(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 1620760 B
(45)授权公告日 2014.02.26

(21)申请号 02828117.9
(22)申请日 2002.12.23
(30)优先权数据
10/032,156 2001.12.21 US
(85)PCT国际申请进入国家阶段日
2004.08.16
(86)PCT国际申请的申请数据
PCT/US2002/041615 2002.12.23
(87)PCT国际申请的公布数据
WO2003/056703 EN 2003.07.10
(73)专利权人 数字方敦股份有限公司
地址 美国加利福尼亚州
(72)发明人 A·M·肖克罗拉希 S·拉森 M·路比
(74)专利代理机构 上海专利商标事务所有限公司 31100
代理人 陈斌
(51)Int.Cl.
H03M 7/00(2006.01)
H03M 13/00(2006.01)
H04L 1/00(2006.01)
H04L 25/49(2006.01)
H04L 27/04(2006.01)
(56)对比文件
US 6307487 B1,2001.10.23, 全文.
US 6320520 B1,2001.11.20, 全文.
审查员 尹剑峰

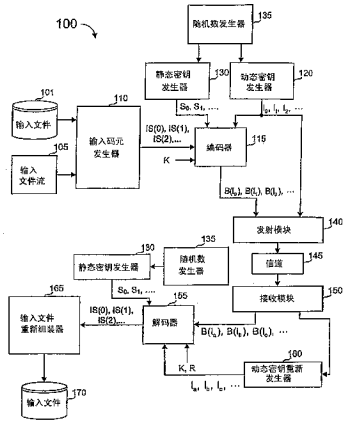
权利要求书6页 说明书28页 附图20页

(54)发明名称

用于通信系统的多级码发生器和解码器

(57)摘要

提供了一种用于在通信信道(145)上从源(101)到目的地(170)的数据传输编码的方法。从要发送的输入码元的排序集合生成多个冗余码元。从包括输入码元和冗余码元的码元组合集合生成多个输出码元,其中可能的输出码元数远远大于码元的组合集合内的码元数,其中从码元的组合集合内多于一个码元且从码元的组合集合内少于所有码元生成至少一个输出码元,使得输入码元的排序集合可以从输出码元的任何预定数N经重新生成达到期望的准确度。



CN 1620760 B

1. 一种用于在通信信道上从源到目的地的数据传输的编码的方法，其特征在于包括：

安排要发送的数据到输入码元的排序集合内；

从所述输入码元生成多个冗余码元；以及

从包括输入码元和冗余码元的码元组合集合生成多个输出码元，其中对于一个给定输入码元集合，可能的输出码元数远远大于码元组合集合中的码元数，其中从码元的组合集合内多于一个码元且从码元的组合集合内少于所有码元中生成至少一个输出码元，使得输入码元的排序集合可以从任何预定数目 N 的输出码元经重新生成达到期望的准确度。

2. 如权利要求 1 所述的方法，其特征在于还包括在所述通信信道上发送多个输出码元。

3. 如权利要求 1 所述的方法，其特征在于进一步包括在存储媒质上存储多个输出码元。

4. 如权利要求 1 所述的方法，其特征在于 N 大于输入码元的排序集合内的输入码元数。

5. 如权利要求 1 所述的方法，其特征在于 N 小于或等于输入码元的排序集合内的输入码元数。

6. 如权利要求 1 所述的方法，其特征在于还包括基于输入码元的排序集合内的输入码元的数目 K 确定要生成的冗余码元的数目 R。

7. 如权利要求 6 所述的方法，其特征在于 K 是输入码元数的估计。

8. 如权利要求 1 所述的方法，其特征在于多个冗余码元根据 LDPC 码生成。

9. 如权利要求 1 所述的方法，其特征在于期望的准确度是输入码元的完整恢复。

10. 如权利要求 1 所述的方法，其特征在于期望的准确度是输入码元以高概率的完整恢复。

11. 如权利要求 1 所述的方法，其特征在于期望的准确度是 G 个输入码元的恢复，其中 G 小于输入码元的排序集合内的输入码元数。

12. 如权利要求 1 所述的方法，其特征在于生成多个输出码元的步骤是使用第一设备实现的，且其中生成多个冗余码元的步骤是使用与第一设备分离的第二设备实现。

13. 一种用于在通信信道上从源到目的地的数据传输的编码的方法，其特征在于包括：

安排要发送的数据到输入码元的排序集合内；

从所述输入码元生成多个冗余码元；以及

从包括输入码元和冗余码元的码元组合集合生成多个输出码元，其中对于一个给定输入码元集合，可能的输出码元数远远大于码元组合集合中的码元数，其中从码元的组合集合内多于一个码元且从码元的组合集合内少于所有码元中生成至少一个输出码元，使得输入码元的排序集合可以从任何 N 个输出码元重新生成达到期望的准确度，

其中所述多个冗余码元包括多个第一冗余码元和多个第二冗余码元，生成所述多个冗余码元的步骤包括：

从所述输入码元生成所述多个第一冗余码元；以及

从所述第一冗余码元和所述输入码元生成所述多个第二冗余码元。

14. 如权利要求 13 所述的方法，其特征在于多个第一冗余码元根据汉明码生成，且其中多个第二冗余码元根据 LDPC 码生成。

15. 如权利要求 14 所述的方法，其特征在于还包括：

基于输入码元的排序集合内的输入码元数 K 确定第一冗余码元的数目 D+1；以及

基于要生成的冗余码元的数目 R 和 D+1 确定第二冗余码元数 E。

16. 如权利要求 15 所述的方法，其特征在于还包括基于 K 确定 R。

17. 如权利要求 15 所述的方法，其特征在于 K 是输入码元数的估计。

18. 如权利要求 15 所述的方法，其特征在于 D 是使得 $2^D-D-1>=K$ 的最小整数，其中 E=R-D-1。

19. 一种用于在通信信道上从源到目的地的数据传输的编码的方法，其特征在于包括：

安排要发送的数据到输入码元的排序集合内；

从所述输入码元生成多个冗余码元；以及

从包括输入码元和冗余码元的码元组合集合生成多个输出码元，其中对于一个给定输入码元集合，可能的输出码元数远远大于码元组合集合中的码元数，其中从码元的组合集合内多于一个码元且从码元的组合集合内少于所有码元中生成至少一个输出码元，使得输入码元的排序集合可以从任何 N 个输出码元重新生成达到期望的准确度，

