节点）的边。该图形中的分量是使得在该图形中的每对节点/边之间存在路径的节点（列）和边（行）的最大集合。分量的大小是该分量中节点（列）的数目。该图形按  $\mathbf{Y}$  标示如下：

在第一阶段至多有  $L$  个步骤。当  $\mathbf{V}$  或者消失或者变为零矩阵时，该阶段结束。在每一步骤中，如下选择  $\mathbf{X}$  的行：

如果  $\mathbf{V}$  的所有项为零，则选择该行并且第一阶段结束。

否则，令  $r$  为使得  $\mathbf{X}$  中的至少一行在  $\mathbf{V}$  中正好有  $r$  个 1 的最小正整数。

如果  $r=1$ ，则选择在  $\mathbf{V}$  中正好具有一个 1 的行。

如果  $r=2$ ，则选择作为由  $\mathbf{Y}$  定义的图形中最大大小分量的一部分、在  $\mathbf{V}$  中正好具有两个 1 的任意行。

如果  $r>2$ ，则选择在  $\mathbf{V}$  中正好具有  $r$  个 1 且在这些行中具有最小的原始权重的行。

在该步骤中选择出该行之后， $\mathbf{X}$  当中与  $\mathbf{V}$  相交的第一行与该被选行相交换以使得所选行是与  $\mathbf{V}$  相交的第一行。 $\mathbf{X}$  当中与  $\mathbf{V}$  相交的那些列被重排序以使得所选行中  $r$  个 1 当中的一个出现在  $\mathbf{V}$  的第一列并且使得剩余的  $r-1$  个 1 出现在  $\mathbf{V}$  的最后列中。然后，所选行被异或到  $\mathbf{X}$  当中该所选行以下在  $\mathbf{V}$  的第一列具有 1 的所有其它行中。换言之，我们在该步骤中执行了 GF(2)-行操作。最后， $i$  被递增 1 且  $u$  被增加  $r-1$ ，结束步骤。

令  $v$  标示该阶段结束时矩阵  $\mathbf{V}$  的列数。在对矩阵  $\mathbf{B}$  的各列转置以使得  $\mathbf{V}$  的各列对应于  $\mathbf{X}$  的最后  $v$  列之后，矩阵  $\mathbf{X}$  将具有图 24b 中给定的形式。

### B.6.2.3 第二阶段

我们修改矩阵  $\mathbf{U}$  以使其另外包括矩阵  $\mathbf{X}$  的最后  $v$  行，并且相应地用  $u+v$  代替  $u$ 。子矩阵  $\mathbf{U}$  被进一步分割成前  $i$  行  $\mathbf{U}_上$  以及剩余的  $N+S-i$  行  $\mathbf{U}_下$ ，如图 25 所示。在第二阶段对  $\mathbf{U}_上$  执行高斯消元。在该步骤之后，矩阵  $\mathbf{U}_下$  将具有图 26 中给定的形式，即在行和列的转置之后，前  $s$  行与前  $s$  列的交集是单位矩阵——被称为  $\mathbf{I}$ ，最后  $m$  行是零，而前  $s$  行与最后  $u-s$  列的交集形成矩阵  $\mathbf{W}$ 。注意， $s+m$  等于矩阵  $\mathbf{U}_下$  的行数  $N+S-i$ 。如果  $s$  的值为  $u$ ，则可跳过下一阶段。如果  $m$

的值大于  $H-v$ , 则返回解码差错, 因为在此情形中矩阵  $\mathbf{A}$  的秩小于  $L$ 。矩阵  $\mathbf{X}$  的最后  $m$  行被丢弃, 以使得在该阶段之后  $\mathbf{A}$  具有图 27 中给出的形式。在该图中,  $\mathbf{B}_1, \dots, \mathbf{B}_3$  是各自具有  $H$  行和 GF(256) 中的项的矩阵。接着, 对矩阵  $\mathbf{B}_1$  和  $\mathbf{B}_2$  执行 G(256)-行操作以将它们消零。这可以两种方式之一来实现。第一种方法中,  $\mathbf{A}$  的前  $i$  行被用来借助 GF(256)-行操作对矩阵  $\mathbf{B}_1$  消零。然后  $\mathbf{A}$  的下  $s$  行被用来对矩阵  $\mathbf{B}_2$  消零。在第二种方法中, 行  $i$  到  $i+s-1$  (内包含) 被用来借助 GF(2)-行操作对  $\mathbf{U}_+$  上的前  $s$  列消零并随后  $\mathbf{X}$  的前  $i+s$  行被用来借助 GF(256)-行操作对  $\mathbf{B}_1$  和  $\mathbf{B}_2$  消零。如对本领域的普通技术人员显而易见的, 上述用于构造 HDPC 码元的方法算法导致用于矩阵  $\mathbf{B}_1$  的消零 (在第一种方法中) 或  $\mathbf{B}_1$  和  $\mathbf{B}_2$  两者的消零 (在第二种方法中) 的类似算法。该算法对于 GF(256) 元素对码元的动作的计算的要求仅为每矩阵列一次加  $\mathbf{H}$  的每行一次。因此, 上述第二种方法在对矩阵  $\mathbf{B}_1$  和  $\mathbf{B}_2$  消零方面导致总体较少的操作。

在该步骤之后, 矩阵  $\mathbf{A}$  具有图 28 中所给的形式。矩阵  $\mathbf{T}$  具有  $H$  行和  $u-s$  列。对矩阵  $\mathbf{T}$  执行高斯消元以将其变为其后跟有  $H-u+s$  行的单位矩阵。如果这不可能, 即如果  $\mathbf{T}$  的秩小于  $u-s$ , 则标志解码差错。在该级结束时, 在丢弃最后  $H-u+s$  行后矩阵  $\mathbf{A}$  具有图 29 中给出的形式。在该图中,  $\mathbf{I}$  标示  $s \times s$  单位矩阵, 而  $\mathbf{J}$  标示  $u-s \times u-s$  单位矩阵。

#### B.6.2.4 第三阶段

在第二阶段之后,  $\mathbf{A}$  当中需要被消零以完成将  $\mathbf{A}$  转换成  $L \times L$  单位矩阵的部分在跟随有将  $\mathbf{B}_1$  和  $\mathbf{B}_2$  消零的第一方法的情形下是  $\mathbf{W}$  和  $\mathbf{U}_+$  上的所有  $u$  列, 或者在跟随有将  $\mathbf{B}_1$  和  $\mathbf{B}_2$  消零的第二方法的情形下是  $\mathbf{W}$  和  $\mathbf{U}_+$  上的最后  $u-s$  列。在前一情形中, 由于矩阵  $\mathbf{W}$  通常较小, 所以可使用元素 GF(2)-行操作来消零。在该步骤之后, 矩阵  $\mathbf{A}$  具有图 30 中给出的形式。在这两种情形下, 矩阵当中需要消零的剩余部分现在是矩形。在前一情形下, 其大小为  $i$  行和  $u$  列, 在后一情形中, 其大小为  $i+s$  行和  $u-s$  列。在下面, 我们将  $i'$  用于该矩阵中的行数以及将  $u'$  用于列数并用  $\hat{\mathbf{U}}$  标示该矩阵。

剩余子矩阵  $\hat{\mathbf{U}}$  的行数  $i'$  一般远大于列数  $u'$ 。有多种方法可用来将  $\hat{\mathbf{U}}$  高效地消零。在一种方法中, 以下预算矩阵  $\mathbf{U}'$  被基于  $\mathbf{A}$  的最后  $u$  行和列 (我们将其标示为  $\mathbf{I}_u$ ) 计算出, 并且随后  $\mathbf{U}'$  被用于将  $\hat{\mathbf{U}}$  消零。 $\mathbf{I}_u$  的  $u$  行被分割成各自为  $\mathbf{z}$

行的  $\text{ceil}(u/z)$  组（针对每一整数  $z$ ）。然后，对于每一组的  $z$  行，计算出  $z$  行的所有非零组合，得到  $2^z - 1$  行（这可以用每组的各行的  $2^z - z - 1$  次异或来实现，因为  $\mathbf{I}_u$  中出现的 Hamming 权重一的组合无需重新计算）。因此，得到的预算矩阵  $\mathbf{U}'$  具有  $\text{ceil}(u/z) \cdot 2^z - 1$  行和  $u$  列。注意  $\mathbf{U}'$  不是矩阵  $\mathbf{A}$  的正式的一部分，但随后将被用来将  $\mathbf{U}$  上消零。在一优选实施例中， $z=8$ 。

对于  $\hat{\mathbf{U}}$  的  $i'$  行中的每一行，对于该行的  $\hat{\mathbf{U}}$  子矩阵的每一组的  $z$  列，如果  $\hat{\mathbf{U}}$  中的  $z$  列项的集合不是全零，则该预算矩阵  $\mathbf{U}'$  当中匹配这  $z$  列中的模式的那一行被异或到该行，由此以将  $\mathbf{U}'$  的 1 行异或到该行为代价将该行中的这  $z$  列消零。

在该阶段后， $\mathbf{A}$  是  $L \times L$  单位矩阵且已经成功形成完整的解码调度。然后，可执行包括将已知编码码元异或的相应解码以基于该解码调度来恢复中间码元。

与所有源码元相关联的三元组是根据 B.5.2.2 计算出的。接收到的源码元的三元组被用于解码中。丢失的源码元的三元组被用来确定哪些中间码元需要被异或以恢复丢失的源码元。

#### 多域、单级连锁反应编码器/解码器

多域单级（MFSS）码具有本文所公开或暗示的有用属性。MFSS 码、编码器以及解码器的新颖设计在本文进行描述。在一个实施例中，数据被编码以从源被传输到目的地，其中每个输出码元是作为输入码元中的一个或多个用取自有限域的系数的线性组合生成的，对于每个输出码元：

- 根据随机过程选择大于 0 的整数  $d$ ——被称为输出码元的幂度，
- 根据随机过程选择大小为  $d$  的输入码元的集合，该输入码元集合被称为输出码元的近邻集合，
- 选择有限域集合以使得对于至少一个输出码元该集合包含至少两个有限域，
- 针对输出码元的近邻集合中的每个输入码元，从所选择的可能有限域集合中选择有限域，
- 针对输出码元的近邻集合中的每个输入码元根据随机过程从以上所选的有限域中选择非零元素。

用于选择输出码元的幂度的随机过程可以是 Luby I 和 Luby II 中所述的过

程，其中幂度是根据幂度分配来选择的。用于选择与每个输出码元相关联的输入码元的随机过程可以是 Luby I 和 Luby II 中所述的过程，其中输入码元是随机且唯一地选择的。如本文所使用的，“随机”可包括“伪随机”、“有偏随机”等。

该可能有限域的集合可以是集合 $\{GF(2), GF(256)\}$ 。

用于选择有限域的过程可以基于参数  $d_l$ ，以使得对于幂度小于  $d_l$  的输出码元，为该输出码元的近邻集合中的所有输入码元选择域  $GF(2)$ ，而对于幂度为  $d_l$  或更大的输出码元，为该输出码元的近邻集合中的至少一个、一些或所有成员选择域  $GF(256)$  并根据需要为该近邻集合中的剩余元素选择域  $GF(2)$ 。

用于从所选域中选择有限域元素的过程可以是简单随机过程，其中元素是从该域的非零元素当中随机唯一地选出的。

接收由如上所述的 MFSS 编码器编码的数据的解码器可通过以下操作解码输出码元以再生输入码元：根据上述方法形成表示该码的矩阵，该矩阵不包括静态行而包括对应该码的每个输出码元的一个动态行，并随后应用高斯消元以找到该矩阵的逆，以确保在高斯消元过程的每一级最小幂度的中枢行被选择。

如本领域的普通技术人员所清楚的，Luby I 和 Luby II 中所述的码的许多公知属性可同等应用于上述码并且特别是对合适的幂度分配的选择可确保高斯消元过程能够以较高概率标识剩余幂度一的行并且由此该解码过程如 Luby I 和 Luby II 中所述的连锁反应过程那样操作。

该 MFSS 码相对于本领域已知的码具有许多其它优点。首先，包括来自  $GF(256)$  的元素显著降低了任意接收到的输出码元相对于先前接收到的输出码元不是加性信息的概率。因此，该码的解码差错概率比先前的码低得多。例如，在一些情形下，Luby I 和 Luby II 中所述的码的失败概率得以改进。

该码相对于基于较大域的其它码的优点是较低幂度的输出码元一般首先将由高斯消元过程进行处理，因此包括的来自  $GF(256)$  的元素可以无需考虑推迟到之后解码过程。由于  $GF(256)$  上的操作相比于  $GF(2)$  上的操作是相对昂贵的，这与其中许多或所有元素都是使用来自  $GF(256)$  或其它较大有限域的元素构造的码相比极大地降低了计算复杂度。

相对于基于较大域的其它码的另一优点是对于使用较大域生成的输出码元，近邻集合中仅一个元素具有取自较大域的系数，由此对于每个这样的输出码元仅需要码元与有限域元素之间的一个操作。这导致总体较低的计算复杂度。

已知使用内码和外码来使用两个（或多个）编码过程编码输入码元导致提供常在更为复杂的码中找到的益处的简单码方案。通过内码和外码的使用，源码元首先被使用其中一种码进行编码，并且第一编码器的输出被提供给根据另一种码进行编码的编码器，而该结果被输出作为输出码元。使用 MFSS 当然与内/外码的使用不同。举例来说，输出码元是从输入码的近邻集合导出的。在本文所述的许多实施例中，每个输出码元都是输入码元的线性组合。在多级码情形下，每个输出码元可以是输入码元和/或冗余码元和/或中间码元的线性组合。

#### 密集多域码和用于这种码的编码器/解码器

在上述讲授的变形中，码的矩阵表示是密集矩阵。如所公知的，纠错码可以从有限域上的密集随机矩阵构造出。例如，可构造出广义矩阵，其中没有静态行且每个动态行包括来自  $GF(2^q)$  的元素，其中每个元素是随机选取的。然后可以构造出固定率的码，其中每个输出码元对应动态行之一并且是作为对于其在矩阵的该行的相应列中具有非零元素的那些输入码元的线性组合——使用这些元素作为线性组合过程中的系数——生成的。

本领域的技术人员公知的是随机选取的具有  $K$  行和  $K+A$  列并具有从  $GF(2^q)$  独立且随机选取的系数的矩阵具有小于  $K$  的秩的概率至多为  $2^{-qA}$ 。因此，具有  $K$  个输入和  $K/R$  个输出码元的码的解码差错概率——其中输出码元是使用从  $GF(2^q)$  随机选择的系数从输入码元独立且随机地生成的——至多为  $2^{-qA}$ ，如果接收到的编码码元的数目为  $K+A$ 。

在  $q=1$  的情形下，上述码具有合理的计算复杂度的优点，因为所有操作都在域  $GF(2)$  内，由此对应常规 XOR 操作。然而，在该情形下，一旦接收到  $A$  个额外码元那么失败概率的下限  $2^{-A}$  比所需的高得多。

在  $q=8$  的情形下，上述码具有较低失败概率的优点（对于接收到  $A$  个额外码元则以  $2^{-8A}$  为限度）。然而，在该情形下，所有操作都在域  $GF(256)$  内，由此在计算上相对较昂贵。

另一实施例允许用接近于以较小的  $q$  值可实现的计算复杂度来实现接近于使用较大  $q$  值实现的解码差错概率。在该实施例中，输出码元是作为具有取自或者  $GF(2^p)$  或者  $GF(2^q)$  的系数的输入码元的线性组合生成的，其中  $p < q$ 。在一个特定实施例中，正好  $(K-2p/q)/R$  个输出码元是使用来自  $GF(2^p)$  的系数生成的，剩余的  $2p/(qR)$  个输出码元是使用来自  $GF(2^q)$  的系数生成的。

在目的地接收到的数据可通过确定该码的收到输出码元与输入码元之间的关系并求解该线性关系集以确定输入码元来解码。

该码的解码差错概率至多为其中所有系数是选自域  $GF(2^p)$  的码的解码差错概率，并且取决于使用来自较大的域  $GF(2^q)$  的系数生成的码元的数目可显著地更低。然而，由于大多数输出码元是使用来自  $GF(2^p)$  的系数生成的，编码的计算复杂度仅比其中所有码元都是使用来自  $GF(2^p)$  的系数生成的码的计算复杂度稍大。此外，解码方法可以如此设计以使得用来自  $GF(2^p)$  的系数生成的码元被首先处理，由此大多数解码操作是完全以  $GF(2^p)$  中的操作来执行的。因此，解码方法的计算复杂度类似接近于仅使用  $GF(2^p)$  构造的码的计算复杂度。在一定优选实施例中， $p=1$  且  $q=8$ 。

#### 某些多域码的一些属性

在上述大多数示例中，输入和输出码元针对相同数目的比特进行编码并且每个输出码元被置于一个分组中（作为或者全部接收或者全部丢失的传输单元的分组）。在一些实施例中，通信系统被修改以使得每个分组包含若干输出码元。输出码元值的大小被设成基于多个因素由在文件或流送块到输入码元的原始拆分中的输入码元值大小确定的大小。解码过程基本保持不变，除非在接收到每个分组时输出码元大量到达。

输入码元和输出码元大小的设置通常由文件或流送块的大小以及将在其上传送输出码元的通信系统指定。例如，如果通信系统将数据比特编组在具有定义大小的分组中或者以其它方式编组，则码元大小的设计以该分组或编组大小开始。从此，设计者将确定在一个分组或组中将携带多少输出码元并确定该输出码元大小。为了简单起见，设计者可能将输入码元大小设置成等于输出码元大小，但如果输入数据使得不同的输入码元大小更为方便，则可使用不同的输入码元大小。

上述编码过程基于原始文件或流送的块来产生包含输出码元的分组流送。该流送中的每个输出码元是独立于所有其它输入码元生成的，因此对于可创建的输出码元的数目没有上下限。键与每个输出码元相关联。该键、输入文件或流送块的一些内容决定输入码元的值。连续生成的输出码元无需具有连续的键，并且在一些应用中，随机生成键序列或伪随机地生成该序列将是较优的。

多级解码具有这样的属性，即  $K$  个等大小的输入码元的块平均可以非常高的概率从  $K+A$  个输出码元恢复，其中  $A$  相对于  $K$  较小。例如，在以上首先描述的优选实施例中，当  $K=100$  时，图 31 示出了根据从生成的前 120 个输出码元当中随机选出的  $K+A$  个输出码元解码的失效的概率，而图 32 的表示出了根据从生成的前 110 个输出码元当中随机选出的  $K+A$  个输出码元解码的失效的概率。

由于特定输出码元是以随机或伪随机次序生成的，并且传输中特定输出码元的丢失通常与码元的值无关，所以需要接收以恢复输入文件或块的输出码元的实际数目的方差较小。在  $K+A$  个输出码元的特定集合不足以将块解码的许多情形中，如果接收机能够从一个或多个源接收更多输出码元则该块仍是可恢复的。

由于输出码元的数目仅受  $I$  的分辨率限制，将可生成大于  $K+A$  个输出码元。例如，如果  $I$  是 32 比特数，则可生成四十亿个不同输出码元，而文件或流送块可包括  $K=50,000$  个输入码元。在一些应用中，这四十亿个输出码元中仅一些可被生成并传送，且输入文件或流送块可使用可能输出码元中的非常小的少部分来恢复，并且输入文件或块可用稍大于  $K$  个输出码元的码元来恢复的概率非常优良（假定输入码元大小与输出码元大小相同）。

在一些应用中，没有能将所有输入码元解码或者以相对较低的概率将所有输入码元解码是可接受的。在这种应用中，接收机可在接收到  $K+A$  个输出码元之后停止尝试将所有输入码元解码。或者接收机可在接收到少于  $K+A$  个输出码元之后停止接收输出码元。在一些应用中，接收机甚至可仅接收  $K$  个或更少的输出码元。因此，应该理解在本发明的一些实施例中，所需的精确度可以不是所有输入码元的完全恢复。

此外，在一些可接受不完整恢复的情形中，数据可被编码成使得不是所有

---

输入码元都可被恢复或者使得输入码元的完全恢复将要求接收比输入码元多得多的输出码元。这种编码一般将要求较少的计算开销，由此是降低编码计算开销的可接受方式。

应该理解上述附图中各种功能块可以用硬件和/或软件的组合来实现，且在特定实现中一些块的部分或所有功能可以被组合。类似地，应该理解本文所述的各种方法可通过硬件和/或软件的组合来实现。

以上描述是说明性而非限制性的。本发明的许多变形对于本领域的技术人员而言在阅读本公开的基础上是显而易见的。因此，本发明的范围不应参照以上描述确定，而应参照所附权利要求以及其等效方案的全部范围来确定。

### 附录 A. 用于系统性索引 $J(K)$ 的值

对于  $K$  的每个值，系统性索引  $J(K)$  被设计成具有这样的属性，即源码元三元组  $(d[0], a[0], b[0]), \dots, (d[L-1], a[L-1], b[L-1])$  的集合使得  $L$  个中间码元被唯一地定义，即 B.5.2.4.2 中的矩阵  $\mathbf{A}$  具有满秩并因此不可逆。以下是 4 与 8192 之间（内包含）的  $K$  值的合适系统性索引列表且是作为示例给出的。这些值的次序是按照阅读的次序，即从第一行的第一个数到第一行的最后一个数，其后是第二行上的第一个数，以此类推。