其中可以从任何数量的输出码元中重新生成最多 G 个输入码元，其中 G 小于输入码元的排序集合内的输入码元数。

20. 一种用于在通信信道上从源到目的地的数据传输的编码的方法，其特征在于包括：

安排要发送的数据到输入码元的排序集合内；

从所述输入码元生成多个冗余码元对于每个冗余码元包括：

a) 根据一分布确定 t 个不同的输入码元；以及

b) 按照 t 个不同输入码元的 XOR 计算每个冗余码元；以及

从包括输入码元和冗余码元的码元组合集合生成多个输出码元，其中对于一个给定输入码元集合，可能的输出码元数远远大于码元组合集合中的码元数，其中从码元的组合集合内多于一个码元且从码元的组合集合内少于所有码元中生成至少一个输出码元，使得输入码元的排序集合可以从任何 N 个输出码元重新生成达到期望的准确度。

21. 如权利要求 20 所述的方法，其特征在于 t 对于所有冗余码元相同。

22. 如权利要求 21 所述的方法，其特征在于 t 是大于 K/2 的最小奇整数，其中 K 是输入码元的排序集合内的输入码元数。

23. 如权利要求 20 所述的方法，其特征在于所述分布是均匀分布。

24. 一种用于在通信信道上从源到目的地的数据传输的编码的方法，其特征在于包括：

安排要发送的数据到输入码元的排序集合内；

从所述输入码元生成多个冗余码元对于每个冗余码元包括：

从包括输入码元和冗余码元的码元组合集合生成多个输出码元，其中，对于一个给定输入码元集合，可能的输出码元数远远大于码元组合集合中的码元数，其中从码元的组合集合内多于一个码元且从码元的组合集合内少于所有码元中生成至少一个输出码元，使得输入码元的排序集合可以从任何 N 个输出码元重新生成达到期望的准确度；以及

在所述通信信道上发送多个输出码元，其中生成多个输出码元的步骤是与发送多个输出码元的步骤大致上并发进行的。

25. 如权利要求 24 所述的方法，其特征在于生成多个冗余码元的步骤是与发送多个输出码元步骤大致并发进行的。

26. 如权利要求 24 所述的方法，其特征在于生成多个冗余码元的步骤是在发送多个输出码元步骤之前进行的。

27. 一种用于在通信信道上对从源到目的地传输的数据编码的系统，其特征在于包括：

被耦合以接收多个输入码元的静态编码器，多个输入码元从要发送的数据中生成，静态编码器包括冗余码元发生器，所述发生器基于输入码元生成多个冗余码元；以及

被耦合以接收多个输入码元和多个冗余码元的动态编码器，动态编码器包括输出码元发生器，所述输出码元发生器从包括多个输入码元和多个冗余码元的码元组合集合中生成多个输出码元，其中可能的输出码元数远远大于码元组合集合中的码元数，其中从组合集合中多于一个码元且小于组合集合中所有码元中生成至少一个输出码元，使得输入码元的排序集合可以从输出码元的任何预定数 N 中重新生成达到期望的准确度。

28. 如权利要求 27 所述的系统，其特征在于 N 大于输入码元排序集合内的输入码元数。

29. 如权利要求 27 所述的系统，其特征在于 N 小于或等于输入码元的排序集合内的输入码元数。

30. 如权利要求 27 所述的系统，其特征在于还包括发射模块，所述发射模块耦合到动态编码器和通信信道，并在通信信道上接收输出码元并发送输出码元。

31. 如权利要求 27 所述的系统，其特征在于还包括密钥发生器，所述密钥发生器耦合到静态编码器，所述静态编码器为至少一些要生成的冗余码元的每个生成密钥，其中静态编码器经耦合以接收每个密钥，且其中静态编码器基于对应的密钥生成至少一些冗余码元的每个。

32. 如权利要求 27 所述的系统，其特征在于静态编码器包括 LDPC 编码器。

33. 一种用于在通信信道上对从源到目的地传输的数据编码的系统，其特征在于包括：

被耦合以接收多个输入码元的静态编码器，多个输入码元从要发送的数据中生成，静态编码器包括冗余码元发生器，所述冗余码元发生器基于输入码元生成多个冗余码元；

被耦合以接收多个输入码元和多个冗余码元的动态编码器，动态编码器包括输出码元发生器，所述输出码元发生器从包括多个输入码元和多个冗余码元的码元组合集合中生成多个输出码元，其中可能的输出码元数远远大于码元组合集合中的码元数，其中从组合集合中多于一个码元且小于组合集合中所有码元中生成至少一个输出码元，使得输入码元的排序集合可以从任何 N 个输出码元中重新生成达到期望的准确度；以及

密钥发生器，所述密钥发生器耦合到动态编码器，所述密钥发生器为每个要生成的输出码元生成一密钥，其中动态编码器经耦合以接收每个密钥，且其中动态编码器基于对应的密钥生成每个输出码元。

34. 一种用于在通信信道上对从源到目的地传输的数据编码的系统，其特征在于包括：

被耦合以接收多个输入码元的静态编码器，多个输入码元从要发送的数据中生成，静态编码器包括冗余码元发生器，所述冗余码元发生器基于输入码元生成多个冗余码元；

被耦合以接收多个输入码元和多个冗余码元的动态编码器，动态编码器包括输出码元发生器，所述输出码元发生器从包括多个输入码元和多个冗余码元的码元组合集合中生成多个输出码元，其中可能的输出码元数远远大于码元组合集合中的码元数，其中从组合集合中多于一个码元且小于组合集合中所有码元中生成至少一个输出码元，使得输入码元的排序集合可以从任何 N 个输出码元中重新生成达到期望的准确度；以及

所述静态编码器还包括带有第一冗余码元发生器的第一静态编码器，以及带有第二冗余码元发生器的第二静态编码器；

所述多个冗余码元包括第一组冗余码元以及第二组冗余码元；

所述第一冗余码元发生器基于输入码元生成第一组冗余码元；以及

所述第二冗余码元发生器基于输入码元和第一组冗余码元生成第二组冗余码元。

35. 如权利要求 34 所述的系统，其特征在于第一静态编码器包括汉明编码器，其中第二静态编码器包括 LDPC 编码器。

36. 一种在通信信道上从源接收发送的数据的方法，其特征在于该方法包括：

接收输出码元，其中每个输出码元从输入码元和冗余码元的组合集合内的至少一个码元生成，其中至少一个输出码元从多于组合集合内的一个码元且少于组合集合内所有码元中生成，其中可能的输出码元数远远大于码元组合集合内的码元数，其中输入码元来自输入码元的排序集合，其中冗余码元从输入码元生成；

在接收到至少输出码元的一个子集后，从 N 个输出码元重新生成组合集合内的至少一个码元的子集，所述子集包括多个重新生成的输入码元以及多个重新生成的冗余码元；

如果从 N 个输出码元重新生成码元的至少一个子集的步骤没有按照期望的准确度重新生成输入码元，则从多个重新生成的冗余码元和多个重新生成的输入码元重新生成至少一些未经重新生成的输入码元。

37. 如权利要求 36 所述的方法，其特征在于冗余码元包括第一组冗余码元和第二组冗余码元，其中至少重新生成一些未经重新生成的输入码元的步骤包括：

从第一组冗余码元的重新生成的冗余码元以及多个经重新生成的输入码元，重新生成未经重新生成的输入码元的至少一个以及第二组冗余码元的未经重新生成冗余码元；以及

如果从第一组冗余码元的重新生成冗余码元和多个重新生成输入码元的重新生成步骤没有按期望的准确度重新生成输入码元，则从第二组冗余码元的冗余码元和多个解码后输入码元重新生成至少一个未经重新生成的输入码元。

38. 如权利要求 37 所述的方法，其特征在于一些未经重新生成的输入码元和第二组冗余码元的未经重新生成的冗余码元使用 LDPC 解码器重新生成，以及

其中一些输入码元使用汉明解码器从第二组冗余码元的冗余码元重新生成。

39. 如权利要求 36 所述的方法，其特征在于重新生成至少一些未经重新生成的输入码元的步骤包括重新生成所有未经重新生成的输入码元。

40. 如权利要求 36 所述的方法，其特征在于重新生成组合集合内的码元子集合的步骤以及重新生成至少一些未经重新生成的输入码元的步骤包括：

形成第一矩阵，所述第一矩阵为每个接收到的输出码元指示与输出码元相关联的组合

集合内的码元；

使用信息增广所述第一矩阵，所述信息为每个冗余码元指示与冗余码元相关的输入码元；以及

重新生成至少一些输入码元，作为由增广的第一矩阵指明的方程组的解。

41. 如权利要求 36 所述的方法，其特征在于 N 大于或等于输入码元数。

42. 如权利要求 36 所述的方法，其特征在于 N 小于输入码元数。

43. 如权利要求 36 所述的方法，其特征在于重新生成至少一些未经重新生成的输入码元包括重新生成所有的输入码元。

44. 如权利要求 36 所述的方法，其特征在于重新生成至少一些未经重新生成的输入码元包括重新生成少于所有的输入码元。

45. 如权利要求 36 所述的方法，其特征在于重新生成码元的至少一子集的步骤在接收到预定数量 N 的任何输出码元后执行。

46. 如权利要求 36 所述的方法，其特征在于重新生成码元的至少一子集的步骤是在接收到数量 N 的任何输出码元后执行，其中的 N 使得输入码元可以被重新生成达到所希望的准确度。

47. 如权利要求 36 所述的方法，其特征在于重新生成码元的至少一子集的步骤是与接收到输出码元大致并发地执行的。

48. 一种用于接收在通信信道上从源发送的数据的系统，其特征在于包括：

接收模块，所述接收模块耦合到通信信道用于接收在通信信道上发送的输出码元，其中每个输出码元从输入码元和冗余码元的组合集合内的至少一个码元生成，其中至少一个输出码元从组合集合内多于一个码元以及组合集合内少于所有码元中生成，其中可能的输出码元数远远大于码元组合集合内的码元数，其中输入码元来自输入码元的排序集合，其中冗余码元从输入码元生成；