22,	1486,	2151,	2239,	4286,	3887,	2016,	3247,	4531,	490,	2223,
705,	1243,	2241,	3457,	3030,	1918,	2349,	455,	1745,	182,	
1130,	29,	381,	1088,	23,	1147,	353,	951,	161,	56,	
421,	401,	1141,	557,	75,	611,	409,	828,	1514,	2722,	
1044,	718,	768,	1696,	2145,	2515,	57,	43,	136,	438,	
815,	540,	88,	270,	364,	236,	158,	14,	49,	160,	
203,	49,	93,	104,	83,	567,	322,	143,	77,	29,	
70,	37,	333,	120,	5,	5,	5,	408,	183,	356,	
110,	233,	4,	251,	90,	1170,	322,	245,	664,	1057,	
95,	512,	184,	1551,	1477,	778,	1038,	809,	719,	1025,	
768,	1127,	689,	76,	1367,	172,	1313,	1557,	311,	311,	
559,	1145,	624,	234,	172,	913,	1555,	1555,	1134,	692,	
687,	692,	1501,	692,	1106,	1040,	195,	25,	1127,	1234,	
1404,	1125,	798,	1511,	75,	947,	671,	1049,	385,	1273,	
834,	1116,	1386,	1386,	399,	84,	1563,	365,	1563,	611,	
611,	611,	1020,	819,	1688,	1688,	1550,	19,	1600,	1600,	
796,	19,	1469,	1571,	1571,	1117,	1059,	476,	789,	1117,	
1207,	1117,	1163,	1163,	512,	1719,	1719,	1719,	1719,	100,	
623,	623,	623,	1567,	1567,	623,	876,	876,	876,	876,	
816,	816,	814,	814,	431,	323,	1,	220,	1286,	1286,	
623,	1286,	1286,	1286,	228,	1642,	1642,	1642,	259,	1410,	
259,	259,	353,	122,	826,	522,	826,	485,	1260,		
1061,	1061,	1260,	1260,	1260,	1678,	353,	870,	870,	870,	
714,	145,	3,	991,	211,	1644,	1644,	288,	288,	857,	
1413,	1413,	1677,	857,	242,	242,	1565,	1565,	1677,	1810,	
714,	714,	714,	714,	714,	1757,	714,	714,	964,	666,	
964,	964,	1432,	219,	219,	908,	365,	679,	80,	1286,	
1432,	1432,	579,	966,	62,	62,	62,	75,	62,	62,	
62,	1261,	62,	62,	62,	62,	894,	690,	690,		
456,	538,	538,	1115,	1999,	1999,	102,	102,	1137,	1115,	
102,	579,	579,	783,	783,	96,	1535,	1073,	1073,	612,	
612,	1386,	1197,	690,	690,	690,	690,	690,	690,	690,	
690,	968,	244,	244,	1812,	244,	968,	160,	656,	160,	
160,	160,	656,	263,	263,	815,	146,	1099,	1074,	1099,	
1099,	1099,	972,	1876,	222,	1994,	222,	327,	327,	1400,	
1441,	1898,	1292,	1400,	1066,	1066,	1755,	54,	54,	54,	
54,	54,	54,	54,	1791,	1791,	1781,	554,	1047,		
1047,	1047,	1047,	1047,	1047,	631,	1219,	289,	289,	1321,	
1191,	6,	1781,	184,	184,	6,	2030,	1704,	1704,	1704,	
1704,	1088,	1088,	1088,	1088,	1223,	1223,	2053,	2053,	8,	
1704,	2053,	2053,	8,	1704,	1969,	770,	1969,	1969,	1371,	
1371,	1353,	1810,	1810,	1810,	652,	652,	652,	652,	652,	
652,	652,	652,	423,	652,	1566,	1566,	1566,	1566,	1566,	
1566,	1566,	1566,	1566,	1617,	965,	965,	1267,	1267,		
1267,	1267,	1892,	1699,	1699,	1267,	981,	1091,	981,	981,	
2053,	2053,	510,	510,	510,	510,	927,	1843,	1843,	2203,	

234,	234,	865,	1800,	865,	944,	927,	927,	1693,	1967,
1379,	1379,	1379,	1379,	1379,	843,	1441,	499,	1441,	160,
601,	1828,	675,	675,	601,	675,	675,	160,	2338,	203,
1138,	1138,	1244,	1244,	1244,	1244,	346,	323,	850,	850,
870,	870,	927,	927,	927,	408,	677,	677,	677,	677,
1166,	762,	677,	131,	965,	965,	965,	965,	233,	965,
540,	540,	540,	540,	540,	540,	540,	540,	540,	540,
411,	411,	81,	411,	411,	411,	943,	943,	91,	1067,
943,	1067,	304,	304,	949,	1046,	916,	751,	751,	1046,
576,	1193,	576,	576,	1193,	1208,	375,	1089,	1089,	375,
945,	945,	540,	540,	540,	540,	540,	540,	540,	540,
540,	540,	81,	41,	571,	571,	571,	723,	723,	723,
723,	723,	723,	723,	589,	589,	403,	403,	403,	1044,
980,	488,	698,	132,	1187,	1219,	1219,	1219,	1219,	1219,
851,	231,	929,	929,	929,	929,	929,	929,	929,	929,
929,	929,	929,	929,	8,	8,	8,	8,	540,	973,
973,	973,	973,	1,	983,	1144,	1071,	1071,	37,	37,
37,	173,	173,	173,	173,	665,	173,	173,	918,	918,
1310,	1297,	1297,	1297,	699,	662,	547,	882,	1234,	1234,
29,	29,	1228,	459,	547,	365,	34,	34,	34,	34,
34,	34,	34,	34,	317,	317,	317,	317,	317,	15,
242,	242,	242,	242,	437,	242,	292,	292,	292,	385,
753,	1061,	1061,	1061,	1061,	37,	906,	37,	1191,	1191,
1012,	1327,	869,	869,	266,	769,	41,	266,	120,	506,
506,	506,	943,	506,	506,	715,	1155,	142,	142,	142,
375,	375,	375,	375,	375,	375,	375,	375,	375,	375,
690,	375,	375,	375,	548,	939,	94,	94,	94,	939,
939,	939,	939,	94,	1219,	1219,	1219,	1219,	1219,	1219,
1253,	359,	666,	666,	94,	666,	666,	666,	94,	666,
666,	666,	633,	477,	968,	498,	968,	968,	865,	8,
165,	1219,	1219,	165,	165,	165,	842,	842,	1219,	1219,
343,	343,	987,	920,	987,	987,	987,	987,	139,	522,
522,	522,	522,	522,	640,	640,	640,	640,	173,	173,
365,	487,	487,	487,	965,	120,	120,	965,	120,	120,
120,	120,	965,	120,	1269,	1269,	965,	965,	1187,	1187,
1187,	1187,	481,	481,	1209,	663,	823,	823,	1143,	46,
46,	46,	46,	46,	46,	46,	46,	46,	46,	46,
46,	46,	46,	46,	46,	46,	939,	757,	757,	757,
757,	757,	902,	902,	1446,	902,	351,	1446,	351,	1446,
1222,	1222,	716,	1222,	1222,	1222,	1222,	833,	833,	833,
912,	912,	912,	912,	94,	912,	915,	808,	1163,	808,
1421,	443,	443,	443,	443,	443,	443,	432,	905,	366,
366,	366,	366,	408,	408,	408,	408,	408,	408,	1156,
1156,	145,	1156,	973,	973,	476,	476,	867,	634,	1077,
1077,	375,	375,	375,	375,	375,	375,	1371,	1371,	1371,
1371,	1371,	1371,	1371,	1371,	522,	383,	148,	1267,	522,
522,	1298,	674,	1298,	939,	674,	674,	674,	674,	1298,
1298,	1298,	197,	197,	197,	197,	197,	120,	918,	918,
918,	918,	918,	47,	47,	1065,	1308,	1065,	1065,	1308,
1308,	1308,	1308,	1308,	1308,	1415,	1415,	417,	417,	417,
417,	417,	417,	1206,	613,	537,	1437,	1437,	476,	476,
476,	476,	1437,	669,	669,	558,	303,	303,	902,	902,
902,	902,	125,	902,	10,	1501,	1472,	535,	535,	535,
535,	535,	535,	977,	977,	977,	977,	409,	1309,	1309,
1309,	1309,	1309,	1309,	1309,	1309,	1309,	1309,	1309,	1309,
1309,	796,	796,	796,	716,	617,	694,	973,	973,	973,
973,	1054,	1054,	1054,	1054,	662,	662,	447,	447,	447,
447,	447,	447,	634,	634,	634,	634,	634,	634,	634,
37,	1459,	949,	1459,	1459,	1459,	1459,	634,	565,	565,
777,	634,	777,	977,	977,	1057,	1057,	657,	657,	657,
657,	1057,	1057,	1057,	657,	657,	657,	657,	657,	657,
657,	657,	657,	657,	657,	1437,	1500,	263,	263,	263,

70,	570,	570,	70,	70,	70,	70,	1497,	1497,	1497
1497,	883,	364,	883,	883,	1128,	1390,	1390,	1390,	1390
1032,	1032,	1032,	1338,	1338,	1338,	1338,	1338,	1338,	1166
1166,	1166,	1166,	1166,	1166,	983,	983,	983,	983,	71
983,	983,	983,	1137,	1137,	1137,	1137,	1137,	1137,	1137
1137,	1137,	1137,	1137,	871,	571,	571,	571,	571,	918
571,	918,	571,	571,	571,	118,	118,	118,	118,	118
118,	809,	809,	1437,	1437,	1437,	1437,	1437,	1437,	537
206,	206,	206,	206,	537,	537,	537,	537,	206,	537
537,	498,	120,	498,	120,	566,	566,	1219,	566,	1219
566,	566,	1219,	1219,	1219,	1219,	1219,	1219,	1219,	1219
1219,	1219,	1219,	95,	550,	1367,	1367,	1367,	1367,	1705
1230,	1230,	1407,	1230,	1230,	363,	363,	1539,	1539,	1539
1539,	1539,	1539,	1380,	1380,	1380,	1380,	1152,	1455,	1345
1345,	366,	366,	1308,	1308,	1308,	1308,	1308,	1308,	1308
1308,	1219,	1219,	1219,	871,	871,	581,	581,	581,	581
581,	581,	581,	1,	1,	121,	1,	1,	1,	121
1249,	1,	1249,	1249,	441,	1249,	1249,	1249,	1249,	1249
1495,	1653,	1249,	1249,	1249,	615,	1064,	1249,	615,	615
1249,	615,	221,	615,	615,	615,	98,	723,	723,	723
723,	723,	723,	1533,	1533,	106,	106,	106,	106,	1232
1232,	1864,	1232,	1232,	1232,	1864,	912,	918,	273,	273
273,	273,	273,	918,	273,	273,	273,	273,	273,	273
273,	273,	273,	273,	273,	1348,	1348,	1348,	1348,	1348
1466,	1348,	1466,	1466,	160,	1460,	1460,	1460,	1460,	1460
1792,	1792,	473,	473,	473,	1460,	473,	1792,	1792,	1009
1009,	1009,	1009,	106,	106,	1434,	902,	1434,	1434,	1387
1387,	1387,	1387,	1387,	1387,	1042,	1512,	1512,	1512,	1512
1047,	1512,	1512,	1588,	1588,	1550,	662,	927,	628,	103
222,	103,	103,	1313,	980,	980,	662,	980,	980,	980
980,	980,	980,	632,	1107,	1048,	632,	1107,	1107,	1620
632,	1067,	1620,	963,	963,	94,	94,	94,	514,	1725
1257,	1676,	1676,	1676,	1676,	1437,	1437,	1417,	1417,	681
1437,	1064,	1064,	1117,	1117,	1064,	1064,	1117,	1117,	1117
1064,	1117,	1117,	453,	677,	871,	871,	871,	871,	871
871,	871,	313,	313,	677,	1259,	1259,	118,	1259,	1259
1707,	441,	441,	1766,	1766,	1766,	1766,	1766,	1766,	1766
1766,	1766,	1766,	1766,	1766,	1767,	1767,	1767,	935,	1767
935,	549,	549,	549,	159,	873,	1868,	1868,	1868,	1598
1868,	1140,	1587,	1587,	1587,	1587,	1587,	1140,	1140,	1457
1457,	1457,	1457,	1937,	1253,	1937,	1937,	1253,	1937,	1195
1937,	986,	986,	1821,	1821,	1797,	41,	1439,	1011,	1011
1011,	1011,	1011,	1011,	1011,	1439,	1439,	1381,	1381,	1381
1381,	1200,	1200,	1200,	1200,	1200,	1200,	858,	858,	1077
1669,	1669,	1669,	108,	108,	968,	690,	108,	968,	973
343,	1313,	1313,	1313,	1313,	1313,	1313,	949,	949,	949
949,	949,	949,	716,	949,	716,	716,	1581,	1431,	1431
1431,	1431,	1431,	1431,	1431,	1581,	1581,	1431,	1431,	1581
1581,	664,	1581,	845,	396,	664,	664,	650,	650,	650
814,	814,	1973,	461,	461,	461,	492,	1997,	306,	1114
865,	865,	1820,	865,	865,	865,	1104,	952,	952,	1104
1104,	1257,	1306,	1306,	968,	952,	952,	175,	175,	175
175,	1066,	1632,	452,	902,	902,	902,	902,	1067,	243
243,	1067,	243,	1360,	408,	408,	408,	498,	498,	4
498,	498,	498,	498,	498,	1218,	1218,	1165,	6,	6
1165,	6,	1381,	1165,	1165,	1165,	1725,	1725,	164,	1725
1725,	1725,	54,	1725,	716,	716,	1114,	1114,	1114,	1114
1067,	1426,	1426,	1426,	1426,	1426,	2162,	2162,	2162,	2162
2162,	2162,	243,	243,	243,	243,	1095,	243,	1095,	243
243,	243,	243,	243,	243,	1982,	243,	243,	243,	243
347,	347,	347,	347,	347,	347,	1823,	1069,	1823,	1069
1671,	1745,	2179,	376,	623,	2179,	376,	376,	2179,	376,

2179,	2179,	2179,	376,	756,	571,	2159,	2159,	1125,	2159,
1125,	1125,	2159,	2159,	1322,	180,	977,	180,	180,	977,
180,	977,	180,	565,	180,	977,	2124,	857,	2124,	857,
2124,	2124,	857,	2124,	842,	842,	842,	842,	842,	842,
62,	62,	1390,	62,	1390,	1420,	62,	62,	62,	62,
1055,	1055,	1055,	1055,	1055,	1055,	175,	1055,	1055,	1055,
1055,	1055,	175,	1055,	918,	918,	718,	233,	233,	718,
1263,	1263,	1263,	1263,	757,	2048,	70,	1723,	1723,	1723,
2154,	2154,	1532,	233,	1532,	1532,	1532,	1532,	1532,	1532,
1532,	1532,	888,	888,	888,	888,	888,	888,	689,	689,
1546,	1546,	1546,	1546,	1546,	1896,	1896,	662,	662,	
450,	264,	1430,	1430,	1430,	1430,	1430,	1430,	1430,	1430,
1430,	1430,	1430,	1430,	30,	30,	1710,	1710,	1710,	
2234,	2101,	498,	432,	498,	498,	1677,	947,	947,	1677,
1677,	498,	947,	498,	1283,	498,	1677,	498,	530,	533,
960,	960,	960,	960,	960,	960,	1099,	960,	960,	1215,
475,	960,	960,	960,	1215,	960,	960,	960,	960,	960,
960,	960,	1553,	1309,	1553,	1553,	1309,	1309,	94,	1647,
94,	94,	94,	94,	94,	1647,	935,	935,	140,	508,
508,	1511,	95,	95,	95,	1306,	1169,	1431,	1431,	
1431,	1431,	1176,	1176,	1176,	1176,	1176,	1176,	1176,	1416,
1839,	1839,	1839,	1839,	2324,	1839,	2425,	1744,	723,	2506,
2506,	2506,	2506,	2506,	2506,	2506,	1823,	1823,	1079,	130,
1838,	1838,	508,	1838,	1838,	1838,	1838,	1838,	424,	424,
424,	424,	424,	424,	424,	424,	424,	424,	424,	582,
582,	1867,	582,	582,	582,	582,	2326,	1091,	1091,	1091,
1091,	1091,	1462,	1462,	1462,	1462,	1462,	1462,	850,	1462,
850,	850,	850,	1413,	2374,	2374,	740,	740,	740,	740,
572,	572,	1084,	415,	2426,	2426,	1134,	2426,	2426,	2426,
2426,	2426,	270,	270,	270,	270,	270,	270,	270,	
270,	270,	270,	270,	1165,	1165,	975,	32,	975,	975,
32,	975,	975,	975,	975,	975,	975,	975,	975,	975,
975,	975,	157,	1066,	160,	669,	160,	160,	160,	160,
461,	834,	834,	834,	292,	36,	156,	27,	27,	27,
27,	27,	27,	27,	27,	27,	645,	690,	690,	1237,
690,	1106,	1106,	690,	949,	550,	949,	550,	949,	550,
949,	949,	949,	949,	949,	949,	949,	949,	949,	949,
949,	949,	273,	273,	273,	273,	273,	273,	273,	273,
273,	273,	273,	273,	273,	273,	273,	273,	273,	273,
273,	273,	273,	273,	353,	353,	306,	306,	306,	396,
355,	355,	355,	192,	355,	355,	918,	918,	918,	918,
918,	918,	918,	918,	970,	970,	970,	970,	970,	970,
970,	490,	970,	970,	490,	490,	970,	490,	490,	490,
807,	273,	374,	118,	374,	374,	1237,	675,	1237,	675,
1144,	1144,	1144,	675,	273,	273,	1251,	1251,	1251,	1251,
1251,	1251,	1251,	1251,	1251,	1251,	1251,	1251,	1251,	1251,
11,	11,	1157,	1157,	1157,	1157,	784,	784,	784,	784,
784,	951,	1259,	951,	1012,	1012,	1012,	1012,	808,	1012,
812,	812,	812,	812,	812,	812,	1206,	1206,	1206,	1206,
734,	482,	482,	482,	1043,	1313,	166,	166,	166,	166,
675,	166,	166,	675,	166,	675,	675,	675,	675,	675,
675,	675,	166,	675,	675,	675,	166,	166,	968,	968,
968,	968,	968,	968,	1311,	1311,	538,	538,	538,	538,
538,	538,	582,	582,	582,	582,	347,	347,	347,	347,
347,	347,	21,	21,	21,	21,	21,	21,	21,	21,
21,	21,	21,	645,	645,	21,	29,	29,	1312,	1312,
1380,	1380,	1411,	1120,	1411,	1411,	306,	306,	806,	806,
669,	669,	669,	681,	1253,	1253,	876,	876,	1036,	1036,
1036,	1036,	1036,	1036,	233,	233,	233,	233,	233,	
233,	233,	233,	233,	233,	233,	271,	271,	271,	271,
271,	271,	871,	871,	871,	871,	871,	871,	1268,	1268,
1268,	1268,	1268,	1268,	1268,	1268,	1268,	1268,	1268,	1268,

1268,	1268,	1194,	1125,	1125,	1125,	353,	353,	353,	353,
353,	353,	865,	865,	913,	865,	865,	913,	865,	865,
865,	913,	913,	913,	409,	409,	409,	409,	409,	409,
409,	409,	954,	954,	954,	954,	954,	954,	159,	159,
159,	159,	585,	585,	585,	585,	585,	585,	585,	585,
585,	585,	585,	585,	585,	585,	585,	585,	585,	585,
585,	585,	585,	585,	585,	585,	585,	103,	529,	529,
529,	529,	529,	529,	529,	529,	529,	529,	529,	529,
529,	529,	529,	529,	529,	529,	871,	694,	310,	871,
871,	310,	310,	310,	310,	1526,	1194,	1194,	1194,	1194,
698,	1194,	1194,	1194,	770,	770,	770,	1103,	1367,	1367,
1367,	361,	1367,	1367,	850,	850,	1420,	1420,	1420,	1420,
1420,	1420,	353,	353,	353,	353,	353,	353,	514,	514,
514,	353,	353,	353,	353,	723,	514,	11,	723,	514,
11,	11,	723,	723,	538,	538,	538,	538,	538,	538,
538,	538,	538,	538,	538,	1379,	1379,	1523,	850,	850,
850,	850,	321,	850,	850,	850,	850,	1523,	1523,	1523,
1523,	1523,	1523,	1523,	1444,	1550,	1550,	1550,	70,	690,
690,	690,	415,	415,	415,	415,	415,	518,	725,	725,
415,	415,	415,	415,	1036,	353,	1036,	353,	353,	353,
1012,	1036,	353,	1036,	353,	353,	1671,	1671,	935,	935,
935,	935,	159,	935,	935,	935,	935,	935,	832,	832,
1283,	1283,	1283,	1283,	121,	121,	121,	121,	121,	121,
121,	121,	756,	756,	1380,	756,	756,	756,	1413,	1413,
1413,	1413,	1413,	1413,	308,	308,	481,	1219,	662,	662,
1512,	1512,	1512,	986,	21,	21,	1144,	1144,	21,	21,
920,	585,	585,	920,	920,	920,	920,	920,	871,	871,
871,	871,	582,	1333,	353,	353,	353,	353,	353,	353,
1244,	1244,	1244,	1244,	1244,	1244,	1244,	1244,	1244,	1244,
316,	316,	918,	918,	918,	918,	918,	918,	918,	918,
918,	1032,	1508,	671,	671,	671,	671,	819,	1437,	1437,
55,	960,	960,	960,	970,	481,	481,	1114,	1114,	1114,
1114,	481,	1114,	481,	1556,	1114,	1556,	1556,	627,	627,
627,	627,	627,	627,	627,	627,	627,	627,	871,	871,
871,	871,	871,	871,	871,	871,	871,	871,	871,	871,
1160,	1160,	1448,	361,	361,	396,	396,	361,	994,	994,
994,	994,	717,	717,	663,	663,	1016,	663,	663,	663,
663,	496,	663,	663,	663,	663,	161,	161,	230,	230,
230,	1348,	230,	230,	1149,	230,	230,	1348,	1348,	1348,
1348,	1348,	1197,	1197,	1197,	1197,	1429,	1429,	1429,	1429,
36,	36,	1707,	1707,	1707,	1707,	1707,	1707,	1707,	1707,
1586,	1586,	1174,	1174,	1174,	1586,	681,	1707,	681,	681,
1314,	1314,	1314,	1314,	1512,	1314,	1314,	1512,	1314,	31,
1512,	1314,	1314,	1512,	1314,	1314,	1314,	1314,	1434,	1434,
1434,	1434,	1434,	1434,	1434,	1434,	353,	1592,	1592,	1592,
1592,	1592,	30,	62,	62,	62,	62,	62,	1485,	1485,
1485,	740,	807,	1485,	963,	963,	963,	963,	832,	832,
963,	963,	1444,	1591,	1444,	1444,	1444,	1444,	1444,	1591,
1591,	1591,	842,	1586,	842,	842,	1586,	842,	842,	1586,
1586,	1776,	1586,	1586,	968,	968,	968,	968,	968,	968,
968,	968,	968,	968,	968,	968,	968,	968,	992,	1362,
992,	992,	983,	983,	983,	983,	983,	983,	1521,	1521,
1521,	1521,	1521,	1165,	1158,	1158,	91,	91,	91,	91,
91,	91,	91,	820,	91,	91,	1069,	1069,	1069,	1069,
91,	91,	91,	91,	91,	91,	91,	91,	91,	91,
91,	91,	91,	91,	1825,	1825,	1432,	1432,	1432,	1432,
1432,	1432,	1432,	1432,	1432,	1432,	30,	30,	30,	30,
30,	30,	30,	30,	1675,	1508,	1675,	1508,	1620,	1620,
1620,	1620,	1620,	1620,	1620,	1620,	1620,	1620,	1620,	1620,
1620,	1620,	1221,	1620,	1221,	935,	334,	334,	334,	231,
334,	334,	952,	952,	952,	952,	554,	952,	1164,	952,
952,	952,	1338,	952,	1832,	1832,	1832,	1832,	1832,	952,