动态解码器，在接收到输出码元的至少一个子集后，对来自 N 个输出码元的组合集合内的码元子集进行解码，所述子集包括多个解码后的输入码元和多个解码后的冗余码元；以及

静态解码器，如果存在未经解码的输入码元，则从多个解码后的冗余码元对未经解码的输入码元的至少一些进行解码。

49. 如权利要求 48 所述的系统，其特征在于所述静态解码器包括 LDPC 解码器。

50. 如权利要求 48 所述的系统，其特征在于冗余码元包括第一组冗余码元和第二组冗余码元，其中静态编码器包括：

第一静态解码器，用于从第一组冗余码元的解码后冗余码元和多个解码后输入码元，对未经解码的输入码元和第二组冗余码元的未经解码的冗余码元的至少一个进行解码；

第二静态解码器，用于从第二组冗余码元的冗余码元和多个解码后的输入码元对至少一个未经解码的输入码元解码。

51. 如权利要求 50 所述的系统，其特征在于第一静态编码器包括 LDPC 解码器，且其中第二静态解码器包括汉明解码器。

52. 如权利要求 48 所述的系统，其特征在于动态解码器包括一处理器，用于执行以下步骤：

形成第一矩阵，所述第一矩阵为每个接收到的输出码元指示与输出码元相关联的组合集合内的码元；

使用信息增广所述第一矩阵，所述信息为每个冗余码元指示与冗余码元相关联的输入码元；以及

重新生成至少一些输入码元，作为由增广的第一矩阵指明的方程组的解。

用于通信系统的多级码发生器和解码器

[0001] 本发明背景

[0002] 本发明涉及通信系统内的编码和解码，尤其是对数据进行编码和解码以考虑通信的数据内的差错和间隔。

[0003] 在通信信道上发送者和接收者间的文件传输是许多文献的主题。最好，接收者期望能接收到有一定确信度的发送者在信道上发送的数据的准确副本。当信道没有完美的确信度时（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丢失或误传。丢失的数据（擦除）比受损的数据（差错）要更容易处理，因为接收者不能知道何时接收到了错误的受损数据。研发了许多差错纠正编码以纠正擦除和/或差错。一般，使用的特定码是基于数据发送通信的信道的不保真度和发送的数据的性质而经选择的。例如，在信道有较长时间的不保真度时，最适合该情况的是纠突发差错码。在只有较短不频繁的差错时，最好是简单的一致校验码。

[0004] 在选择编码时的另一考虑是用于传输的协议。在称为“因特网”的全球互连网络中，使用分组协议用于数据传输。该协议被称为互联网协议或简称“IP”。当文件或其他数据块在IP网络上传输时，它被分成等大小的输入码元，且输入码元被放入连续的分组。不管输入码元是否实际被分成比特流，输入码元的“大小”可以用比特测量，其中当输入码元从2^M的字母表中选出，输入码元大小为M比特。在该种基于分组的通信系统中，面向分组的编码方案比较合适。如果接收者能在网络内有擦除的情况下恢复原始文件的准确副本，则称该传输是可靠的。在因特网上，分组丢失经常发生，因为偶发的拥塞引起路由器内的缓冲机制达到其极限，迫使它丢失进入的分组。传输期间的擦除保护是许多研究的主题。

[0005] 传输控制协议（“TCP”）是有确认机制的一种普遍使用的点到点分组控制方案。TCP对于一到一的通信性能颇佳，其中发送者和接收者在何时进行传输并接收传输以及使用的发射机和接收机上取得一致。然而，TCP一般不适合一到多或多到多的通信，其中发送者和接收者独立地确定何时以及哪里他们会发送和接收数据。

[0006] 使用TCP，发送者发送排序的分组，接收者确认每个分组的接收。如果丢失了分组，则没有确认会被发送回发送者，且发送者会重发分组。分组丢失有多个原因。诸如TCP/IP协议中，确认规则使得分组不是完全失败地丢失，因为丢失的分组的可以或根据没有确认或根据来自接收者的显式请求而被重发。无论哪种方法，确认协议要求来自接收者到发送者的返回信道，该信道在丢失的分组数很大时使用率非常高。

[0007] 虽然基于确认的协议一般适用于许多应用，且实际上在当前的因特网上被广泛使用，但是它们效率不高，且有时对于诸如在给Michael G.Luby的美国专利号6307487内可以完全不可行，该专利题为“Information Additive CodeGenerator and Decoder for Communication Systems”（以下被称为“Luby I”）。另外，基于确认的协议不适合广播，其中一个发送者同时发送一个文件到多个用户。例如，假设发送者在卫星信道上广播一个文件到多个接收者。每个接收者经历不同的分组丢失模式。依赖确认数据（或是肯定或是否定）以进行可靠的文件发送的确认协议需要来自每个接收者到发送者的返回信道，要提供这些信道代价极大。另外，这需要复杂并强大的发送者以能合适地处理来自接收者的所有

确认数据。另一缺点是如果不同的接收者丢失不同的分组集合，重新广播只有一些接收者丢失的分组会引起其他接收者无用的重复分组接收。

[0008] 有时在实际中使用的基于确认协议的另一方法是基于旋转的协议。旋转协议将输入文件分成等长度的输入码元，将每个输入码元放入分组，并连续循环通过并发送所有分组。该旋转协议的主要缺点是如果接收者即使丢失了一个分组，则接收者必须等待整个循环才能接收到丢失的那个分组。另一种角度看，即基于旋转的协议会引起大量的无用复制数据接收。例如，如果接收者接收到来自旋换开始的分组，停止接收一段时间，然后开始在旋转的开始再次接收，则会接收到大量无用的重复分组。

[0009] 一种提出的解决以上问题的方案是避免使用基于确认的协议，而是使用前向差错纠正 (FEC) 码，诸如 Reed-Solomon 码或 Tornado 码，或是信息可加码的链式反应码以增加可靠性。使用这些编码，输出码元从内容中被生成，且被发送，而不是只发送组成内容的输入码元。擦除纠正编码诸如 Reed-Solomon 或 Tornado 编码为固定长度的内容生成固定数目的输出码元。例如，对于 K 个输入码元，可能生成 N 个输出码元。这些 N 个输出码元可以包括 K 个原始输入码元以及 N-K 个冗余码元。例如存储允许，则服务器可以为每个内容只计算输出码元集合一次，并使用旋转协议发送输出码元。

[0010] 一些 FEC 编码的一个问题在于需要过度的计算功率或存储器进行操作。另一问题是输出码元的数目必须在编码过程前预先被确定。如果过度估计分组丢失率则会导致低效，且如果过低估计分组丢失率，则会导致失败。

[0011] 对于传统的 FEC 编码，可以生成的可能的输出码元数与内容被划分的输入码元数是相同量级的。一般，大多数或所有的这些输出码元在发送步骤前的预处理步骤内生成。这些输出码元的特性为所有的输入码元可以从长度等于原始内容或比原始内容稍微长一点的输出码元任何子集合内重新生成。

[0012] Luby I 内描述的实施例（在此被称为“链式反应码”）提供了一种不同于前向差错纠正的形式，解决以上问题。对于链式反应码，可以被生成的可能输出码元池一般数量要大于输入码元的数目，且来自可能池的随机输出码元可以很快地被生成。对于链式反应码，输出码元可以按在发送步骤并发基础上需要的在运行中被生成 (on the fly)。链式反应码的特性是内容的所有输入码元可以从长度略微长于原始内容的随机生成输出码元集合的子集合中重新生成。

[0013] 在链式反应码的一实施例中，每个输出码元作为一些输入码元的异或 (XOR，用⊕表示）而获得。如果 K 表示总输入码元数，则每个输出码元平均上是 c*ln(K) 个输入码元的 XOR，其中 ln(K) 是 K 个自然对数，且 c 是合适的常量。例如，当 K 等于 60000 时，每个输出码元平均上是 28.68 个输入码元的 XOR，且当 K = 10000 时，每个输出码元平均上是 22.86 个输入码元的 XOR。大量的 XOR 导致输出码元更长的计算时间，因为每个该种操作涉及从存储器获取数据、实现 XOR 操作并更新存储器位置。

[0014] 链式反应码器生成的输出码元的一特性是接收机能一旦接收到足够的输出码元时能恢复原始文件。尤其是，为了以较高概率恢复原始 K 输入码元，接收机需要大致 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，且取决于解码器的可靠性。例如，在一特定实施例中，且 K 等于 60000，5％的相对接收开销保证解码器以至少 $1-10^{-8}$ 的概率成功地对输入文件解码，且当 K 等于 10000 时，15％的相对

接收开销保证解码器相同的成功概率。在一实施例中，链式反应码的相对接收开销可以由(13*sqrt(k)+200)/K 而计算，其中 sqrt(K) 是输入码元 K 个数目的平方根。在该实施例中，链式反应码的相对接收开销对于小值的 K 更大。

[0015] 在一些实施例中，输出码元使用 XOR 函数进行编码，连锁解码器的主计算操作实现存储器位置的 XOR。该 XOR 的数目的大小比例与链式反应码器相同。

[0016] 链式反应码在基于分组网络的通信非常有用。然而，其计算量也相对较强。例如，在链式反应码的一些特定实施例中，当输入码元 K 的数目为 60000 时，则计算每个输出码元需要平均获取 28.68 个随机选择的输入码元并对其进行 XOR。由于服务器可以同时服务的文件数与每个输出码元需要的操作数成反比，则减少每个输出码元需要的操作数会有用。例如减少后者例如为 3 倍，则增加了可以从一个服务器被同时服务的文件数为 3 的因子。

[0017] 链式反应码的另一特性需要接收开销，该开销对于给定的目标成功概率相对较大。例如，如上指出，在链式反应码的一些特定实施例中，如果 K = 10000，则 15％的相对接收开销保证了至少 $1-10^{-8}$ 的解码成功概率。接收开销对于更小的 K 要增加。例如，在一些链式反应码的特定实施例中，如果 K 为 1000，则 61％的相对接收开销保证了相同概率的成功解码。而且，降低目标差错概率到 10^{-12} 左右的一数目，如在诸如卫星网络上的高速内容传输的一些应用中要求的，该数目需要更大的接收开销。

[0018] 本发明的简要描述

[0019] 根据本发明的一实施例，提供了一种用于在通信信道上从源到目的地的数据传输编码的方法。该方法对输入码元的排序集合进行操作，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合生成多个输出码元，其中可能的输出码元数远远大于码元的组合集合内的码元数，其中从码元的组合集合内多于一个码元且从码元的组合集合内少于所有码元中生成至少一个输出码元，使得输入码元的排序集合可以从输出码元的任何预定数经重新生成达到期望的准确度。

[0020] 根据本发明另一实施例，提供了一系统，用于在通信信道上从源到目的地的数据编码。该系统包括被耦合以接收多个输入码元的静态编码器，多个输入码元从要发送的数据中生成。静态编码器包括冗余码元发生器，所述发生器基于输入码元生成多个冗余码元。系统附加地包括被耦合以接收多个输入码元和多个冗余码元的动态编码器，动态编码器包括输出码元发生器，所述发生器从包括多个输入码元和多个冗余码元的码元组合集合中生成多个输出码元，其中可能输出码元数远远大于组合集合内的码元数，其中从多于组合集合的码元且小于组合集合的所有码元中生成至少一个输出码元，使得输入码元的排序集合可以从输出码元的任何预定数中重新生成达到期望的准确度。

[0021] 根据本发明的另一实施例，提供了在通信信道上对从源发送的数据进行接收的方法，该方法包括接收输出码元，其中每个输出码元从输入码元和冗余码元的组合集合内的至少一个码元生成，其中至少一个输出码元从多于组合集合内的一个码元且小于组合集合内所有码元中生成，其中可能的输出码元数远远大于组合集合内的码元数，其中输入码元来自输入码元的排序集合，其中冗余码元从输入码元中生成。该方法还包括在接收到任何预定数目 N 输出码元时，从 N 个输出码元重新生成组合集合内至少一个码元的子集，所述子集包括多个重新生成的输入码元以及多个重新生成的冗余码元。该方法进一步包括如果从 N 个输出码元重新生成码元的至少一个子集的步骤没有按照期望的准确度重新生成输入码

元，则从多个重新生成的冗余码元和 / 或多个重新生成的输入码元的一些重新生成至少一些未经重新生成的输入码元。

[0022] 根据本发明的另一实施例，提供一系统用于接收在通信信道上从源发送的数据。该系统包括一接收模块，所述模块耦合到通信信道用于接收在通信信道上发送的输出码元，其中每个输出码元从输入码元和冗余码元的组合集合内的至少一个码元生成，其中至少一个输出码元从组合集合内多于一个码元以及组合集合内少于所有码元中生成，其中可能的输出码元数远远大于组合集合内的码元数，其中输入码元来自输入码元的排序集合，其中冗余码元从输入码元生成。系统还附加地包括动态解码器，在接收到输出码元的预定数量 N 后，对来自 N 个输出码元的组合集合内的码元子集进行解码，所述组合集合内的码元子集包括多个解码后的输入码元和多个解码后的冗余码元。系统还包括静态解码器，从多个解码后的冗余码元对未经解码的输入码元（如果有的话）的至少一些进行解码。