355,	355,	355,	355,	355,	355,	355,	355,	355,	355,
355,	355,	1741,	383,	1741,	1741,	1748,	1748,	1748,	1748,
1748,	1748,	550,	550,	550,	550,	550,	550,	550,	550,
550,	550,	550,	1669,	1669,	1669,	1669,	1669,	550,	1669,
1669,	550,	1588,	1588,	1588,	1588,	1588,	1588,	1588,	1588,
1588,	1588,	1669,	1669,	368,	368,	368,	368,	1313,	1313,
368,	1313,	1313,	1313,	1313,	1313,	1221,	1221,	1221,	1221,
1221,	1221,	1221,	1221,	1221,	1221,	1221,	1221,	1221,	1221,
1221,	1221,	1221,	1221,	1221,	1221,	1221,	1221,	1221,	1221,
1221,	1221,	1671,	1671,	1671,	1671,	832,	832,	963,	963,
963,	963,	1822,	963,	1381,	963,	522,	963,	963,	522,
6,	6,	6,	6,	6,	6,	1009,	1009,	1009,	1009,
1839,	1839,	1839,	1707,	1839,	1839,	1707,	1839,	1839,	1707,
1839,	1707,	1827,	1916,	1089,	1457,	1827,	1089,	64,	64,
64,	64,	64,	64,	64,	64,	22,	22,	22,	22,
740,	1087,	740,	740,	35,	35,	740,	740,	1221,	1221,
1706,	1706,	1706,	1221,	1011,	1706,	1706,	1011,	1706,	1011,
1706,	1706,	1706,	1706,	1706,	1706,	1706,	1011,	1011,	1403,
265,	265,	121,	1099,	121,	1099,	582,	2018,	1972,	1373,
1373,	1972,	415,	415,	1256,	1256,	1256,	1936,	1936,	1256,
1797,	1797,	1797,	740,	740,	935,	935,	1797,	935,	1797,
935,	935,	1797,	1797,	1797,	1797,	1797,	1797,	1797,	1797,
6,	1797,	1797,	1797,	1797,	1797,	1797,	1797,	1795,	1795,
1727,	1727,	1727,	1727,	1727,	582,	375,	375,	375,	375,
1176,	67,	81,	1915,	81,	81,	81,	81,	740,	740,
740,	740,	1564,	1564,	1564,	1564,	158,	158,	136,	136,
136,	136,	1598,	1598,	1598,	1598,	1875,	1505,	1598,	1875,
1773,	1773,	1773,	175,	461,	780,	1952,	1952,	312,	312,
1219,	1219,	1219,	312,	312,	1952,	1947,	1727,	492,	30,
492,	492,	492,	492,	492,	388,	492,	492,	388,	388,
388,	388,	492,	492,	1382,	1382,	424,	424,	424,	424,
424,	424,	424,	424,	1592,	554,	424,	1592,	424,	424,
1592,	1592,	516,	516,	516,	516,	516,	516,	625,	625,
625,	625,	625,	625,	625,	625,	625,	625,	625,	625,
625,	625,	625,	625,	625,	625,	625,	625,	1809,	183,
1809,	183,	183,	183,	1809,	1562,	1562,	1562,	1562,	1562,
1562,	1562,	1562,	1779,	1779,	1779,	1779,	1779,	1779,	1779,
1779,	1158,	1158,	1158,	1158,	1592,	1592,	63,	63,	63,
63,	1448,	1448,	538,	751,	538,	538,	751,	751,	538,
538,	538,	751,	751,	751,	751,	751,	538,	751,	538,
538,	538,	538,	538,	1089,	1089,	1089,	1089,	1089,	1089,
1089,	1089,	1089,	948,	948,	948,	948,	948,	948,	948,
948,	948,	948,	948,	948,	1617,	1617,	1617,	1617,	1617,
1617,	1958,	1958,	1958,	1958,	1958,	1958,	1958,	1958,	1958,
1958,	409,	409,	1362,	1661,	1362,	1362,	2283,	850,	2283,
2283,	2283,	2283,	865,	1464,	1592,	2174,	1592,	784,	784,
784,	1060,	1060,	1060,	1060,	1060,	1060,	241,	1060,	241,
1060,	736,	736,	442,	442,	442,	442,	442,	442,	442,
442,	442,	442,	442,	442,	271,	271,	271,	271,	271,
271,	271,	271,	271,	271,	353,	353,	1222,	2224,	1222,
1222,	1222,	1222,	1222,	1222,	1222,	755,	755,	755,	755,
755,	755,	755,	1331,	1331,	1331,	1331,	1331,	1331,	1331,
1331,	1569,	1569,	1569,	1569,	1569,	1219,	2138,	1219,	1219,
1219,	35,	265,	409,	409,	409,	35,	22,	22,	812,
812,	154,	1023,	812,	409,	1916,	1916,	1916,	1916,	1916,
1916,	2047,	2047,	2047,	2047,	1448,	1448,	299,	299,	299,
299,	299,	299,	299,	299,	1745,	1745,	299,	299,	299,
299,	299,	299,	1764,	1764,	1764,	1764,	1764,	1764,	1764,
1764,	1764,	1764,	1764,	1514,	1514,	1514,	552,	552,	552,
552,	1687,	1687,	1687,	1687,	1687,	1687,	1687,	1687,	1687,
1687,	1687,	1687,	1687,	1687,	172,	172,	172,	172,	172,
172,	2115,	2115,	2115,	2115,	196,	1858,	196,	1858,	1858,

1858,	1858,	1858,	364,	364,	364,	364,	364,	364,	364,
364,	1517,	364,	646,	646,	646,	646,	646,	646,	646,
646,	128,	128,	128,	128,	128,	1163,	1163,	427,	
427,	427,	427,	1371,	1371,	1371,	1868,	1868,	1868,	960,
1868,	1868,	1371,	960,	694,	1371,	425,	694,	694,	694,
694,	425,	1540,	694,	694,	1466,	1466,	1466,	1466,	1466,
1466,	1604,	1604,	334,	334,	334,	334,	334,	334,	30,
30,	30,	30,	1851,	83,	83,	1629,	1851,	1629,	784,
1629,	1629,	1629,	1629,	1629,	1629,	391,	178,	178,	
2044,	398,	1947,	398,	398,	398,	1641,	1641,	1641,	
1641,	1641,	1641,	1641,	1641,	30,	30,	30,	30,	30,
1779,	30,	1779,	1779,	1779,	1707,	1707,	1128,	1128,	1128,
1128,	1128,	1128,	1128,	1128,	1128,	1128,	1569,	420,	1569,
420,	1569,	420,	420,	420,	420,	420,	420,	420,	420,
420,	1592,	2086,	1429,	1347,	2011,	2011,	2161,	1448,	1448,
1448,	1448,	1448,	1448,	1448,	1448,	1448,	1457,	1457,	1457,
1457,	1457,	1457,	1457,	1457,	1457,	1457,	1457,	1457,	1457,
1457,	998,	998,	998,	998,	749,	749,	749,	2305,	295,
2441,	2441,	2441,	159,	159,	886,	886,	886,	886,	159,
2201,	886,	886,	886,	159,	2201,	886,	2201,	2201,	886,
2201,	2346,	2131,	2131,	2131,	514,	2075,	2075,	2075,	2075,
2075,	1188,	1188,	1444,	1444,	1444,	1444,	2263,	2263,	2263,
2263,	2263,	2263,	1125,	1125,	933,	933,	933,	933,	933,
933,	364,	933,	1128,	933,	933,	933,	818,	1926,	1926,
1926,	1128,	2440,	1128,	1128,	1128,	1128,	1128,	1128,	1128,
1128,	1128,	1128,	1128,	1128,	1128,	1128,	1128,	1128,	1128,
1128,	2174,	1651,	1651,	1651,	1651,	1651,	1651,	2174,	2174,
1651,	1651,	1651,	1651,	1651,	1651,	1651,	1651,	1651,	22,
22,	22,	22,	2263,	22,	22,	2263,	2263,	22,	22,
22,	2548,	2548,	2665,	1998,	2427,	2053,	920,	920,	920,
920,	920,	36,	13,	19,	1790,	2471,	1197,	1197,	2502,
2502,	266,	266,	266,	266,	266,	521,	521,	521,	521,
521,	521,	521,	521,	521,	521,	521,	521,	521,	521,
521,	521,	521,	521,	521,	521,	521,	521,	521,	1418,
1418,	2232,	2232,	2232,	2232,	2232,	2232,	2806,	2806,	2806,
2806,	2878,	2806,	2806,	2878,	174,	2636,	2636,	2878,	1450,
2878,	2806,	2878,	253,	253,	253,	253,	253,	253,	979,
979,	979,	979,	979,	979,	979,	979,	979,	979,	979,
979,	552,	552,	295,	295,	295,	295,	295,	295,	1707,
2465,	1707,	1707,	1707,	978,	1707,	1707,	1707,	2465,	1707,
1707,	132,	132,	2842,	130,	132,	132,	130,	130,	130,
132,	132,	130,	132,	132,	132,	132,	1495,	1495,	1973,
1973,	1973,	1973,	723,	912,	2482,	2482,	2482,	723,	89,
89,	89,	89,	89,	89,	1671,	89,	89,	89,	1767,
89,	1009,	1009,	1177,	1177,	1177,	1177,	1177,	1177,	1177,
1177,	1177,	1177,	1177,	1177,	1177,	1177,	1177,	1177,	1177,
1744,	2076,	2076,	2076,	2076,	2811,	2076,	2076,	2076,	2811,
2076,	607,	2076,	2076,	2076,	2076,	2076,	2076,	2076,	607,
2076,	2076,	2076,	2076,	2076,	2306,	2306,	2306,	2306,	2306,
2306,	2306,	2306,	2306,	2306,	770,	1514,	1514,	770,	1514,
1514,	1421,	1616,	2110,	2110,	2110,	2110,	2110,	2110,	2110,
2110,	2110,	2110,	127,	127,	2521,	2521,	2521,	2521,	2521,
2521,	2521,	2521,	2521,	2521,	1588,	1588,	1505,	1505,	272,
272,	1505,	1505,	1505,	1505,	1505,	1505,	2961,	695,	695,
1174,	1174,	695,	2135,	2135,	838,	838,	2155,	2155,	2155,
2155,	2155,	2122,	2122,	2122,	465,	465,	583,	583,	583,
583,	583,	583,	583,	583,	583,	583,	2451,	1845,	1845,
1617,	2451,	1845,	978,	978,	978,	978,	978,	978,	978,
978,	1197,	1197,	1197,	1197,	1197,	1197,	1197,	1197,	1197,
1197,	1197,	1197,	1197,	1197,	1197,	1197,	3017,	1197,	1197,
1197,	3017,	1197,	3017,	1197,	1197,	1197,	1197,	1197,	1197,
1197,	2863,	2863,	2863,	2863,	2863,	2863,	2863,	2863,	2863,

2863,	852,	852,	1168,	1168,	1168,	1168,	1168,	1168,	2748,
1168,	1168,	1168,	1495,	1495,	1495,	1495,	1495,	1669,	1669,
1669,	1329,	1329,	1329,	1329,	1329,	1329,	1727,	1727,	2441,
1532,	1532,	1532,	1727,	2441,	1727,	1727,	550,	550,	550,
550,	550,	550,	550,	550,	550,	550,	550,	2263,	550,
2263,	550,	550,	550,	550,	353,	353,	353,	353,	353,
353,	2245,	1592,	1612,	2245,	1834,	2245,	1612,	1822,	1612,
2826,	2245,	1592,	477,	477,	477,	477,	477,	477,	477,
477,	477,	477,	477,	2724,	2724,	353,	209,	486,	
353,	1485,	2113,	353,	353,	2113,	1485,	353,	353,	353,
1485,	1485,	2113,	2113,	2113,	2811,	2811,	2811,	2811,	
2811,	2308,	1587,	3336,	1587,	1401,	1285,	1285,	1285,	1099,
1099,	1099,	1099,	1099,	1099,	21,	21,	21,	21,	21,
21,	21,	21,	21,	21,	21,	21,	21,	21,	21,
21,	21,	21,	2071,	2071,	1188,	1188,	1188,	1188,	1188,
1188,	1188,	1188,	2494,	2494,	2963,	2963,	2963,	2963,	2963,
2963,	2963,	2963,	2963,	2963,	2963,	2963,	2963,	2963,	398,
398,	398,	398,	398,	398,	3153,	3102,	3153,	169,	2089,
2089,	2273,	1530,	1530,	3307,	1222,	1222,	1222,	1222,	1222,
1222,	1222,	1222,	1222,	1222,	1222,	1222,	1222,	1222,	1222,
1222,	1222,	1222,	1222,	1222,	1222,	1222,	1222,	1222,	1118,
1118,	492,	492,	492,	492,	492,	1489,	1489,	1489,	555,
555,	2645,	555,	555,	555,	555,	555,	555,	555,	555,
555,	555,	555,	555,	2656,	2656,	2656,	2656,	2656,	2656,
1015,	1015,	1015,	2619,	2619,	2619,	2619,	2619,	2619,	1401,
1118,	1401,	2780,	1118,	1118,	1778,	1778,	1778,	1778,	1778,
1778,	1778,	1778,	1778,	1778,	1778,	1778,	1778,	1778,	1778,
1778,	1778,	1778,	1263,	1263,	1263,	1263,	1263,	1263,	1263,
1263,	1263,	1263,	1263,	1263,	1263,	1263,	3649,	1263,	2860,
2860,	2924,	1592,	1592,	3223,	1490,	9,	2620,	954,	1490,
954,	785,	785,	2627,	772,	772,	2627,	772,	772,	2347,
2347,	2347,	2347,	2347,	2347,	2534,	2534,	2534,	2534,	2534,
2534,	2534,	2534,	2534,	2534,	1260,	1260,	1260,	1260,	1260,
1260,	647,	647,	647,	647,	647,	647,	647,	647,	647,
647,	647,	647,	2282,	2282,	2282,	2282,	2282,	2282,	2282,
2282,	2282,	2282,	2282,	2282,	960,	960,	960,	960,	960,
960,	960,	960,	960,	960,	960,	960,	960,	960,	960,
960,	960,	960,	2263,	2263,	287,	287,	287,	287,	287,
287,	1797,	1797,	572,	1797,	1179,	1179,	1179,	1179,	1179,
1179,	1179,	1179,	1179,	1179,	1179,	1179,	1179,	1179,	1179,
1179,	3355,	1179,	784,	784,	784,	364,	1381,	1381,	1381,
364,	647,	647,	647,	647,	647,	647,	647,	647,	647,
647,	647,	647,	647,	647,	647,	647,	647,	647,	647,
647,	647,	647,	647,	647,	112,	112,	112,	2626,	1490,
2179,	2041,	2041,	2041,	2041,	481,	2094,	2094,	2094,	514,
2094,	2455,	2455,	582,	306,	306,	306,	306,	306,	306,
306,	306,	306,	306,	2076,	3103,	3103,	3103,	3103,	488,
488,	488,	488,	488,	488,	488,	488,	488,	488,	488,
488,	82,	82,	241,	241,	241,	241,	241,	241,	3535,
241,	241,	241,	241,	241,	241,	241,	241,	241,	241,
241,	3535,	241,	241,	3535,	241,	241,	241,	241,	241,
241,	241,	241,	891,	285,	285,	285,	1188,	1188,	1188,
1188,	1188,	1188,	3011,	1325,	1325,	567,	1325,	3549,	1367,
1367,	1367,	1367,	1367,	1367,	1367,	1367,	1367,	1367,	1367,
1367,	2665,	23,	23,	3017,	3017,	3017,	3017,	3017,	3017,
3017,	3017,	3017,	3017,	3017,	1423,	2897,	1423,	1423,	918,
918,	166,	166,	166,	166,	166,	166,	2815,	166,	166,
166,	2282,	2282,	2282,	2282,	2282,	2282,	2282,	2282,	2282,
2282,	2282,	2282,	2619,	2619,	2161,	2161,	2161,	2161,	2696,
2814,	3336,	3336,	2814,	2696,	2452,	2452,	2452,	3482,	2452,
1392,	2452,	2452,	1229,	1229,	2627,	303,	2263,	2263,	3346,
838,	838,	3346,	838,	3346,	838,	2583,	2583,	2583,	3232,

3232,	1596,	1720,	1720,	1720,	2044,	2044,	409,	2044,	2044,
2044,	2044,	2044,	409,	21,	409,	2044,	409,	2044,	21,
21,	21,	21,	21,	21,	3268,	3268,	1727,	1727,	3994,
3268,	1335,	1335,	1335,	1335,	3547,	3547,	393,	1069,	3547,
1260,	465,	465,	2041,	2656,	2656,	2656,	2656,	2656,	2656,
2656,	2656,	2656,	1355,	2185,	1260,	1260,	1260,	1260,	1260,
1260,	1260,	1260,	1260,	1571,	1778,	1778,	1778,	1571,	1260,
1260,	2294,	2294,	2294,	2294,	2294,	2294,	2294,	2294,	2294,
2294,	2294,	2294,	516,	2214,	1811,	1811,	1811,	1811,	1811,
1811,	420,	420,	420,	420,	420,	420,	420,	420,	420,
420,	420,	420,	420,	420,	420,	420,	1086,	1086,	1086,
1086,	1104,	1610,	1610,	1610,	1610,	1610,	1404,	1404,	1404,
1404,	1404,	1404,	1404,	1404,	1404,	1404,	1404,	3811,	4368,
4368,	1338,	1338,	1338,	1338,	1338,	1338,	1260,	1260,	550,
1260,	1260,	550,	920,	2161,	920,	2161,	920,	920,	3641,
3357,	3641,	3641,	3641,	3641,	3641,	3357,	3357,	3641,	3641,
3641,	3641,	3641,	3641,	998,	3641,	3641,	3641,	3641,	3641,
3641,	3641,	3641,	3641,	3641,	3641,	3641,	231,	231,	231,
231,	231,	231,	1457,	1457,	1457,	1457,	1457,	3525,	1457,
1457,	1457,	1457,	1457,	1457,	1457,	409,	409,	409,	1423,
409,	4075,	409,	2968,	2968,	2968,	2968,	2968,	2968,	2968,
2968,	2968,	2968,	3676,	3676,	2386,	3676,	3676,	3676,	3676,
3676,	1129,	1129,	1129,	1129,	1129,	1129,	1129,	1129,	1129,
1129,	1129,	1129,	1650,	1650,	1650,	1650,	4246,	4246,	4246,
4246,	1650,	1650,	1650,	4246,	4246,	4246,	4246,	4246,	4246,
4246,	2224,	2224,	2224,	425,	2220,	826,	663,	663,	663,
2423,	2707,	2707,	2707,	2707,	2707,	2707,	2707,	2707,	2707,
2707,	2707,	2707,	2707,	2707,	2707,	2707,	2707,	2707,	2707,
2707,	2707,	2707,	2707,	2707,	3178,	2519,	2519,	3178,	3178,
3178,	51,	3178,	51,	3178,	4507,	4507,	2437,	2437,	2521,
231,	231,	2521,	2854,	2854,	3019,	3019,	3611,	3611,	3019,
918,	3019,	3545,	3545,	3545,	3545,	3545,	3611,	3611,	3611,
3611,	2514,	2514,	2514,	2514,	2514,	2514,	4151,	4151,	4151,
4151,	4151,	4151,	586,	586,	4343,	2664,	2664,	2664,	586,
2664,	2664,	2664,	2679,	2664,	2664,	2664,	1017,	1423,	1423,
1423,	1017,	1423,	1017,	1017,	1423,	2665,	1888,	1888,	788,
788,	788,	788,	788,	788,	1888,	788,	788,	1888,	788,
788,	871,	871,	954,	871,	1477,	4453,	4241,	4453,	1477,
4453,	4453,	4241,	4453,	4453,	4241,	838,	4241,	4453,	4453,
4241,	4241,	3726,	838,	4241,	3726,	1477,	1477,	1477,	4968,
3726,	4968,	4968,	1477,	4968,	2444,	2444,	2444,	2444,	2444,
2444,	1429,	1429,	1429,	1429,	1429,	1429,	893,	893,	893,
893,	893,	893,	4101,	2263,	4101,	4101,	2263,	4101,	2263,
2263,	4173,	4173,	4173,	4173,	4173,	4173,	4620,	3269,	3461,
4394,	3830,	3830,	423,	423,	423,	423,	423,	423,	423,
423,	423,	423,	423,	423,	3220,	3220,	3220,	3220,	3220,
2081,	1599,	3780,	1599,	3780,	2311,	2311,	2632,	2632,	2632,
2632,	2632,	2632,	3360,	1158,	3360,	3085,	3085,	1158,	3360,
3085,	3085,	2464,	3085,	2464,	3085,	2464,	2464,	2464,	3085,
3085,	3085,	3085,	3085,	734,	3285,	920,	920,	920,	920,
920,	364,	364,	364,	364,	364,	364,	364,	364,	364,
364,	364,	364,	364,	364,	364,	364,	364,	364,	4978,
4978,	4131,	4131,	4131,	4131,	2903,	303,	303,	303,	1189,
303,	1189,	1189,	1189,	303,	1189,	303,	2263,	2263,	2599,
2599,	2599,	2599,	2599,	3029,	1060,	1060,	1060,	1060,	1727,
1727,	1727,	1727,	1727,	1727,	1727,	1727,	1727,	1727,	1727,
1727,	1727,	1727,	1727,	1727,	1727,	1727,	1727,	1727,	1727,
1727,	1727,	1727,	1727,	1727,	2831,	2728,	2831,	3421,	3421,
3421,	2582,	2582,	2582,	2582,	1333,	1965,	465,	465,	465,
465,	5182,	5182,	5182,	5182,	5182,	5182,	73,	5182,	398,
398,	3619,	398,	398,	398,	398,	398,	398,	398,	4554,
4554,	5110,	5110,	5110,	4554,	5110,	5110,	5110,	5110,	5110,