[0023] 根据本发明的另一实施例，提供体现在载波上的计算机数据信号。计算机数据信号包括多个输出码元，其中多个输出码元表示从包括输入码元和冗余码元的排序集合的码元组合集合生成的码元，其中冗余码元从输入码元中被生成，其中可能的输出码元数远远大于码元组合集合内的码元数，其中至少一个输出码元从码元的组合集合内多于一个码元以及码元组合集合内少于所有码元中生成，且使得数据信号接收机能从输出码元的任何预定数量按期望的准确度重新生成输入码元的排序集合。

[0024] 本发明可以获得多种好处。例如，在特定实施例中，减少了在信道上传输的数据编码的计算化费。在另一特定实施例中，减少了对该种数据解码的计算代价。取决于实施例，可以获得一个或多个该种好处。这些和其它好处在本发明说明中更详细地描述，尤其在以下。

[0025] 在此揭示的本发明的性质和优势的进一步理解可以通过参考规范和所附附图实现。

[0026] 附图的简要描述

[0027] 图 1 是根据本发明的一实施例的通信系统框图；

[0028] 图 2 是根据本发明的一实施例的编码器框图；

[0029] 图 3 是根据本发明的一实施例生成冗余码元方法的简化框图；

[0030] 图 4 是根据本发明的一实施例的静态编码器基本操作的简化框图；

[0031] 图 5 是根据本发明的一实施例的动态编码器的简化框图；

[0032] 图 6 是根据本发明的一实施例的动态编码器基本操作的简化框图；

[0033] 图 7 是根据本发明一实施例的静态编码器的简化框图；

[0034] 图 8 是根据本发明一实施例的静态编码器的基本操作简化框图；

[0035] 图 9 是根据静态编码器的特定实施例用于计算编码参数的方法简化图；

[0036] 图 10 是根据本发明的另一实施例的静态编码器的简化流图；

[0037] 图 11 是根据本发明的一实施例的解码器的简化框图；

[0038] 图 12 是根据本发明的一实施例的解码器操作简化流图；

[0039] 图 13 是根据本发明的另一实施例的解码器操作简化流图；

[0040] 图 14 是根据本发明的另一实施例的解码器操作的简化流图；

[0041] 图 15 是根据本发明的一实施例的动态解码器的简化框图；

[0042] 图 16 是根据本发明的一实施例的静态解码器的简化框图；

[0043] 图 17 是根据本发明的另一实施例的静态解码器的简化框图；

[0044] 图 18 是根据本发明的实施例的解码器操作的简化流图；

[0045] 图 19 是根据本发明的实施例的解码器另一操作的简化流图；

[0046] 图 20 是根据本发明的一实施例的关联器的简化流图；

[0047] 图 21 是根据本发明的一特定实施例的加权选择器的简化框图；

[0048] 图 22 是根据本发明的实施例被加权选择器使用的过程简化流图。

[0049] 特定实施例的详细说明

[0050] 本揭示参考美国专利号 6307487(Luby I) 和美国专利号 6320520，两个专利发布给 Michael G. Luby，题为“Information Additive Group Code Generatorand Decoder for Communication Systems”（此后被称为“Luby iII”），整个揭示在此引入作为各种目的的参考。Luby I 和 II 提供可以用于根据本发明的一些实施例中使用的系统的方法的原理。然而可以理解，本发明不需要这些系统和方法，还可以使用其它的变体、修改或其它。

[0051] 在此描述的特定实施例中，描述了被称为“多级编码”的编码方案，先对在该描述中使用的各项的意义和范围作解释。

[0052] 多级编码如在此描述的对数据在多级内进行编码。一般但不总是，第一级向数据加入预定量的冗余。第二级然后使用链式反应码或类似的，以从原始数据生成输出码元，以及由第一级编码计算的冗余码元。在本发明的一个特定实施例中，接收到的数据首先使用连锁解码过程进行解码。如果该过程不能成功地完整恢复原始数据，则应用第二解码步骤。

[0053] 在此描述的一些实施例的好处在于与以下将描述的链式反应码相比，只需要较少的算术操作。一些包括第一级编码和第二级编码的特定实施例的其它优势在于编码的第一级和编码的第二级可以在分开的时间和 / 或分开的设备完成，因此划分了计算负载。例如如果希望在传输时并发地实现编码的一部分，这会是有利的。特别是，第一级编码可以在传输前实现，而第二级编码可以与传输并发地实现。然而可以理解，在一些实施例中，第一和第二级编码可以与 / 或一个设备大致与传输并发地进行。许多其它的变体、修改和变化对于领域内的技术人员在阅读本申请后会明显。

[0054] 在多级编码的实施例中，冗余码元在第一级编码时从输入文件生成。在这些实施例中，在第二级编码中，输出码元从输入文件和冗余码元的组合中生成。在一些这些实施例中，输出码元可以如需要被生成。在第二级包括链式反应码的实施例中，每个输出码元可以不用管其它输出码元是如何生成的而经生成。一旦生成了，这些输出码元然后可以放入分组并发送到其目的地，每个分组包含一个或多个输出码元。还可以使用非分组化传输技术作为替代。

[0055] 如在此使用的，“文件”一词指任何存储在一个或多个源处且作为一个单元发送到一个或多个目的地的数据。因此，来自文件服务器或计算机存储设备的文档、图像和文件是可以被发送的“文件”的示例。文件可以是已知的大小（诸如存储在硬盘上的一兆字节图像）或可以是未知的大小（诸如从流源输出获得的文件）。无论是哪种，文件是输入码元的序列，其中每个输入码元在文件内有一位置和值。

[0056] 传输是通过信道从一个或多个发送者到一个或多个接收者的数据发送过程以传送一文件。发送者有时还被称为编码器。如果一个发送者通过完美信道连接到任何数量的

接收者，如果所有的数据都被正确接收，则接收到的数据可以是输入文件的准确副本。在此，我们假设信道不是完美的，这一般是实际信道的情况。在许多信道不完美中，两种在此感兴趣的是数据擦除和数据不完整（这可以被视为数据擦除的特殊情况）。数据擦除发生在信道丢失或丢弃数据时。数据不完整发生在当接收者在一些数据已经经过而接收者还未开始接收数据、当接收者在传输结束前停止接收数据、当接收者只选择接收一部分发送的数据和／或当接收者间歇断地停止并再次开始接收数据。作为数据不完整的示例，移动卫星发送者可能发送表示输入文件的数据并在接收者在范围内前开始传输。一旦接收者在范围内，数据可以被接收直到卫星移出范围，在该点接收者可以重新调整其碟形卫星天线（在该时间内它不接收数据）以开始接收另一移入范围内的卫星发送的相同输入文件的数据。如从读该描述中清楚看出的，数据不完整性是数据擦除的特殊情况，因为接收者可以将数据不完整性（且接收者有相同的问题）视为如同接收者整段时间都在，但信道丢失了在接收者开始接收数据前的所有数据。而且，如在通信系统设计内已知的，可检测差错可以被认为等价于简单地丢弃所有有检测到差错的数据块或码元的擦除。

[0057] 在一些通信系统中，接收者接收多个发送者生成的数据，或是使用多个连接的一个发送者的数据。例如为了加快下载，接收者可以同时连接到多于一个发送相同文件的发送者。作为另一示例，在多播传输中，多个多波数据流可以被发送以使的接收者能将一个或多个这些流连接以匹配与连接到发送者的信道带宽匹配的集体传输速率。在所有该种情况下，要保证所有发送的数据对于接收者是独立使用的，即多个源数据在流间不是冗余的，即使当传输速率对于不同流非常不同时或当有任意丢失模式时。

[0058] 一般，通信信道是连接数据传输的发送者和接收者的信道。通信信道可以是实时信道，其中信道在信道获得数据时将数据从发送者移到接收者，或通信信道可能是存储信道，该信道在将数据从发送者转移到接收者过程中存储一些或所有的数据。后者的示例是盘存储或其它存储设备。在该示例中，生成数据的程序或设备可以被认为是发送者，将数据发送到存储设备。接收者是从存储设备读取数据的程序或设备。发送者使用的将数据送到存储设备的机制、存储设备本身以及接收者使用的从存储设备获得数据的机制共同形成了信道。如果这些机制或存储设备会丢失数据，则它可以被视为通信信道内的数据擦除。

[0059] 当发送者和接收者为通信信道隔开，且该通信信道内码元被擦除，则最好不要发送输入文件的准确副本，而是发送从输入文件生成的数据，该数据能帮助恢复擦除。编码器是处理该任务的电路、设备模块或编码片段。一种观看编码器操作的方式是编码器从输入码元生成输出码元，其中输入码元值序列表示输入文件。每个输入码元因此在输入文件内有一位置和一值。解码器是电路、设备、模块或编码片段，它们从接收者接收到的输出码元重建输入码元。在多级编码中，编码器和解码器进一步被分成子模块，每个实现不同的任务。

[0060] 在多级编码系统的实施例中，编码器和解码器可以被进一步分成子模块，每个实现不同的任务。例如，在一些实施例中，编码器包括在此被称为静态编码器和动态编码器。如在此使用的，“静态”编码器是从输入码元集合生成多个冗余码元的编码器，其中冗余码元数在编码前被确定。静态编码码示例包括 Reed-Solomon 编码、Tornado 编码、汉明码、低密度一致校验 (LDPC) 编码等。“静态解码器”一词在此被称为解码器，该解码器可以对由静态编码器编码的数据解码。

[0061] 如在此使用的，“动态编码器”是从输入码元集合生成输出码元的编码器，其中可能的输出码元的数量级大于输入码元数，且其中要生成的输出码元数不需要是固定的。动态编码器的一例是链式反应码器，诸如 Luby I 和 Luby II 内描述的编码器。“动态解码器”一词在此用于指可以对由动态编码器编码的数据解码的解码器。

[0062] 多级编码的实施例不受到任何特定类型的输入码元的限制。一般，输入码元的值从对一些正整数 M 的 2^M 个码元的字母表中被选择。在该情况下，输入码元可以用来自输入文件的 M 比特数据的序列表示。M 的值一般基于例如使用的应用程序、通信信道和 / 或输出码元的大小而确定。另外，输出码元的大小还经常基于应用、信道和 / 或输入码元的大小而经确定。在一些情况下，如果输出码元值和输入码元值大小相同（即用相同数量的比特表示，或从相同的字母表选出），则编码过程可以被简化。如果是该情况，则输入码元值大小在输出码元值受限时受到限制。例如，可能期望将输出码元放入有限大小的分组内。如果与输出码元相关的密钥的一些数据为了在接收机处恢复密钥而被发送，则输出码元最好较小，以在一个分组内包括输出码元值和关于密钥的数据。

[0063] 作为一例，如果输入文件是多兆字节的文件，则输入文件可能被分成几千、几万或几百万的输入码元，每个输入码元对几千、几百或几个字节编码。作为另一示例，对于基于分组的因特网信道，1024 字节大小的有效负荷分组可能比较合适（一字节为 8 比特）。在该示例中，假设每个分组包含一个输出码元和 8 字节辅助信息，输出码元大小为 8128 比特 ((1024-8)*8) 比较合适。因此，输入码元大小可以被选为 M = (1024-8)*8，即 8128 比特。作为另一示例，一些卫星系统使用 MPEG 分组标准，其中每个分组的有效负荷包括 188 个字节。在该示例中，假设每个分组包含一个输出码元和 4 字节辅助信息，则输出码元大小为 1472 比特 ((188-4)*8) 比较合适。因此，输入码元大小可以被选为 M = (188-4)*8，即 1472 比特。在使用多级编码的通用通信系统内，诸如输入码元大小（即 M 为输入码元编码的比特数）的应用专用参数可以是应用设定的变量。