5110,	5110,	5110,	2907,	2907,	2907,	2907,	2907,	2907,	2907
2907,	2907,	2907,	2907,	2907,	5110,	2907,	2907,	2907,	2907
2907,	784,	784,	784,	3138,	3138,	784,	3138,	3138,	3138
1710,	3138,	1710,	3138,	3138,	1710,	1710,	1367,	1367,	4114
4114,	1614,	4114,	4114,	4114,	5082,	1778,	1778,	1778,	2562
2562,	780,	780,	780,	780,	2843,	2843,	4642,	3355,	3355
3355,	3355,	3355,	3355,	4478,	3355,	3355,	2665,	2665,	4274
4274,	4274,	2665,	4274,	4274,	4274,	4274,	4274,	4274,	4274
4274,	325,	325,	325,	325,	325,	325,	1222,	1222,	1222
1222,	2665,	2665,	2665,	2665,	3351,	2665,	2665,	2665,	2359
1053,	2359,	2294,	2359,	3599,	3599,	2359,	2359,	2359,	2831
2831,	3672,	2831,	3672,	1335,	2160,	2160,	2160,	4009,	4240
4240,	4240,	4240,	4240,	4240,	4009,	4240,	4240,	4240,	4009
4240,	4009,	4240,	2160,	2160,	2386,	681,	681,	2386,	3270
3270,	2631,	2631,	2631,	2631,	2631,	2631,	2434,	2434,	2434
2434,	2434,	2434,	2434,	2434,	2434,	2434,	2434,	2434,	2434
2434,	2434,	2434,	2434,	2434,	2434,	1514,	2434,	2434,	2434
2434,	2434,	2434,	2434,	2434,	2434,	4465,	3926,	3926,	3926
3926,	3926,	3926,	3926,	3926,	1275,	266,	266,	1275,	1275
1275,	266,	1275,	266,	1275,	5397,	1879,	1879,	1879,	1879
1879,	1331,	681,	681,	1331,	1275,	1275,	5208,	5208,	5208
5208,	3399,	3399,	3399,	3399,	5208,	327,	327,	327,	327
327,	496,	496,	496,	496,	496,	496,	496,	496,	496
496,	496,	496,	691,	4587,	4642,	691,	3767,	2607,	2607
4527,	4527,	3767,	3946,	3946,	960,	2194,	2194,	2194,	2194
960,	4282,	55,	55,	55,	920,	550,	920,	920,	957
920,	822,	822,	5031,	5031,	5031,	5031,	5031,	5031,	5031
4850,	5031,	5031,	5164,	5332,	3277,	3277,	3277,	3277,	3277
3277,	3277,	3277,	3277,	3277,	3277,	3277,	3277,	3277,	3277
3277,	2116,	2116,	2116,	4467,	4418,	4467,	1896,	1596,	1596
1596,	1596,	1596,	1596,	1896,	1896,	4129,	4129,	4129,	4129
4129,	4129,	4129,	4129,	4129,	4129,	4129,	1780,	1780,	1780
1780,	1367,	1367,	1367,	2162,	1367,	2162,	2162,	2162,	2162
2162,	2162,	2162,	238,	238,	238,	238,	238,	238,	238
238,	238,	238,	238,	238,	238,	238,	3830,	3830,	3533
3533,	3533,	3732,	3533,	3533,	3533,	3533,	3533,	3533,	3533
3533,	3533,	3533,	3533,	3533,	3533,	3533,	3533,	3533,	3533
3533,	3533,	3533,	3470,	3533,	4588,	355,	3591,	3591,	355
355,	355,	355,	355,	355,	355,	355,	355,	355,	355
355,	355,	355,	355,	355,	3268,	1877,	1877,	1877,	4001
4001,	4001,	4001,	4001,	4001,	4001,	4001,	4001,	4001,	4001
4001,	4001,	4001,	4001,	4001,	4001,	4001,	656,	656,	4713
4713,	656,	4713,	4713,	4713,	322,	322,	322,	322,	322
322,	4218,	4218,	4218,	4218,	5182,	5182,	2845,	5182,	5182
5182,	4518,	4518,	4518,	2069,	4518,	4518,	4518,	4518,	4518
2069,	2069,	2069,	2069,	2069,	4650,	759,	4650,	759,	759
4650,	552,	552,	552,	552,	552,	552,	2331,	2331,	2331
2331,	2331,	2331,	2331,	2331,	2331,	2331,	6187,	6187,	5172
5842,	5842,	5842,	5842,	5842,	5842,	5842,	5842,	5842,	2162
2162,	2162,	2162,	2162,	5031,	2162,	2162,	2162,	2162,	2162
2162,	3832,	2386,	5945,	6126,	3832,	5945,	5945,	5945,	1283
1283,	1283,	1283,	1283,	1283,	1283,	1283,	1283,	1592,	4352
4352,	3349,	3349,	6917,	6917,	3349,	6917,	3349,	2751,	3349
3349,	6135,	2837,	2837,	6135,	6135,	6135,	2794,	6135,	4780
4780,	4780,	4780,	4780,	4780,	4780,	4780,	4780,	4780,	4780
4780,	1060,	1060,	1060,	1060,	1060,	2764,	1060,	1060,	1060
1060,	1489,	308,	1423,	1423,	1423,	1423,	1423,	1423,	1423
1423,	1423,	1423,	1423,	1423,	1423,	1423,	1423,	1423,	1423
1423,	1423,	1423,	1423,	1423,	4558,	4330,	1654,	1654,	1654
1654,	5061,	5061,	5061,	5061,	1425,	5061,	1425,	1425,	971
971,	971,	716,	971,	971,	2814,	2616,	2814,	2814,	308
308,	308,	308,	308,	308,	308,	308,	5380,	785,	5380,

5380,	5380,	5380,	5380,	1592,	5380,	5380,	5380,	5380,	5380,
5380,	5380,	3464,	4468,	4468,	2244,	2244,	2244,	2244,	2244,
2244,	2244,	2244,	2244,	2244,	3880,	3880,	3880,	3880,	3880
3880,	60,	60,	60,	60,	60,	60,	734,	734,	5647
5647,	5647,	5647,	5647,	3145,	6661,	3145,	3145,	3145,	3145
3145,	3145,	3145,	3145,	2071,	2071,	2071,	2071,	2071,	2071
2071,	4680,	4680,	676,	676,	676,	676,	676,	676,	676
676,	676,	676,	2641,	2641,	2641,	2641,	2641,	2641,	6532
6532,	6532,	6532,	6532,	5098,	5098,	5098,	5098,	5098,	1628
5098,	1592,	1592,	1592,	1592,	1592,	1592,	1592,	1592,	55
6221,	6221,	6221,	6221,	3977,	550,	550,	550,	550,	550
550,	3977,	550,	550,	3977,	4055,	3651,	3651,	3651,	3651
550,	21,	21,	2639,	2639,	2639,	2639,	2639,	2639,	863
863,	863,	863,	863,	1086,	4557,	4557,	4557,	4557,	5449
5449,	277,	1497,	1497,	277,	1497,	1497,	277,	277,	1497
277,	2432,	2432,	2432,	2432,	2432,	2432,	2432,	2432,	2603
2603,	3041,	3041,	3041,	3041,	2603,	3041,	3041,	2702,	2702
2603,	2603,	2603,	2603,	2603,	3041,	3041,	3041,	3041,	2702
3041,	3041,	3041,	6043,	6043,	6043,	6043,	6043,	6043,	2365
784,	784,	2365,	2365,	2365,	2365,	2365,	2365,	784,	784
784,	2365,	784,	784,	4718,	4718,	4718,	4718,	4718,	4718
4718,	55,	681,	681,	5658,	5658,	5658,	5658,	5658,	681
5658,	681,	5658,	5658,	5658,	681,	681,	5658,	5658,	5658
6510,	6510,	6510,	7236,	5014,	5014,	5014,	5014,	5014,	5014
5014,	5014,	4116,	4116,	4116,	4116,	4116,	4116,	4116,	4116
4116,	4116,	4885,	5997,	4885,	4885,	5997,	5997,	4885,	4885
4885,	7266,	7266,	7266,	7266,	7266,	4885,	4885,	5997,	
4885,	4885,	5997,	4885,	4885,	4885,	4885,	4885,	4885,	
4885,	4885,	327,	327,	327,	327,	327,	327,	327,	
4650,	4650,	4650,	2396,	4650,	4650,	4650,	4650,	4650,	
4650,	3676,	4650,	4650,	3676,	4650,	4650,	4650,	6481,	
6481,	7188,	1195,	4368,	5475,	4518,	4518,	4518,	4518,	
4518,	4518,	4518,	4518,	425,	425,	425,	425,	4272,	
6009,	6009,	4650,	4650,	4650,	4650,	4650,	4650,	1267,	
1267,	1267,	1229,	1229,	1229,	1229,	1229,	1229,	1229,	
1229,	1229,	1229,	1229,	3357,	1229,	1229,	1229,	3357,	
6476,	6476,	6476,	6476,	6476,	6476,	6476,	6476,	6305,	
6305,	6305,	6305,	6305,	6305,	6305,	6305,	6305,	6305,	
8903,	8903,	8903,	8903,	36,	8903,	36,	36,	36,	
36,	36,	36,	36,	36,	8903,	425,	425,	425,	
425,	425,	425,	425,	425,	425,	425,	425,	425,	
425,	425,	6380,	6380,	6380,	367,	6380,	2877,	1916,	
4512,	4195,	4195,	4512,	4195,	4195,	4195,	4512,	4512,	
4771,	4771,	7692,	7692,	7692,	7692,	7692,	1839,	1839,	
1839,	1839,	5201,	1839,	5201,	5201,	5201,	1953,	1413,	
5740,	5957,	5740,	5740,	5740,	5957,	5957,	5957,	5957,	
3017,	4290,	759,	759,	759,	1066,	1066,	759,	6569,	
6569,	7782,	6569,	6569,	6569,	6569,	7782,	7782,	3651,	
3651,	3651,	427,	427,	427,	3651,	427,	427,	427,	
427,	427,	2622,	2107,	2107,	2107,	7035,	7035,	7035,	
7334,	7035,	7334,	7334,	7035,	7035,	7035,	7035,	7035,	
7035,	6186,	3300,	3300,	7031,	3300,	3300,	7031,	3300,	
3300,	3300,	3300,	3300,	7031,	3300,	3300,	3300,	6835,	
5164,	5164,	5164,	7092,	7092,	5164,	5164,	5164,	5164,	
3722,	3722,	869,	869,	869,	869,	869,	869,	869,	
869,	869,	5122,	5122,	5122,	5122,	5122,	4840,	5122,	
5122,	5122,	5122,	5122,	5122,	5122,	4840,	4840,	5122,	
3743,	5122,	5658,	5658,	5658,	5658,	2388,	5475,	4805,	
4805,	4805,	2597,	2597,	2597,	6009,	6009,	1778,	1778,	
1778,	1778,	1778,	1778,	1778,	1778,	1778,	1778,	1778,	
4645,	1778,	1778,	1778,	1778,	2751,	4364,	4364,	4094,	
4094,	4094,	1222,	1222,	1222,	1222,	1222,	5041,	5041,	

5110,	5110,	5110,	5110,	1335,	5110,	5110,	5110,	5110,	5110,
383,	5110,	3610,	3610,	3610,	3610,	3610,	3610,	3610,	3610,
7965,	7965,	7965,	7965,	7965,	3610,	7965,	3610,	2814,	2814,
2814,	8731,	2814,	2814,	2814,	2814,	1335,	1026,	1026,	1026,
6683,	2129,	6683,	6683,	2129,	2129,	827,	827,	827,	827,
	827,	827,	827,	827,	827,	827,	827,	827,	827,
	827,	827,	827,	827,	827,	827,	827,	827,	827,
5723,	5723,	5723,	5723,	5723,	5017,	4055,	4055,	3756,	3756,
3756,	3756,	3756,	3756,	3756,	3756,	3756,	3756,	616,	6933,
7871,	7871,	7871,	7871,	7871,	7871,	5846,	5846,	5846,	5846,
2019,	2019,	2019,	2019,	2019,	2019,	5261,	5261,	5261,	5261,
5261,	5261,	1737,	1737,	2041,	1737,	6532,	6532,	6532,	5676,
442,	442,	442,	1859,	1859,	442,	3101,	3101,	3101,	3101,
4143,	4143,	4143,	4143,	4143,	4143,	4143,	4143,	4143,	4143,
4143,	4143,	5726,	3102,	3102,	3102,	3102,	3102,	4880,	4880,
4880,	4880,	4880,	2595,	2595,	7052,	7052,	7052,	7052,	4885,
7052,	7052,	7052,	7052,	5177,	7052,	7052,	7052,	691,	691,
691,	691,	2000,	2000,	2000,	2000,	2000,	2000,	5999,	2000,
2000,	2000,	2000,	2000,	2000,	2000,	3692,	3692,	3692,	3692,
3692,	3692,	6111,	6111,	3692,	6111,	3692,	3692,	5741,	5741,
5741,	5741,	5741,	5741,	5741,	5741,	5741,	5741,	8670,	10473,
8670,	8670,	8670,	10473,	8670,	4035,	4035,	8670,	5261,	8670,
4035,	4035,	5261,	4035,	4035,	8670,	4035,	5261,	5261,	4035,
6497,	10666,	734,	734,	734,	734,	734,	734,	9318,	9318,
8980,	8980,	9155,	9155,	9155,	9318,	5991,	9318,	314,	9318,
2378,	2378,	3102,	2378,	3102,	2378,	2854,	5416,	2889,	2889,
2889,	2889,	2889,	2889,	2889,	2889,	2889,	2889,	2889,	2889,
2889,	2889,	2889,	2889,	2931,	2931,	2889,	2889,	2889,	2889,
4240,	4240,	4240,	4240,	4240,	2162,	9862,	2162,	7246,	7246,
7246,	7246,	7434,	7246,	7246,	7246,	7246,	7246,	7246,	7246,
7246,	7246,	7246,	7246,	7246,	7246,	4333,	4333,	4333,	4333,
4333,	4333,	4333,	4333,	4333,	4333,	5244,	5244,	5244,	5244,
5244,	5244,	51,	51,	51,	51,	51,	51,	51,	51,
	51,	51,	51,	51,	51,	6447,	6810,	6810,	6447,
2921,	2921,	2046,	2046,	3424,	2046,	2046,	2046,	6599,	1911,
6599,	6599,	1911,	1911,	6599,	1911,	6599,	1911,	327,	327,
327,	327,	327,	327,	327,	327,	2678,	4094,	4094,	4094,
4094,	4094,	5633,	5633,	5633,	5985,	7152,	7152,	7152,	7152,
7152,	7152,	8279,	5283,	5283,	32,	5283,	32,	32,	32,
	32,	32,	32,	32,	32,	32,	32,	32,	32,
	32,	32,	32,	32,	32,	5283,	32,	32,	32,
	32,	32,	3634,	3634,	3634,	1158,	3634,	3634,	3634,
3634,	3634,	3634,	3634,	3634,	3634,	3101,	3101,	3101,	1392,
1392,	1392,	3101,	3101,	3101,	1392,	4929,	4929,	4929,	4929,
4929,	4929,	4929,	4929,	4929,	4929,	4929,	4929,	1803,	1803,
1803,	9232,	9232,	9232,	9232,	1803,	9232,	9232,	7228,	7228,
2129,	3435,	2129,	2129,	2129,	2641,	9158,	3351,	2641,	9158,
9158,	3351,	7167,	9158,	2129,	2129,	10358,	10358,	4330,	4330,
3814,	4330,	6012,	6012,	6012,	6012,	6012,	3814,	3814,	6012,
6012,	4330,	4330,	7264,	4330,	6012,	1446,	1446,	1446,	1446,
1446,	1446,	1446,	1446,	1446,	1446,	1446,	1446,	11793,	1446,
1446,	1446,	1446,	1446,	1446,	1446,	1446,	1446,	1446,	1446,
8729,	8729,	427,	8729,	8729,	8729,	8729,	427,	8729,	8729,
8729,	8729,	8729,	8729,	8729,	8729,	7741,	7741,	7741,	7741,
4400,	4400,	4400,	4400,	4400,	4400,	4400,	4400,	4400,	4400,
4400,	4400,	4400,	4400,	4400,	4400,	11052,	5355,	5355,	11052,
11052,	11052,	11052,	5355,	11052,	11052,	11052,	11052,	11052,	11052,
11052,	11052,	11052,	11052,	3090,	2757,	3090,	3090,	5310,	5310,
7829,	7829,	4155,	4155,	5740,	5740,	4155,	5740,	4155,	5740,
4155,	4155,	4155,	4155,	4155,	4155,	4155,	4155,	5740,	5740,
3017,	3017,	3017,	3017,	5920,	2420,	4001,	5920,	4001,	4001,
1641,	1641,	6807,	6807,	6807,	6807,	6807,	6807,	6807,	6807,

6807,	6807,	4460,	4460,	4460,	4460,	4460,	4460,	4460,	4460,	4460,
9860,	9860,	9860,	9860,	9860,	9860,	9860,	9860,	9860,	9860,	9860,
6596,	6596,	6596,	6596,	5190,	5190,	2599,	3921,	2599,	2599,	2599,
6569,	6569,	8982,	8982,	8982,	8982,	8982,	8982,	8982,	8982,	8982,
10063,	1655,	10063,	10063,	11787,	11787,	11787,	11787,	11787,	11787,	11787,
85,	85,	550,	550,	550,	550,	550,	550,	550,	550,	550,
550,	550,	4328,	1089,	10623,	10623,	10623,	10623,	10623,	10623,	10623,
10623,	10623,	10623,	10623,	8757,	8757,	5876,	6540,	4682,	4682,	4682,
2308,	2308,	6219,	2308,	2308,	4664,	3360,	3360,	3360,	3360,	3360,
3360,	3360,	7152,	7152,	6024,	6024,	6024,	6024,	6024,	6024,	6024,
6024,	6024,	6024,	6024,	6024,	6024,	5319,	5319,	5319,	5319,	5319,
713,	713,	713,	713,	713,	713,	10037,	713,	10037,	10037,	10037,
10037,	10037,	10950,	10900,	10950,	10950,	10900,	10950,	10950,	10950,	10950,
10950,	10950,	10950,	10950,	10950,	10950,	10950,	10950,	10950,	10950,	10950,
4518,	4518,	4518,	4518,	2263,	2263,	9473,	9473,	9473,	9473,	7371,
3818,	3818,	3818,	3818,	3818,	7505,	3818,	3818,	3818,	3818,	3818,
7505,	3818,	3818,	3818,	7505,	3818,	3818,	3818,	3818,	3818,	3818,
6810,	6810,	6810,	6810,	1790,	1118,	1118,	1790,	1118,	1790,	1790,
1118,	1790,	8982,	8982,	8982,	8982,	1165,	8982,	9317,	9317,	9317,
9317,	9317,	173,	173,	173,	173,	173,	173,	173,	173,	173,
5069,	5069,	5069,	5069,	8058,	8058,	8058,	8058,	8058,	8058,	8058,
8058,	8058,	8058,	8058,	8058,	8058,	8058,	8058,	6230,	6574,	6574,
6574,	6574,	6574,	6574,	6574,	6574,	6574,	6574,	6574,	6574,	6574,
3101,	3101,	3101,	3101,	3101,	3101,	3101,	3101,	3101,	3101,	3101,
3101,	3101,	7502,	7502,	7502,	7502,	6313,	6313,	4518,	4518,	4518,
4518,	1606,	1606,	1606,	1606,	1606,	9284,	9284,	9284,	9284,	9837,
3830,	3269,	3269,	1606,	1606,	1606,	9284,	1606,	6912,	9284,	9284,
6912,	6912,	9284,	9284,	1606,	9284,	6545,	9284,	8502,	8502,	8502,
8502,	8502,	398,	398,	398,	398,	398,	398,	398,	398,	398,
398,	398,	398,	398,	398,	398,	398,	398,	398,	398,	398,
398,	398,	398,	398,	398,	398,	9158,	5275,	5275,	5275,	5275,
5275,	5275,	8169,	8169,	8169,	8169,	8169,	8169,	8962,	11967,	11967,
11967,	11967,	8962,	8962,	8962,	8962,	11967,	11967,	8962,	8962,	8962,
12618,	12618,	12618,	12618,	12618,	12618,	12618,	12618,	12618,	12618,	12618,
12618,	12618,	7414,	7414,	7414,	7414,	7414,	7414,	7414,	7414,	7414,
7414,	7414,	7414,	7414,	7414,	7414,	1426,	1426,	1426,	1426,	1426,
1426,	1426,	7020,	8968,	8731,	8968,	6251,	6251,	327,	327,	327,
327,	327,	3649,	3649,	3649,	4765,	4765,	4765,	4765,	4765,	7688,
4765,	7688,	4765,	4765,	4765,	4765,	4765,	4765,	3267,	7688,	7688,
4682,	3115,	3115,	3115,	3115,	3115,	5929,	5929,	5929,	5929,	5929,
5929,	5929,	5929,	5929,	5929,	5929,	9332,	5929,	5321,	5321,	5321,
5321,	5321,	5321,	5321,	5321,	8753,	4410,	4410,	4410,	4410,	4410,
4410,	4410,	3300,	3300,	3300,	3300,	4069,	4069,	4069,	818,	818,
818,	818,	4069,	4069,	4069,	4069,	4069,	10900,	656,	656,	656,
3830,	3830,	3830,	3830,	3830,	3830,	3830,	3830,	3830,	3830,	3830,
3830,	3830,	9632,	9632,	9632,	9632,	9632,	7871,	9632,	7871,	7871,
9632,	9632,	7871,	7871,	7871,	7871,	7871,	9632,	7871,	7871,	7871,
7871,	7871,	7871,	7871,	7871,	7871,	7871,	7871,	7871,	7871,	7871,
7871,	7871,	5261,	10356,	5261,	5261,	5261,	5261,	5261,	5261,	5261,
5261,	5261,	5261,	5261,	6979,	10356,	10010,	10356,	5846,	10089,	10089,
5267,	5267,	5267,	5267,	4823,	5267,	10958,	11674,	11674,	11674,	11674,
11674,	11674,	11674,	11674,	10958,	10958,	11674,	11674,	11674,	11674,	11674,
8410,	11674,	8410,	734,	734,	734,	1885,	8410,	7020,	7020,	7020,
7020,	7020,	7020,	7020,	7020,	7020,	7020,	7020,	7020,	7020,	7020,
7020,	7020,	1055,	1055,	1055,	1055,	1055,	1055,	7167,	7167,	7167,
7167,	7167,	7167,	7167,	7167,	7167,	7167,	7167,	8443,	8443,	8443,
7471,	7471,	7471,	7471,	8443,	8443,	7471,	8443,	8443,	8443,	8443,
3269,	3269,	3269,	3269,	3269,	3269,	8286,	8286,	5925,	5925,	5925,
5925,	5925,	4901,	4901,	4901,	4901,	4901,	4901,	4901,	4901,	4901,
13343,	13343,	13343,	13343,	13343,	13343,	13343,	13343,	13343,	13343,	13343,
8111,	8111,	8111,	8111,	8952,	6978,	2484,	7742,	2484,	7742,	7742,
2484,	7742,	1953,	1953,	1953,	1953,	1953,	1953,	1953,	1953,	1953,

1953, 1953, 1953, 1953, 1953, 1953, 1953, 1953, 1953, 1953, 1953,  
1953, 1953, 1953, 1964, 1201, 1964, 1501, 1501, 1501, 1501, 1501, 1501,  
1501, 1501, 1501, 1501, 1501, 1501, 1501, 1501, 1501, 1501, 1501,  
9497, 9497, 9497, 9497, 9497, 9497, 9497, 7751, 7751, 7751, 7751, 7751,  
13738, 13738, 13738, 13738, 13738, 13738, 13738, 13738, 13738, 2387, 2387,  
2387, 2387, 2387, 2387, 10089, 10245, 10245, 10245, 10245, 10245, 10245,  
14486, 767, 8427, 767, 767, 8427, 1099, 8427, 8427, 8427, 8427,  
8427, 8427, 767, 8427, 767, 767, 767, 767, 450, 15435,  
8482, 8482, 8482, 4195, 18171, 15435, 15435, 450, 6203, 6203,  
13051, 13051, 6143, 13051, 13051, 13051, 13051, 7956, 7218, 7218,  
14111, 14111, 8643, 8643, 8643, 8643, 5832, 5832, 5832, 5832,  
5832, 5832, 12089, 12089, 12089, 12089, 12089, 12089, 12089, 12089, 12089,  
12089, 12089, 12089, 12089, 12089, 12089, 12089, 12089, 12089,  
12089, 12089, 585, 585, 585, 585, 585, 585, 585, 6468, 6468,  
6468, 6468, 795, 10272, 10272, 1620, 10272, 12097, 12097, 12097,  
12097, 12097, 10272, 12097, 10272, 12097, 1997, 5561, 5474, 5474,  
14969, 14969, 11391, 11391, 11391, 11391, 2518, 2518, 2518, 2518,  
2518, 2518, 2518, 2518, 2518, 2518, 2518, 2518, 2518,  
6594, 6594, 6594, 6594, 6594, 6594, 6170, 16423, 6170, 6170,  
16423, 16423, 16423, 6170, 16423, 16423, 6170, 16423, 5601, 5601,  
5601, 5601, 5601, 5601, 5601, 1934, 5601, 5601, 5601,  
5601, 5601, 5601, 5601, 5601, 5601, 5601, 8349,  
8349, 8349, 10612, 3223, 10612, 10612, 10612, 10612, 10612, 10612, 7965,  
10612, 3223, 3830, 3830, 3830, 3830, 3830, 3830, 10583, 10583,  
10583, 10583, 10583, 10583, 516, 10583, 11222, 11222, 11222, 11222, 11222, 1373,  
11222, 11222, 11222, 11222, 11222, 11222, 7871, 7871, 7871, 17470, 17470,  
17470, 17470, 17470, 17470, 17470, 17470, 17470, 17470, 17470,  
17470, 8502, 17470, 17431, 17470, 17470, 17431, 17431, 17431, 17431, 17431, 17470,  
17470, 17431, 17470, 17431, 17470, 17470, 17431, 17431, 17431, 17470

### 附录 B.1 表 $V_0$ 的值

这些值表示在 B.5.4.1 中的上述应用中所述的表  $V_0$  的值的示例集合。每项是以十进制表示的 32 比特整数。这些值应从第一列由上至下、然后第二列由上至下的读取，以此类推。

251291136	2581671944	1521339547	2068983570	2859178611
3952231631	3312220480	3041843489	2247491078	3284308411
3370958628	681232419	420130494	3669524410	3819792700
4070167936	307306866	10677091	1575146607	3557526733
123631495	4112503940	515623176	828029864	451874476
3351110283	1158111502	3457502702	3732001371	1740576081
3218676425	709227802	2115821274	3422026452	3592838701
2011642291	2724140433	2720124766	3370954177	1709429513
774603218	4201101115	3242576090	4006626915	3702918379
2402805061	4215970289	854310108	543812220	3533351328
1004366930	4048876515	425973987	1243116171	1641660745
1843948209	3031661061	325832382	3928372514	179350258
428891132	1909085522	1796851292	2791443445	2380520112
3746331984	510985033	2462744411	4081325272	3936163904
1591258008	1361682810	1976681690	2280435605	3685256204
3067016507	129243379	1408671665	885616073	3156252216
1433388735	3142379587	1228817808	616452097	1854258901
504005498	2569842483	3917210003	3188863436	2861641019
2032657933	3033268270	263976645	2780382310	3176611298
3419319784	1658118006	2593736473	2340014831	834787554
2805686246	932109358	2471651269	1208439576	331353807
3102436986	1982290045	4291353919	258356309	517858103
3808671154	2983082771	650792940	3837963200	3010168884
2501582075	3007670818	1191583883	2075009450	4012642001
3978944421	3448104768	3046561335	3214181212	2217188075
246043949	683749698	2466530435	3303882142	3756943137
4016898363	778296777	2545983082	880813252	3077882590
649743608	1399125101	969168436	1355575717	2054995199
1974987508	1939403708	2019348792	207231484	3081443129
2651273766	1692176003	2268075521	2420803184	3895398812
2357956801	3868299200	1169345068	358923368	1141097543
689605112	1422476658	3250240009	1617557768	2376261053
715807172	593093658	3963499681	3272161958	2626898255
2722736134	1878973865	2560755113	1771154147	2554703076
191939188	2526292949	911182396	2842106362	401233789
3535520147	1591602827	760842409	1751209208	1460049922
3277019569	3986158854	3569308693	1421030790	678083952
1470435941	3964389521	2687243553	658316681	1064990737
3763101702	2695031039	381854665	194065839	940909784
3232409631	1942050155	2613828404	3241510581	1673396780
122701163	424618399	2761078866	38625260	528881783
3920852693	1347204291	1456668111	301875395	1712547446
782246947	2669179716	883760091	4176141739	3629685652
372121310	2434425874	3294951678	297312930	1358307511
2995604341	2540801947	1604598575	2137802113	
2045698575	1384069776	1985308198	1502984205	
2332962102	4123580443	1014570543	3669376622	
4005368743	1523670218	2724959607	3728477036	
218596347	2708475297	3062518035	234652930	
3415381967	1046771089	3115293053	2213589897	
4207612806	2229796016	138853680	2734638932	
861117671	1255426612	4160398285	1129721478	
3676575285	4213663089	3322241130	3187422815	

## 附录 B.2 表 $V_1$ 的值

这些值表示在 B.5.4.1 中的上述应用中所述的表  $V_1$  的值的示例集合。每项是以十进制表示的 32 比特整数。这些值应从第一列由上至下、然后第二列由上至下的读取，以此类推。

807385413	3843885867	3545667983	644189126	922673938
2043073223	4201106668	332038910	226475395	3877430102
3336749796	415906198	976628269	307789415	3422391938
1302105833	19296841	3123492423	1196105631	1414347295
2278607931	2402488407	3041418372	3191691839	1971054608
541015020	2137119134	2258059298	782852669	3061798054
1684564270	1744097284	2139377204	1608507813	830555096
372709334	579965637	3243642973	1847685900	2822905141
3508252125	2037662632	3226247917	4069766876	167033190
1768346005	852173610	3674004636	3931548641	1079139428
1270451292	2681403713	2698992189	2526471011	4210126723
2603029534	1047144830	3453843574	766865139	3593797804
2049387273	2982173936	1963216666	2115084288	429192890
3891424859	910285038	3509855005	4259411376	372093950
2152948345	4187576520	2358481858	3323683436	1779187770
4114760273	2589870048	747331248	568512177	3312189287
915180310	989448887	1957348676	3736601419	204349348
3754787998	3292758024	1097574450	1800276898	452421568
700503826	506322719	2435697214	4012458395	2800540462
2131559305	176010738	3870972145	1823982	3733109044
1308908630	1865471968	1888833893	27980198	1235082423
224437350	2619324712	2914085525	2023839966	1765319556
4065424007	564829442	4161315584	869505096	3174729780
3638665944	1996870325	1273113343	431161506	3762994475
1679385496	339697593	3269644828	1024804023	3171962488
3431345226	4071072948	3681293816	1853869307	442160826
1779595665	3618966336	412536684	3393537983	198349622
3068494238	2111320126	1156034077	1500703614	45942637
1424062773	1093955153	3823026442	3019471560	1324086311
1033448464	957978696	1066971017	1351086955	2901868599
4050396853	892010560	3598330293	3096933631	678860040
3302235057	1854601078	1979273937	3034634988	3812229107
420600373	1873407527	2079029895	2544598006	19936821
2868446243	2498544695	1195045909	1230942551	1119590141
311689386	2694156259	1071986421	3362230798	3640121682
259047959	1927339682	2712821515	159984793	3545931032
4057180909	1650555729	3377754595	491590373	2102949142
1575367248	183933047	2184151095	3993872886	2828208598
4151214153	3061444337	750918864	3681855622	3603378023
110249784	2067387204	2585729879	903593547	4135048896
3006865921	228962564	4249895712	3535062472	
4293710613	3904109414	1832579367	1799803217	
3501256572	1595995433	1192240192	772984149	
998007483	1780701372	946734366	895863112	
499288295	2463145963	31230688	1899036275	
1205710710	307281463	3174399083	4187322100	
2997199489	3237929991	3549375728	101856048	
640417429	3852995239	1642430184	234650315	
3044194711	2398693510	1904857554	3183125617	
486690751	3754138664	861877404	3190039692	
2686640734	522074127	3277825584	525584357	
2394526209	146352474	4267074718	1286834489	
2521660077	4104915256	3122860549	455810374	
49993987	3029415884	666423581	1869181575	

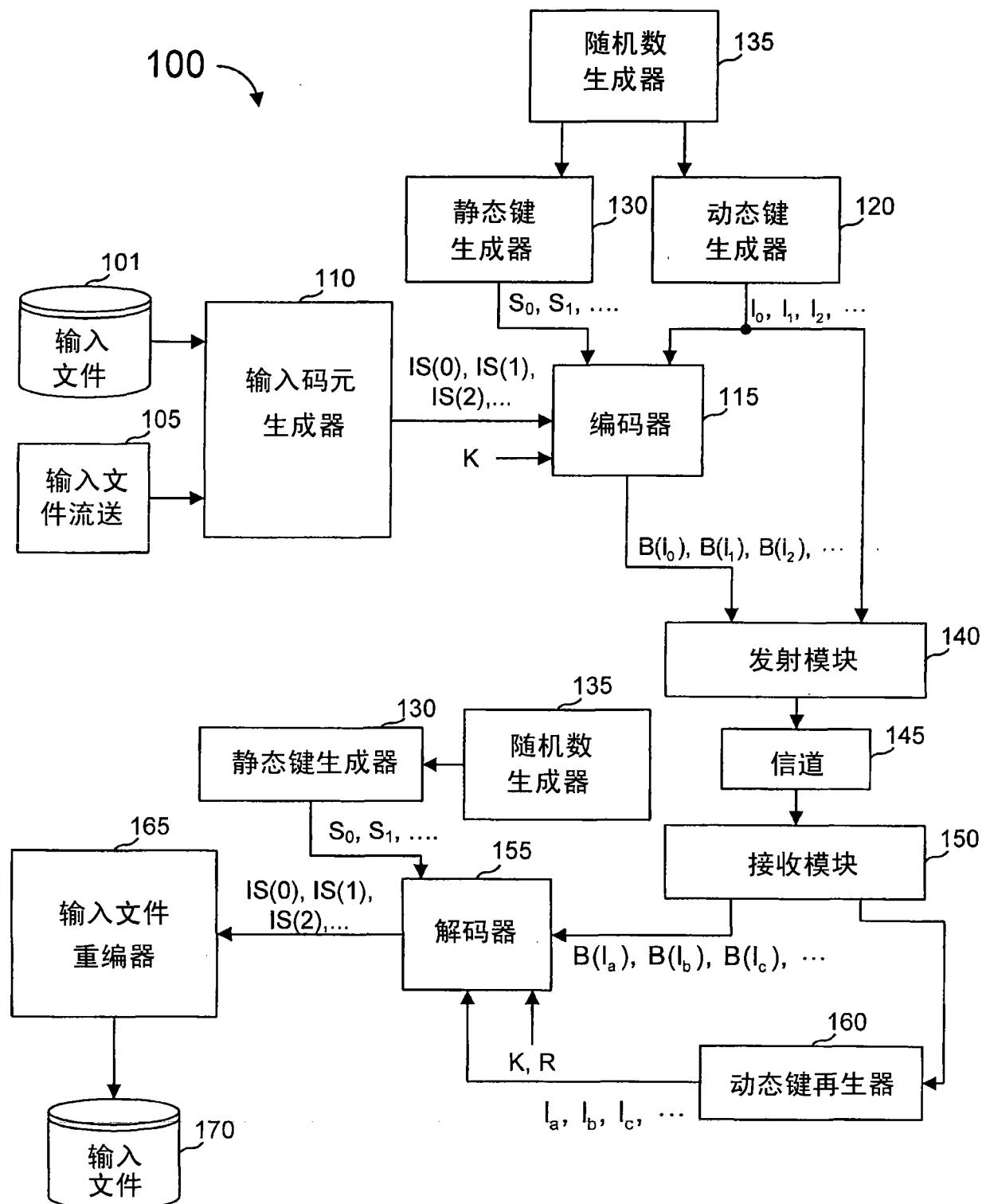


图 1

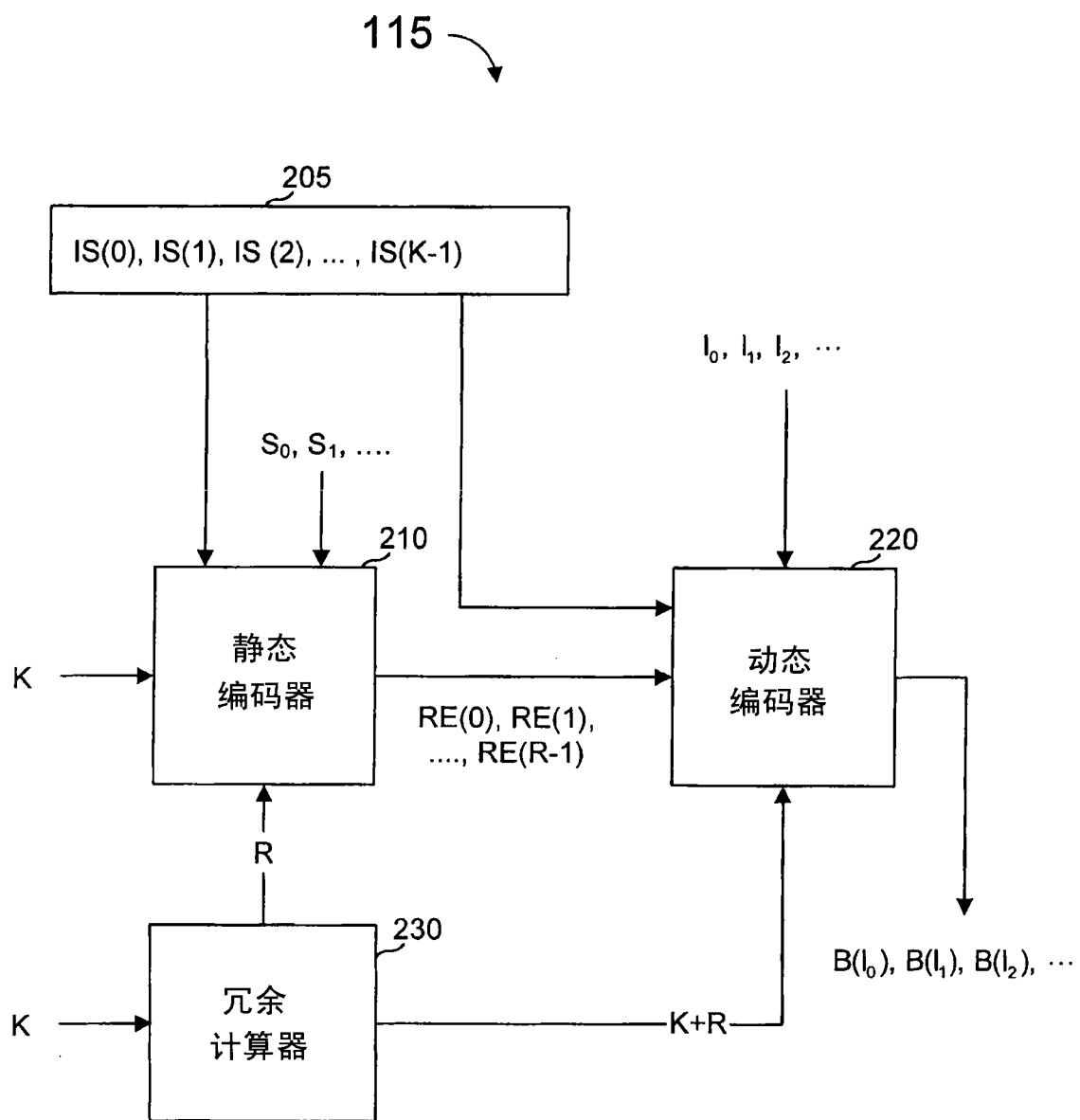


图 2

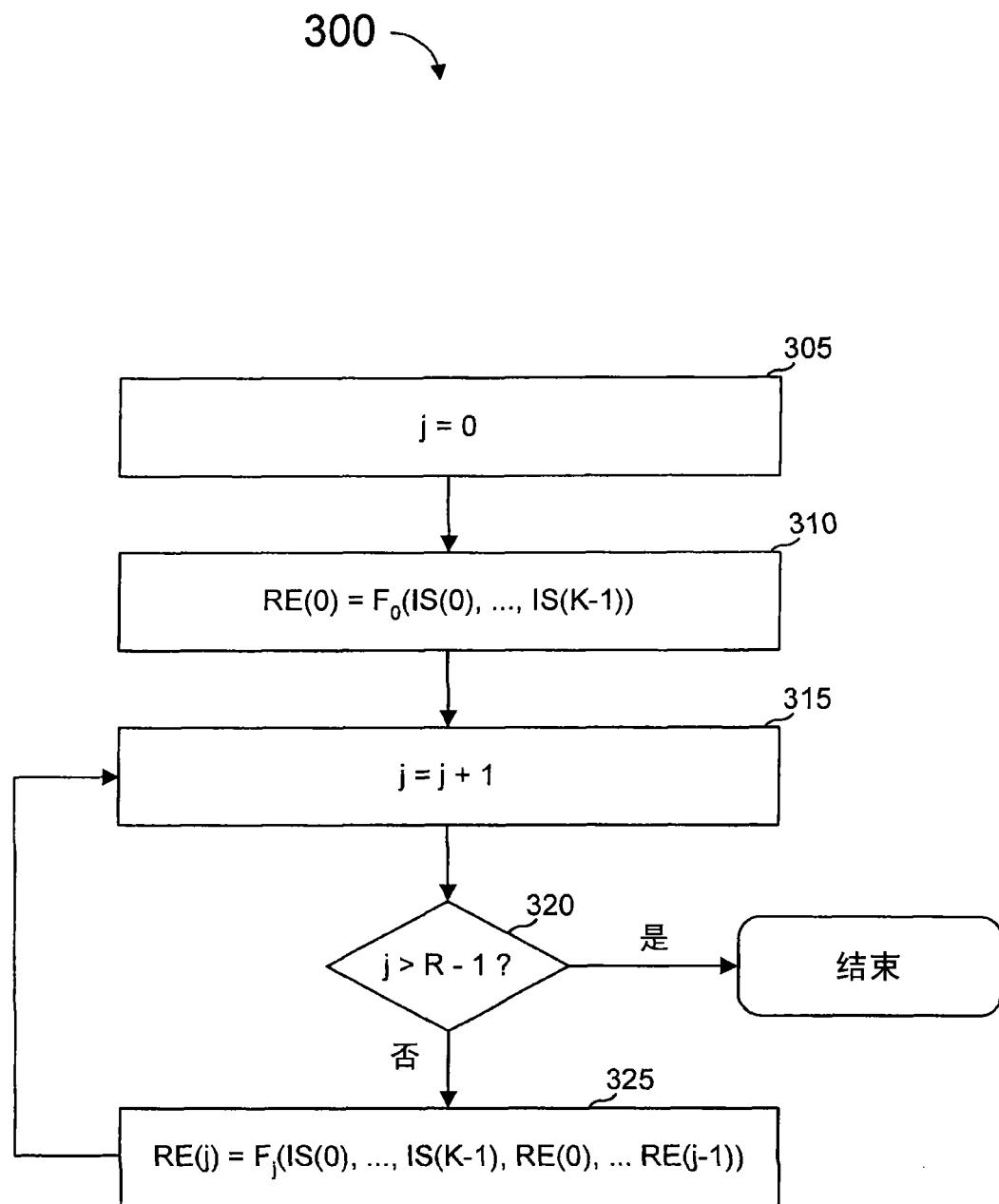


图 3

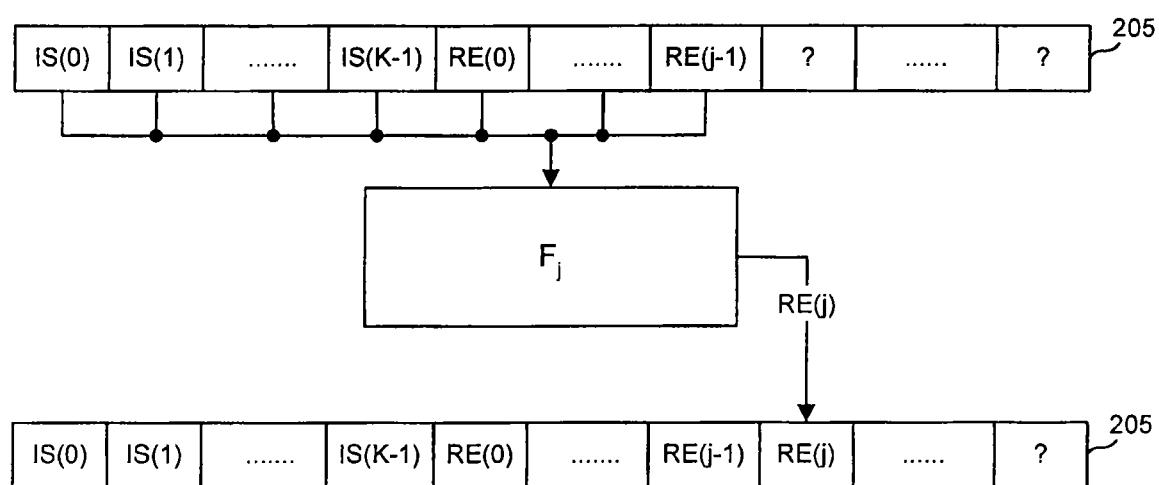


图 4

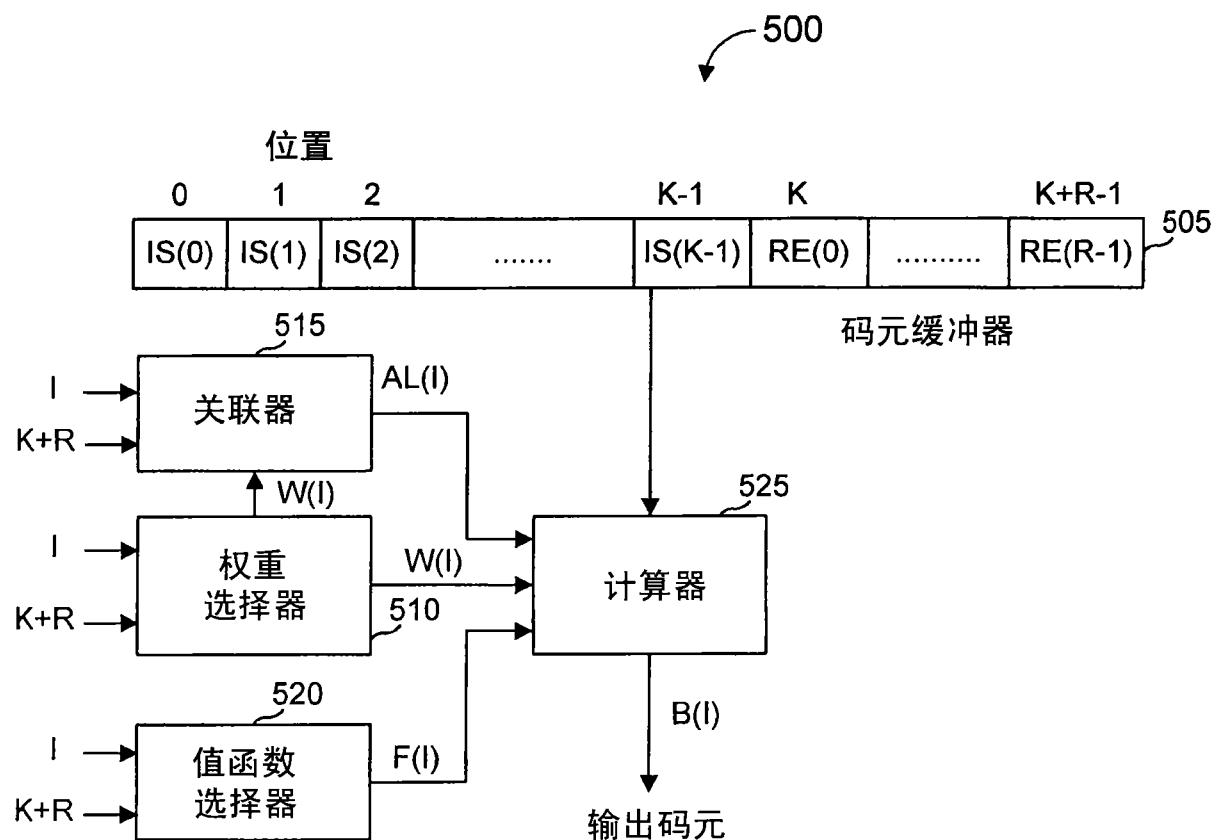


图 5

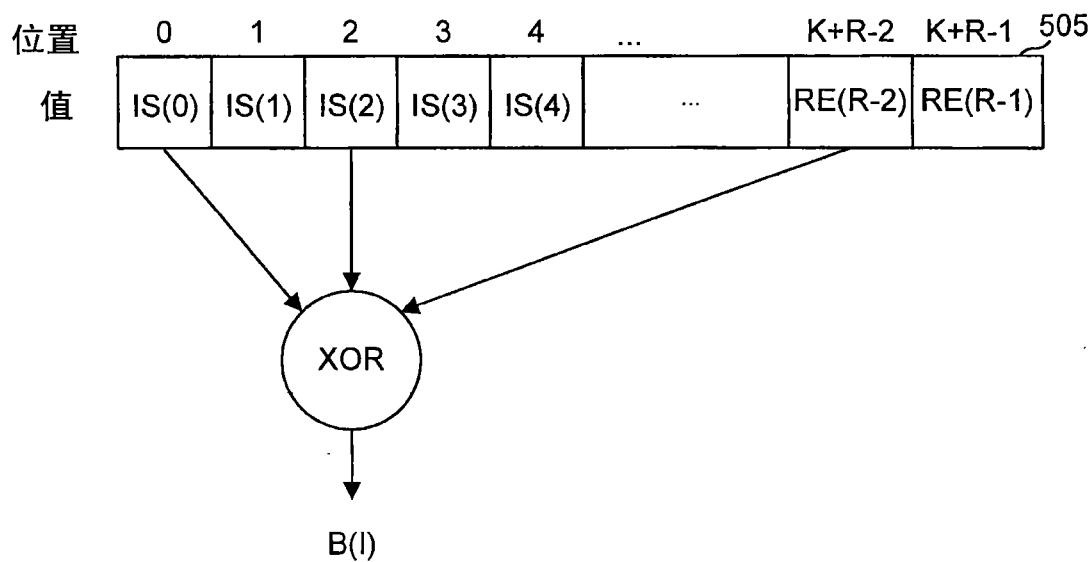


图 6

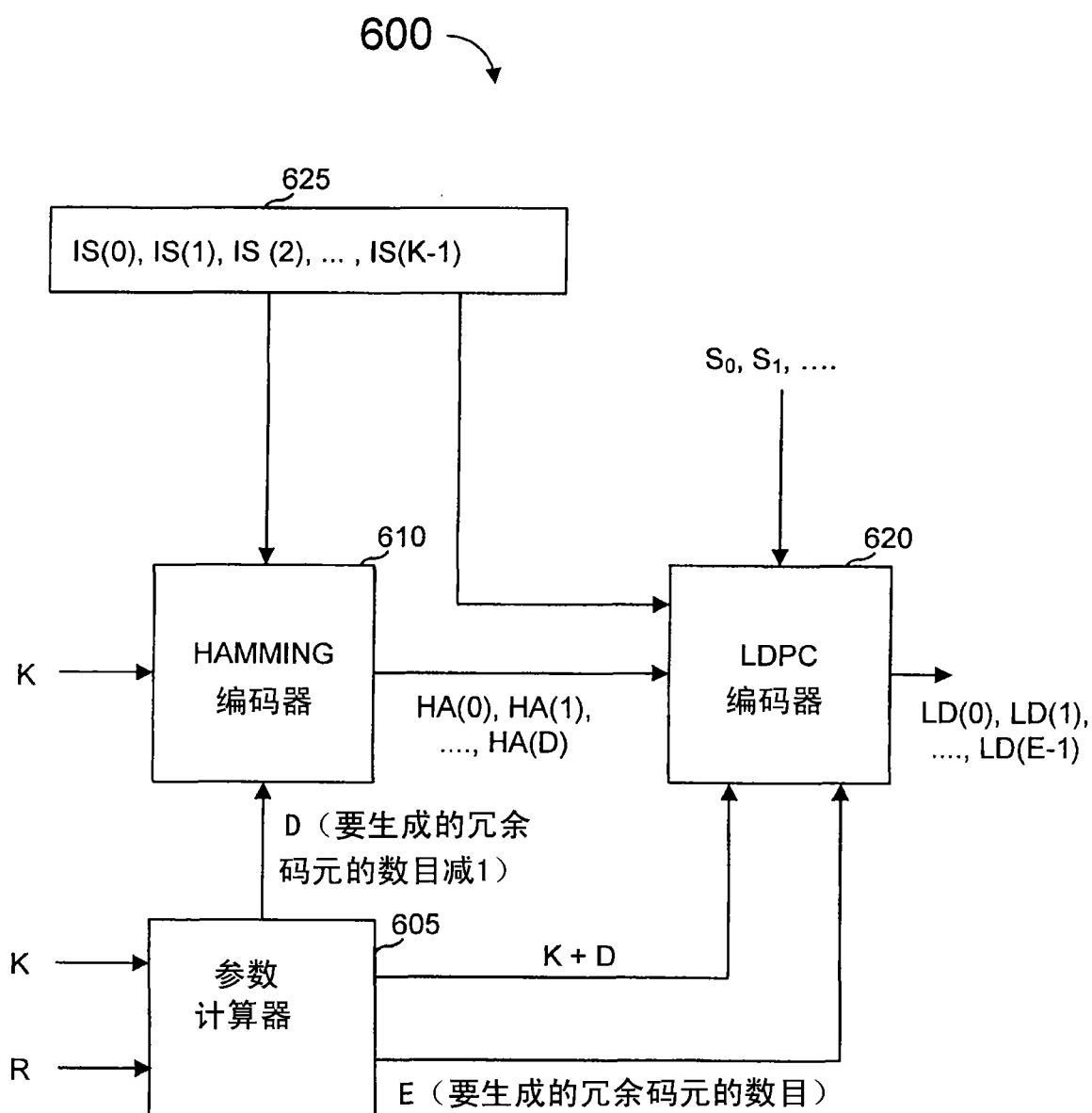


图 7

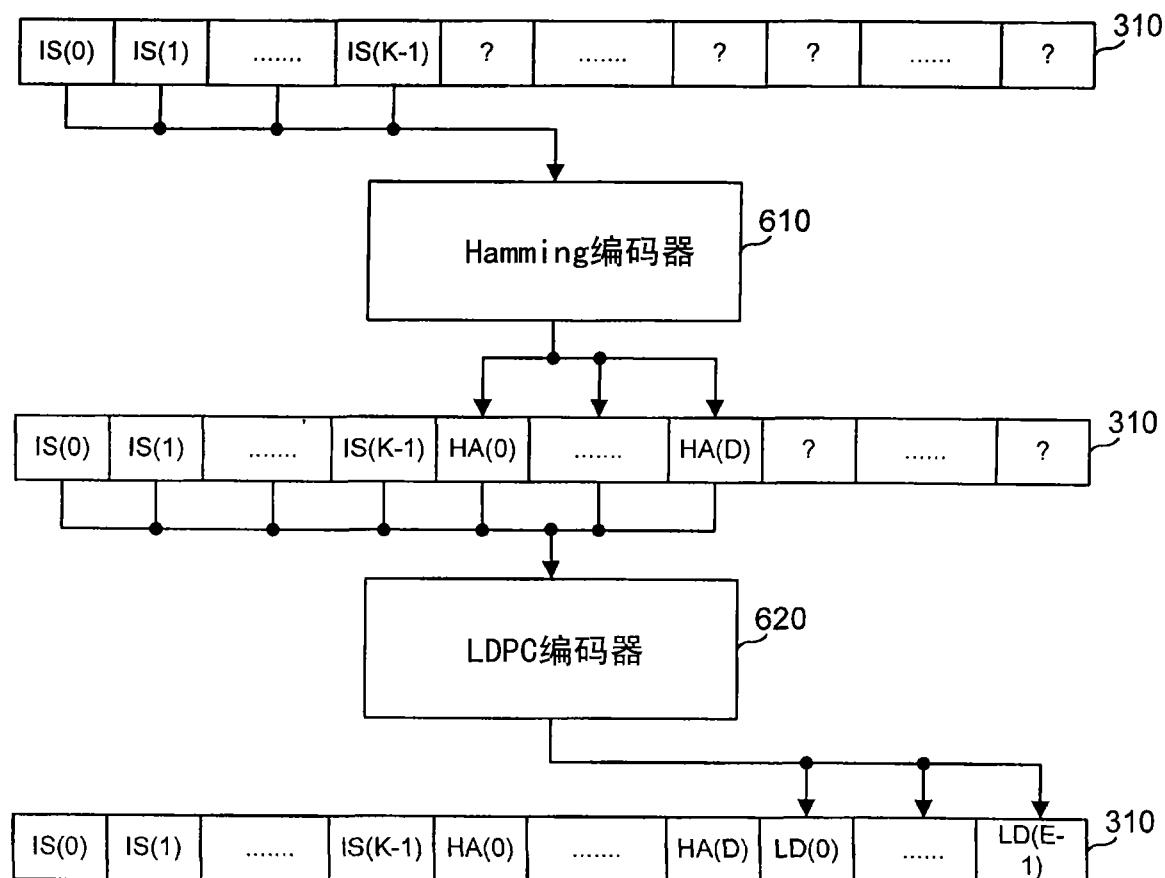


图 8

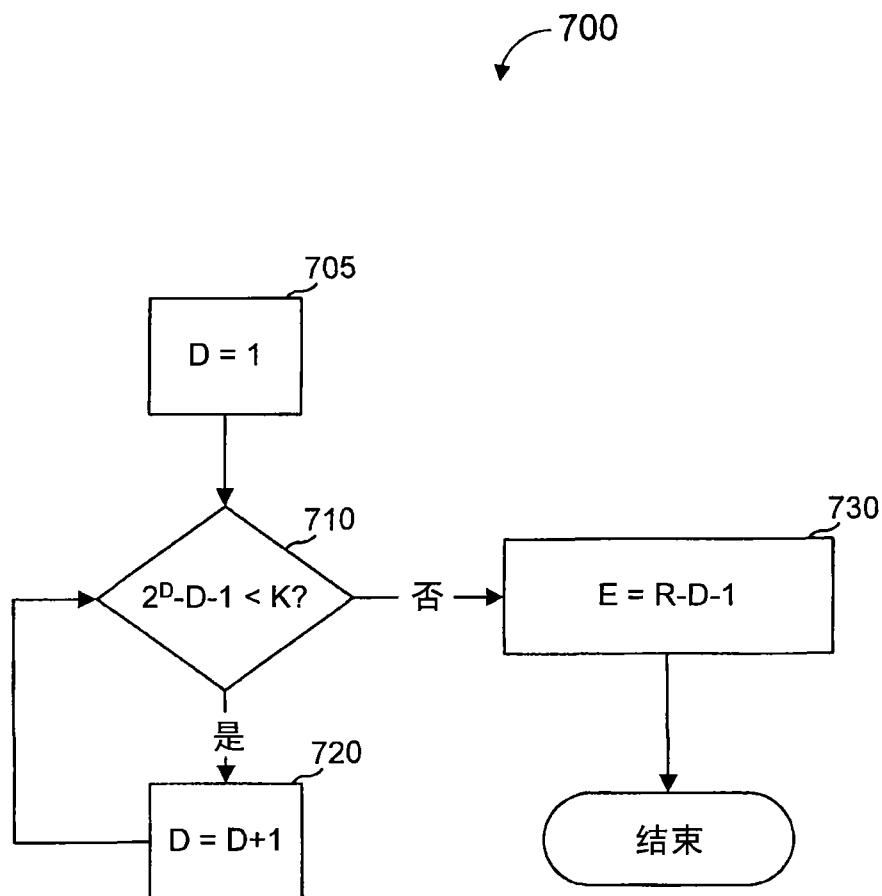


图 9

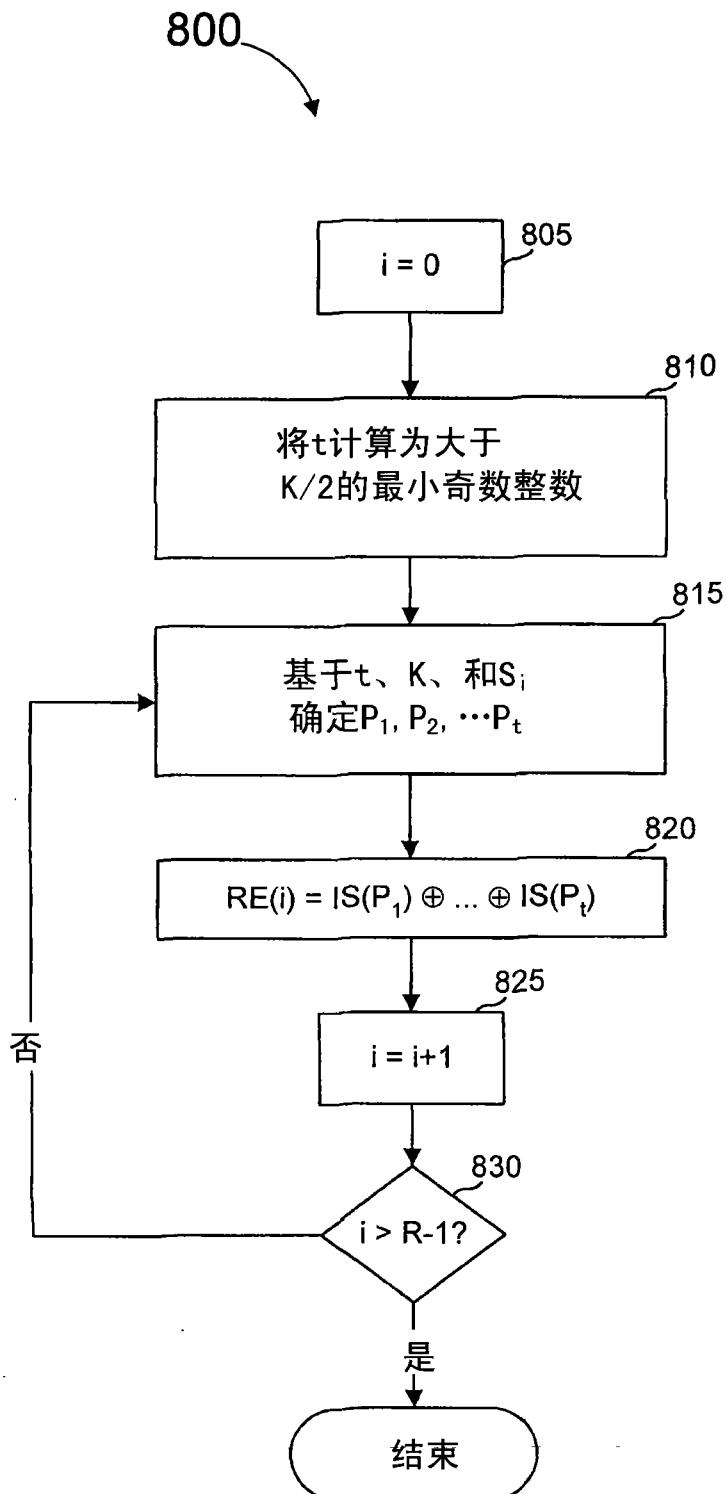


图 10

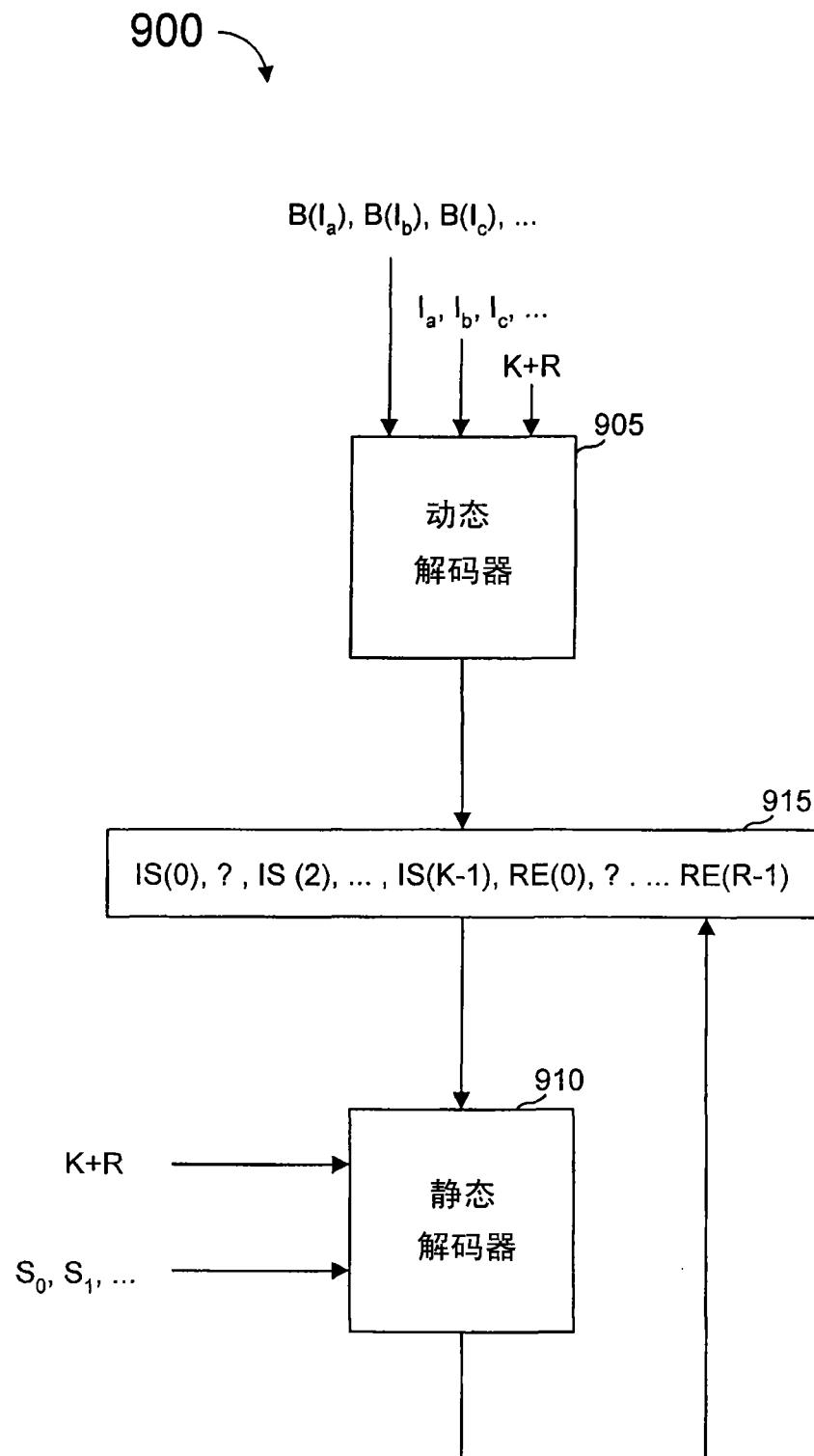


图 11

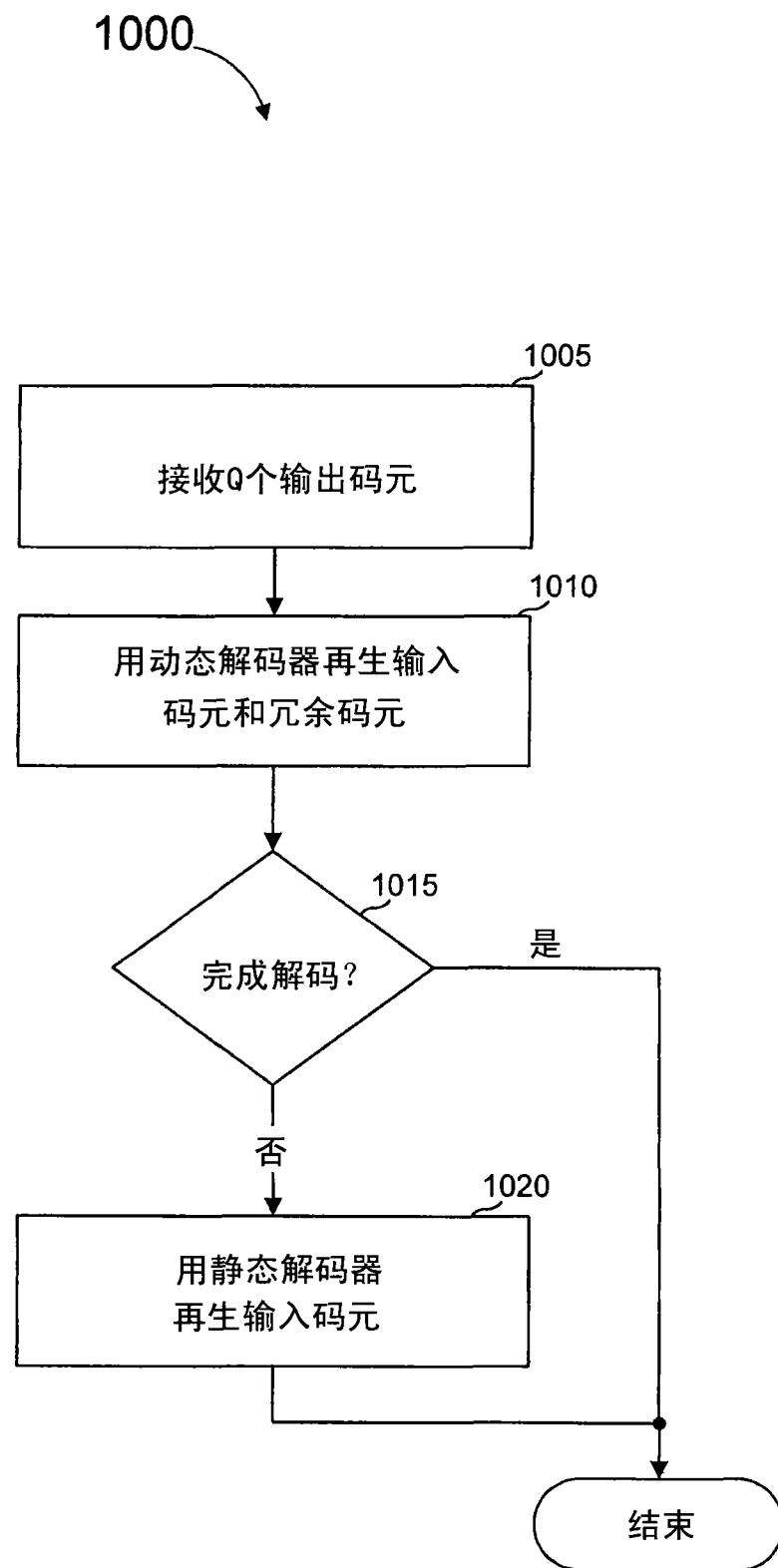


图 12

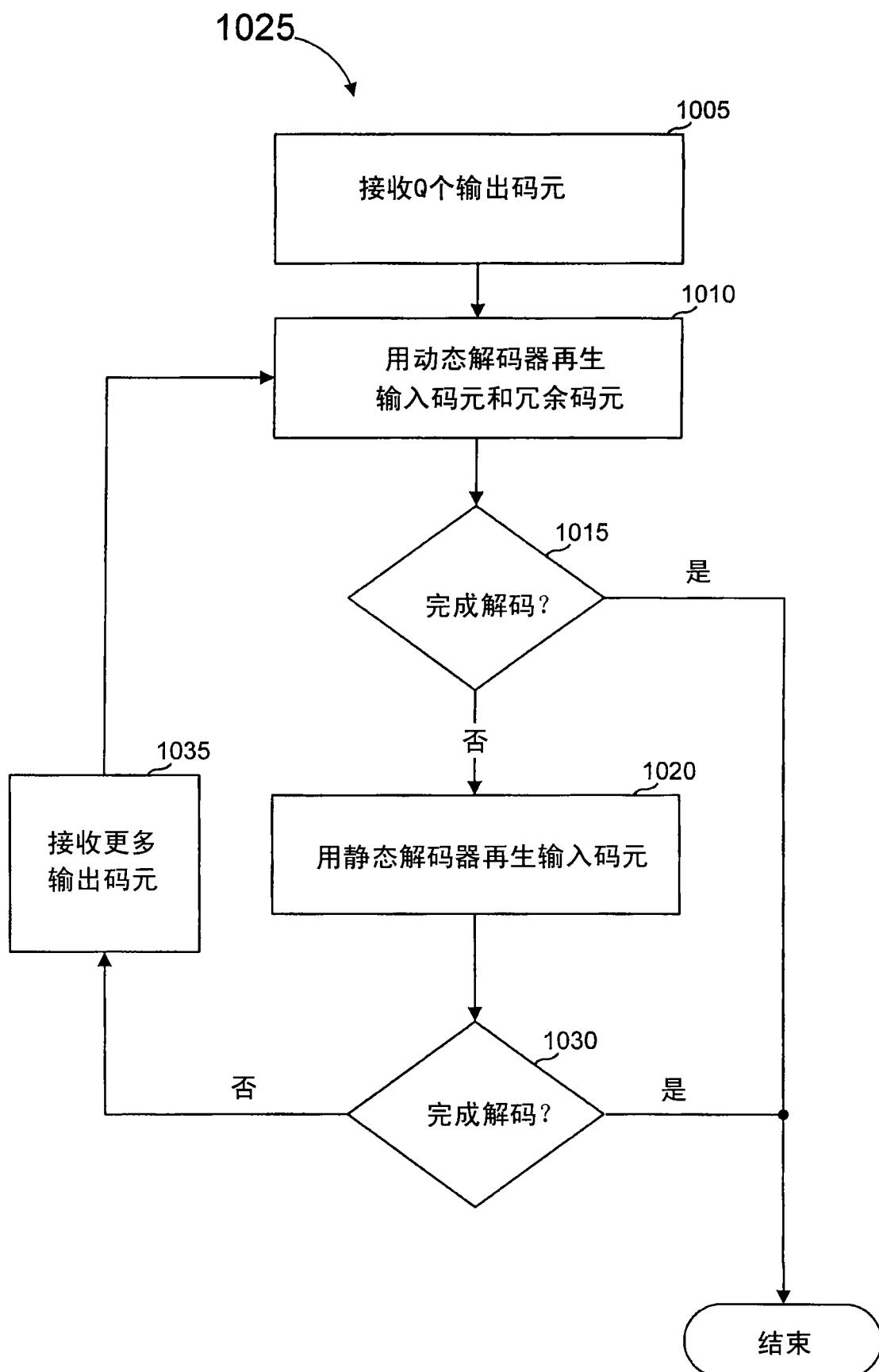


图 13

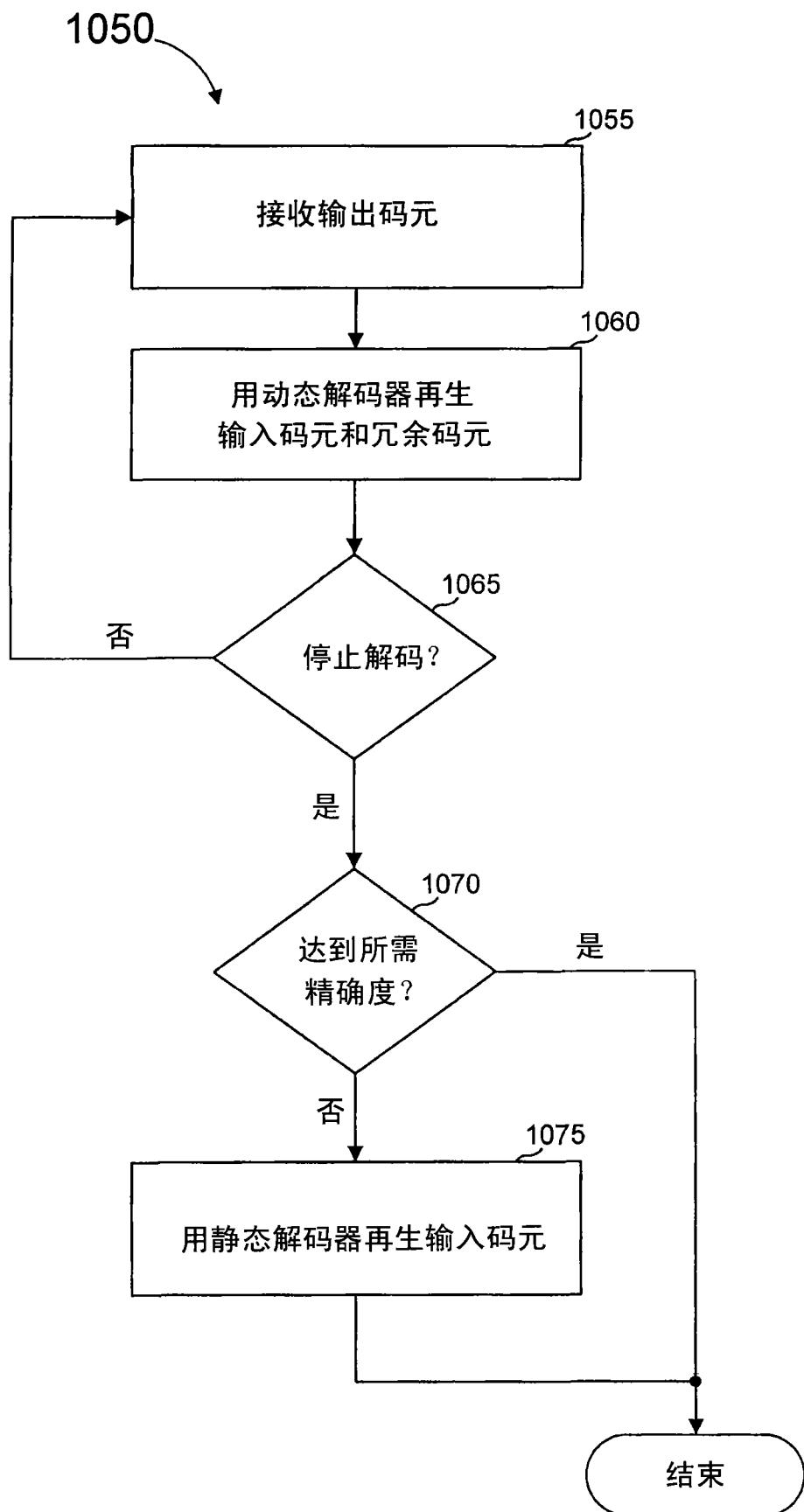


图 14

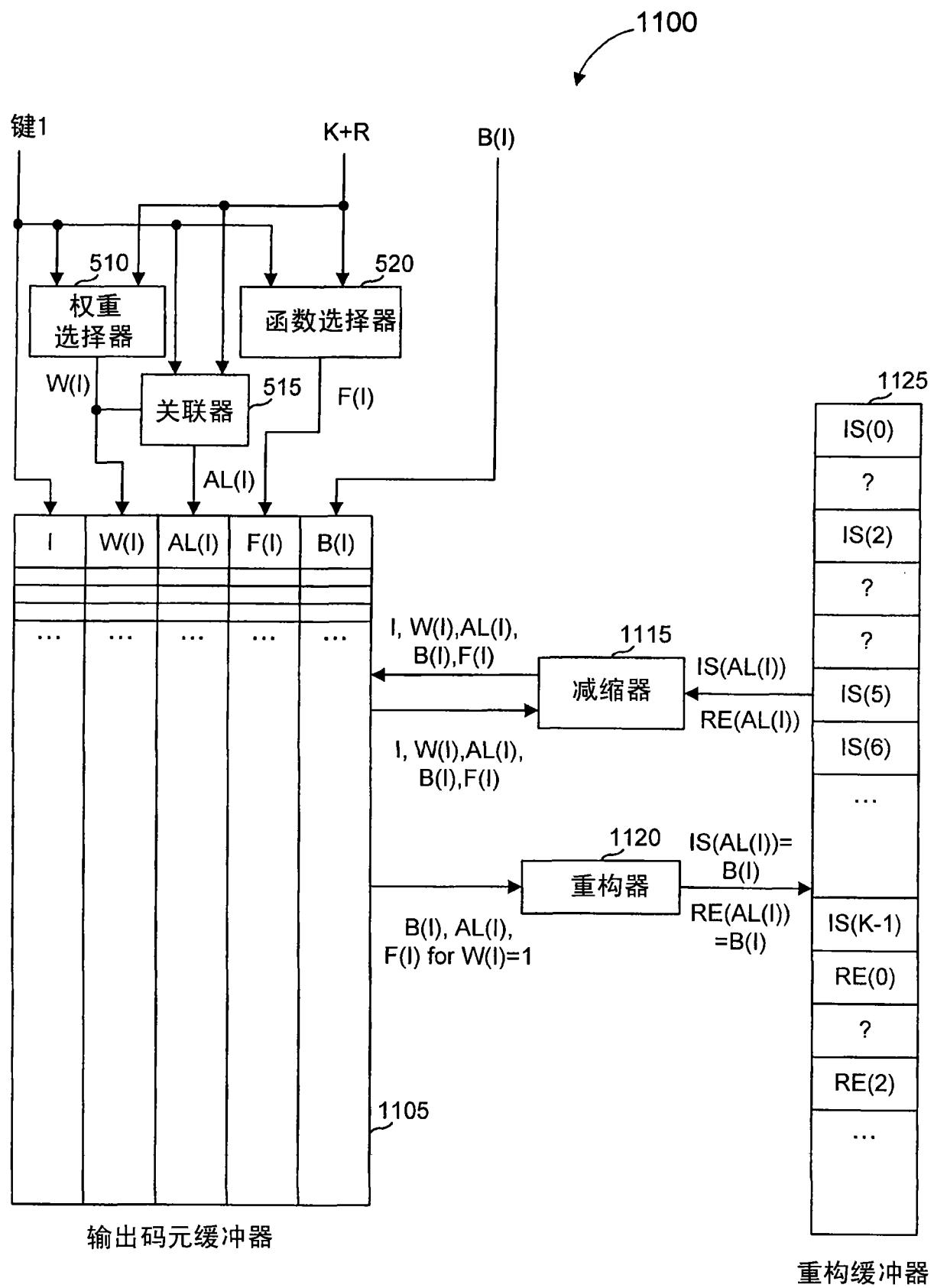


图 15

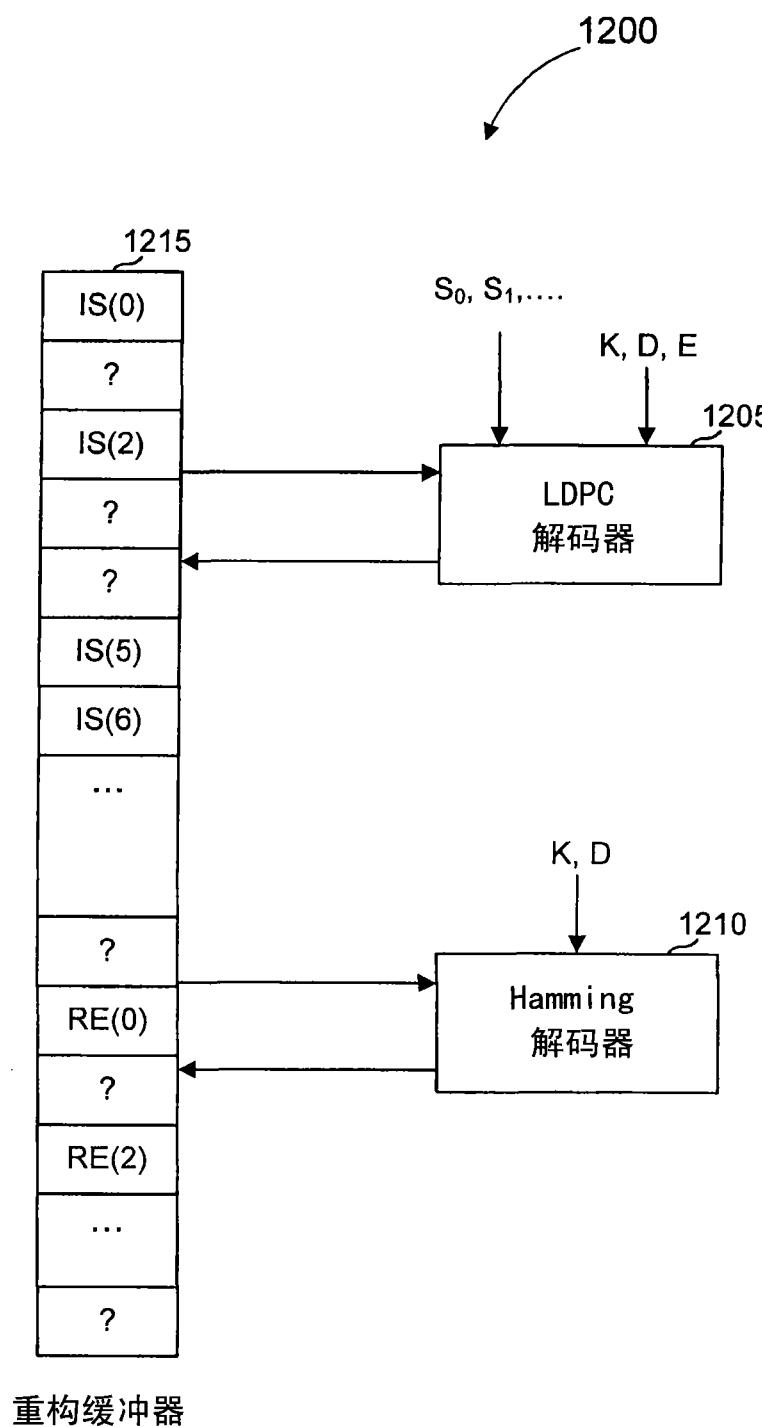


图 16

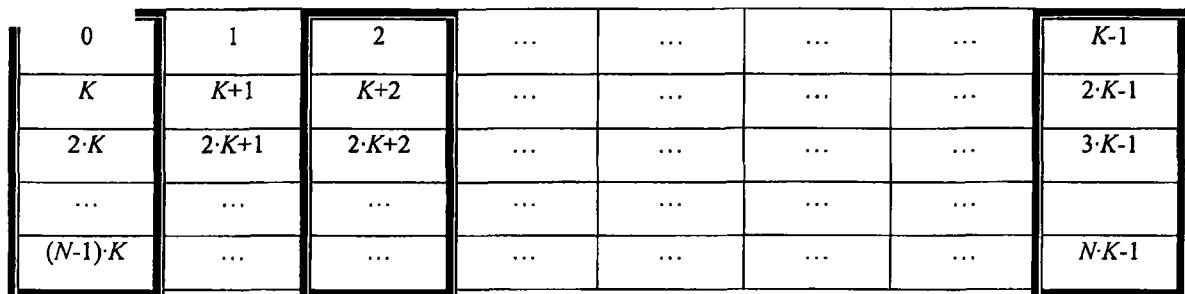


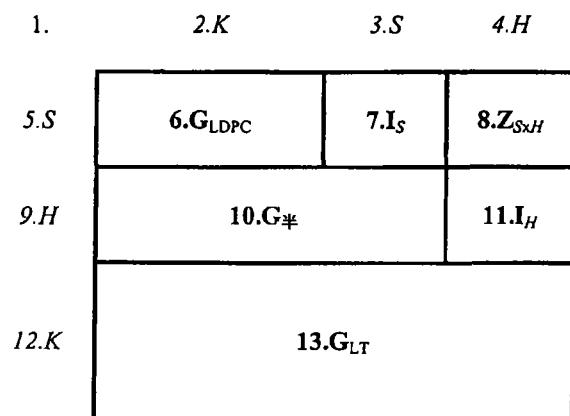
图 17

文件大小 <i>F</i>	<i>G</i>	码元大小 <i>T</i>	<i>G*T</i>	<i>K<sub>t</sub></i>	源块 <i>Z</i>	子块 <i>N</i>	<i>K<sub>L</sub></i>	<i>K<sub>S</sub></i>	<i>T<sub>L</sub>*A</i>	<i>T<sub>S</sub>*A</i>
100 KB	1	512	512	200	1	1	200	200	N/A	N/A
300 KB	1	512	512	600	1	2	600	600	128	128
1,000 KB	1	512	512	2,000	1	5	2,000	2,000	104	100
3,000 KB	1	512	512	6,000	1	12	6,000	6,000	44	40
10,000 KB	1	512	512	20,000	3	14	6,666	6,667	40	36

图 18

最大源块 大小B	<i>P<sub>30</sub></i>	<i>G</i>	码元大小
16KB	1424	1	1424
32KB	1424	1	1424
128KB	700	2	712

图 19



**图 20**

索引 $j$	$f[j]$	$d[j]$
0	0	--
1	10241	1
2	491582	2
3	712794	3
4	831695	4
5	948446	10
6	1032189	11
7	1048576	40

**图 21**

单位矩阵 I	全零	
全零	V	U

图 22

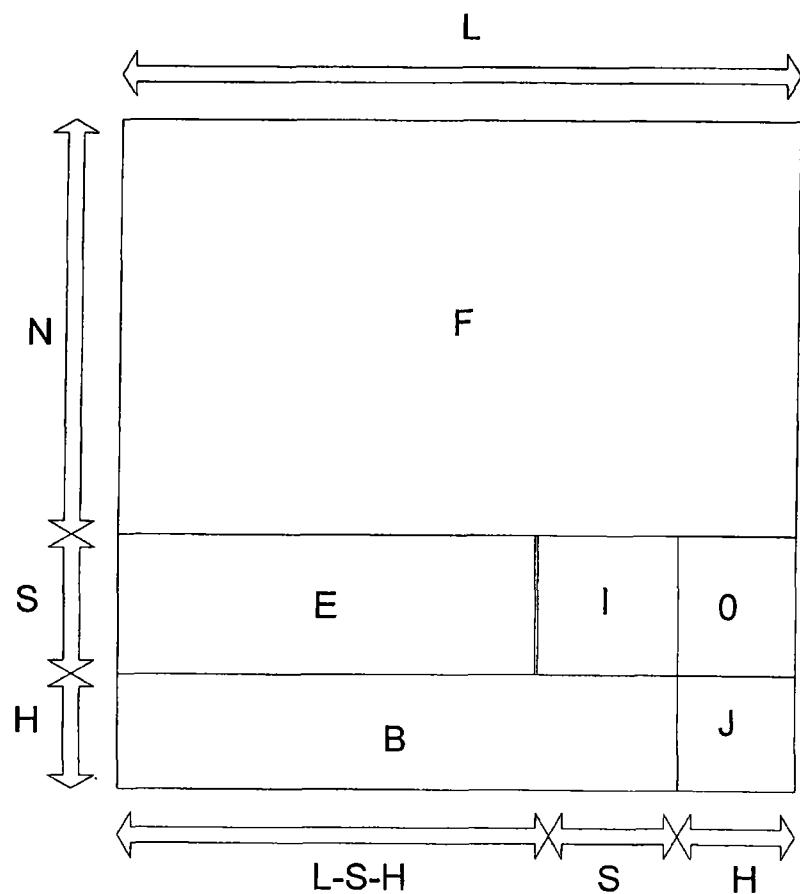


图 23

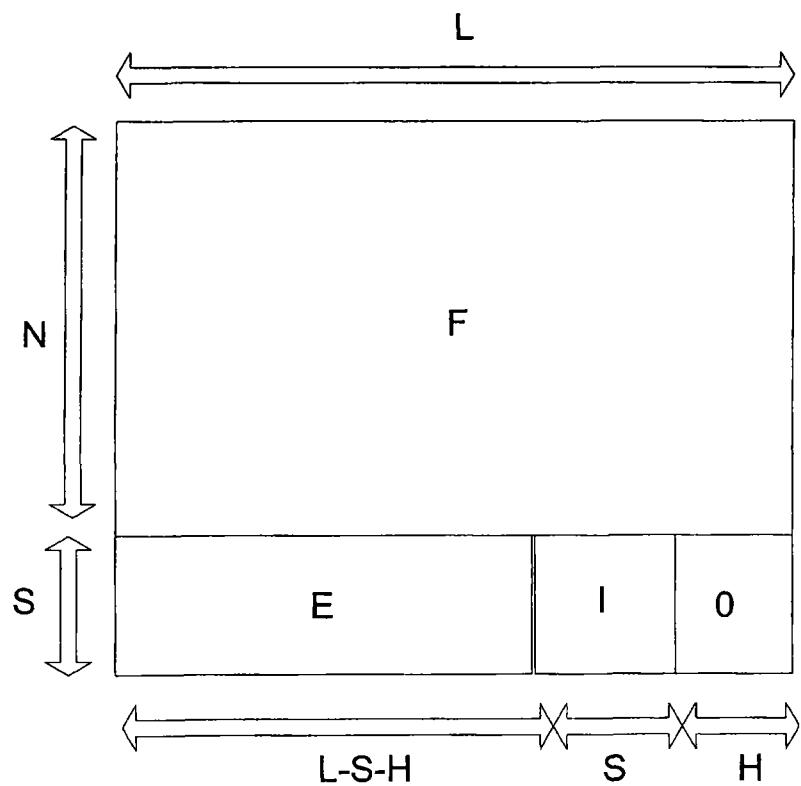


图 24a

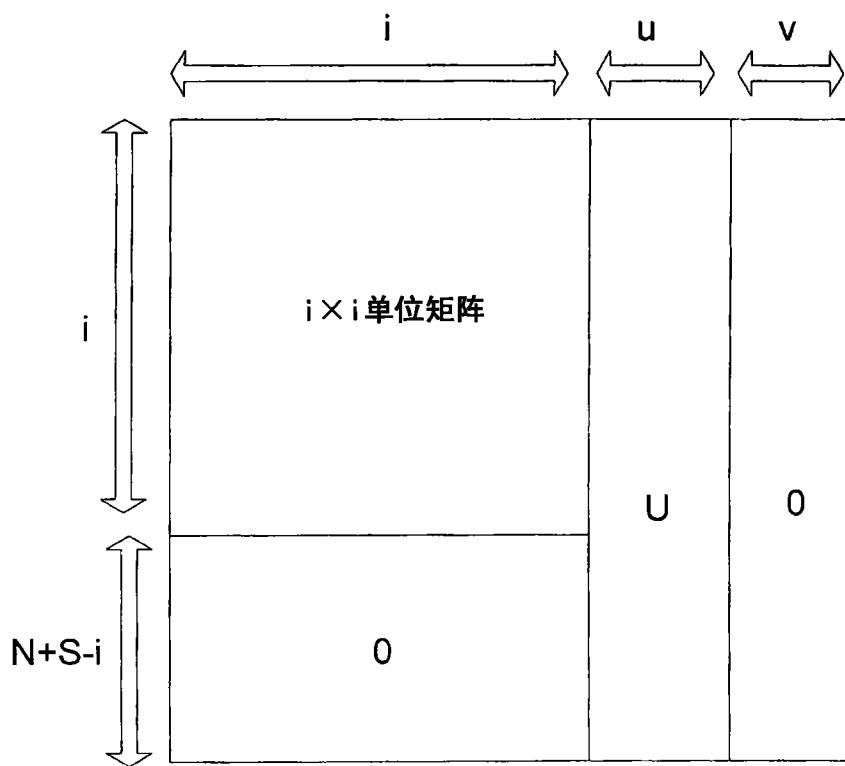


图 24b

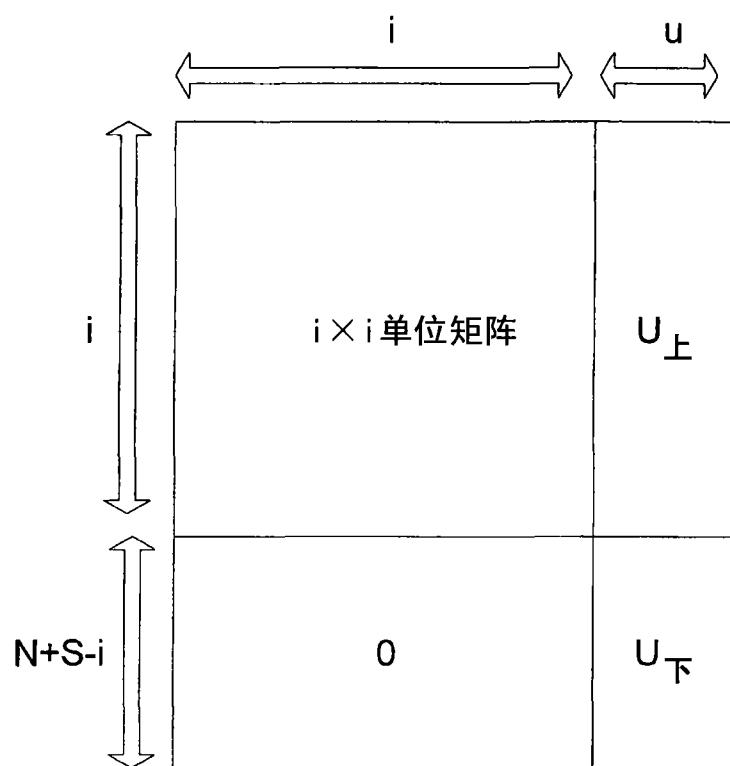


图 25

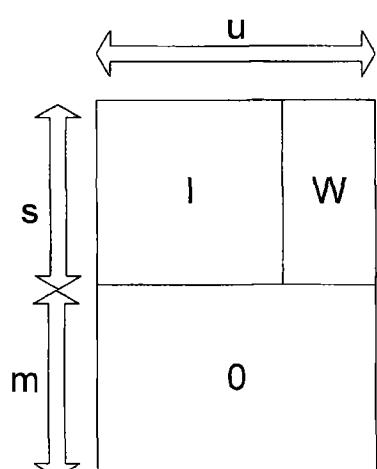


图 26

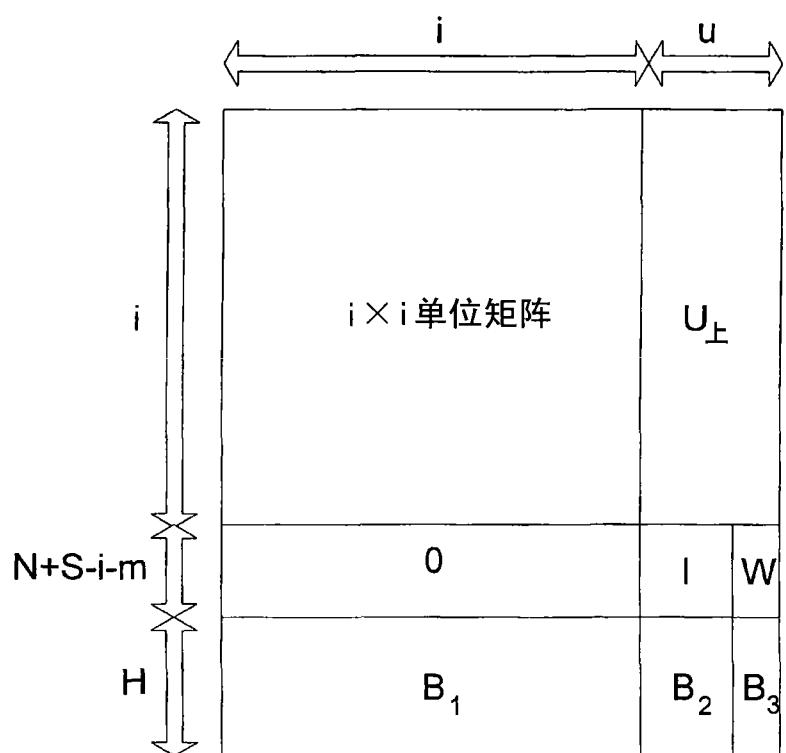


图 27

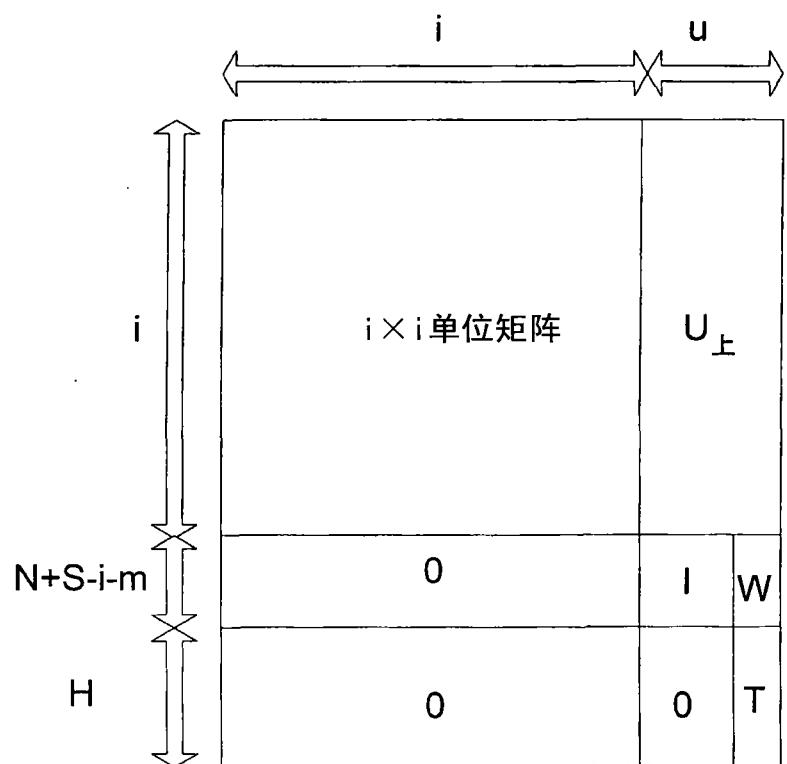


图 28

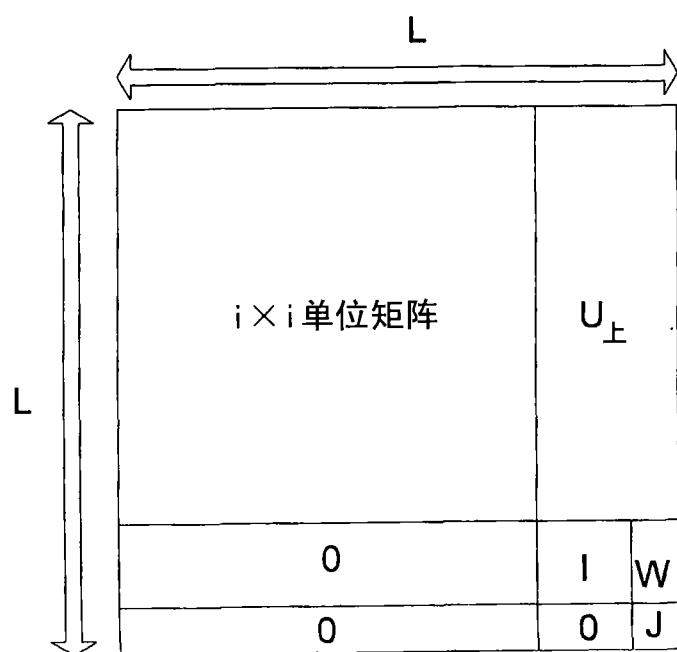


图 29

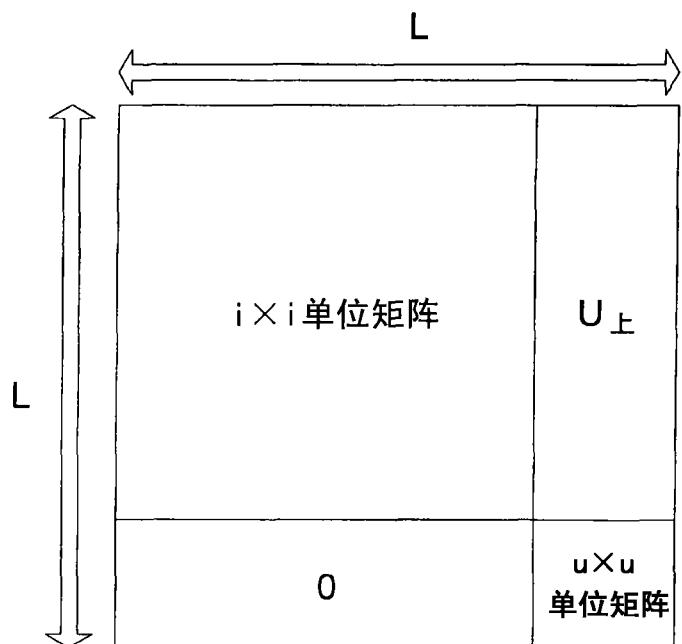


图 30

(120, 100) 码的失败概率	
接收开销 (A)	失败概率
0	$1.7 \times 10^{-2}$
1	$1.25 \times 10^{-4}$
2	$2.56 \times 10^{-6}$
3	$< 1 \times 10^{-8}$
4	$< 1 \times 10^{-8}$
5	$< 1 \times 10^{-8}$
6	$< 1 \times 10^{-8}$
7	$< 1 \times 10^{-8}$
8	$< 1 \times 10^{-8}$
9	$< 1 \times 10^{-8}$
10	$< 1 \times 10^{-8}$
11	$< 1 \times 10^{-8}$
12	$< 1 \times 10^{-8}$
13	$< 1 \times 10^{-8}$
14	$< 1 \times 10^{-8}$
15	0
16	0
17	0
18	0
19	0
20	0

(110, 100) 码的失败概率	
接收开销 (A)	失败概率
0	$4.8 \times 10^{-3}$
1	$3.11 \times 10^{-5}$
2	$1.37 \times 10^{-7}$
3	$< 1 \times 10^{-8}$
4	$< 1 \times 10^{-8}$
5	$< 1 \times 10^{-8}$
6	0
7	0
8	0
9	0
10	0

图 32

图 31