[0064] 每个输出码元有一个值。在一个以下考虑的最优实施例中，每个输出码元还与一标识符相关，该标识符被称为“密钥”。最好，每个输出码元的密钥可以简单地由接受者确定以允许接收者能互相区别输出码元。最好，一输出码元的密钥不同于其他输出码元的密钥。在本领域内有许多各种形式的密钥。例如，Luby I 描述了各种形式的可以在本发明的实施例中使用的密钥。

[0065] 多级编码在期望有数据擦除或接收者不在传输正好开始和结束的时候开始时特别有用。后者的情况在此被称为“数据不完整性”。关于擦除事件，多级编码共享许多在 Luby I 内描述的链式反应码的好处。特别是，多级输出码元是信息可加的，所以任何合适数量的分组可以被用于以一定准确性恢复输入文件。这些条件不会负面影响使用多级编码的通信过程，因为用多级编码生成的输出码元是信息附加的。例如，如果因为引起数据擦除的噪声突发而丢失了一百个分组，则可以在突发后发送额外的一百个分组以替代擦除的分组的丢失。如果因为由于接收机没有调谐到发射机开始发送时的发射机而丢失了成千个分组，则接收机可以从任何其他传输时间段甚至从另一发射机开始重新接收这些成千个分组。使用多级编码，接收机不限于接收任何特定分组集合，所以它可以从一个发射机接收一些分组，切换到另一发射机，丢失一些分组，失去给定传输的开始或结束，且仍恢复输入文件。加入并离开传输而不需要接收机 - 发射机协调的能力可以帮助简化通信过程。

[0066] 在一些实施例中，使用多级编码发送文件可以包括从输入文件生成、形成或抽取输入码元，计算冗余码元、将输入和冗余码元编码成为一个或多个输出码元，其中每个输出码元基于其独立于所有其他输出码元的密钥而经生成，并将输出码元在信道上发送到一个或多个接收者。另外，在一些实施例中，使用多级编码接收（重建）输入文件的副本可以包括从一个或多个数据流接收输出码元的一些集合或子集，并从值和接收到的输出码元的密钥对输入码元解码。

[0067] 本发明的各方面现在将参考附图进行描述。

[0068] 系统概述

[0069] 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，向输入码元发生器 110 提供输入文件 101 或输入流 105。输入码元发生器 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)，...)，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发生器 110 的码元大小输入，使得 M 可以随每次使用而改变。输入码元发生器 110 的输出可以被提供给编码器 115。

[0070] 静态编码器 130 生成静态密钥流 S_0，S_1...。生成的静态密钥数一般是有限的，且取决于编码器 115 的特定实施例。静态密钥的生成在以下将接着更详细地描述。动态密钥发生器 120 为每个要由编码器 115 生成的输出码元生成一动态密钥。每个动态密钥被生成，使得同一输入文件的动态密钥的很大部分是唯一的。例如，Luby I 描述可以使用的密钥发生器的实施例。动态密钥发生器 120 和静态密钥发生器 130 的输出被提供给编码器 115。

[0071] 从动态密钥发生器 120 提供的每个密钥 I，编码器 115 从输入码元发生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。每个输出码元的值基于其密钥、一个或多个输入码元的一些函数以及可能从输入码元经计算的一个或多个冗余码元被生成。引起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”或只是其“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。

[0072] 在一些实施例中，输入码元数 K 为编码器 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 还可以为编码器 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。

[0073] 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密钥发生器 120 的每个该种输出码元的密钥。发射模块 140 发送输出码元，且取决于使用的密钥方法，发送模块 140 可能在信道 145 上将一些关于发送的输出码元的密钥的一些数据发送到接收模块 150。信道 145 被假设为擦除信道，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、物理媒质或其任何组合，只要发射模块 140 用于将输出码元和任何关于密钥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，以及潜在的一些关于其密钥的数据。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。

[0074] 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。

[0075] 由于信道 145 被假设是擦除信道，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，输出码元的密钥使用一个或多个上述的密钥方案被确定，且不一定按输出码元离开接收模块 150 的顺序确定。

[0076] 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密钥，这些密钥被提供给动态密钥重新发生器 160。动态密钥重新发生器 160 重新生成接收到的输出码元的动态密钥并将这些动态密钥提供给解码器 155。静态密钥发生器 163 重新生成静态密钥 S_0，S_1，... 并将其提供给解码器 155。静态密钥发生器访问在编码和解码过程中都使用的随机数发生器 135。如果随机数在该种设备上生成，则这可以是以对相同物理设备的访问形式，或是对用于生成随机数的相同算法的访问形式以获得相同的行为。解码器 155 使用动态密钥发生器 160 和静态密钥发生器 163 提供的密钥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组装器 165，该组装器生成输入文件 101 的副本 170 或输入流 105。

[0077] 编码器

[0078] 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) 输入码元发生器 110 提供的原始输入码元 (IS(0)，IS(1)，...，IS(K-1)) 并存储在输入码元缓冲器 205 内 ；b) 原始输入码元数 K ；c) 由静态密钥发生器 130 提供的静态密钥 S_0，S_1，... ；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)，...，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，即文件可以物理地被存储在一个地方，且码元缓冲器 205 内的输入码元的位置只能是原始文件内的这些码元位置的重命名。

[0079] 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。

[0080] 冗余计算器 230 从 K 个输入码元计算 R 个冗余码元。该计算在以下详述。

[0081] 在生成输出码元的速度是关键因素的情况下，输入文件可以使用静态编码器 205 被编码并在输出码元传输开始前存储在中间设备上。该设备可以是例如附加的位于不同于与动态编码器 220 的物理位置的存储设备，它可以被包括在与动态编码器 220 等相同的物理设备内等。在使用动态编码器 220 编码前文件使用静态编码器 205 被编码的情况下，实现动态编码器 220 的计算设备不需要将资源用于静态编码。因此，它可以将资源更多地用于动态编码以例如增加输入文件的生成输出码元速度、生成其它文件的输出码元、实现其

他任务等。静态编码是否能或应该在动态编码前实现取决于特定的实现。

[0082] 静态编码器概览

[0083] 静态编码器 210 的一般操作参考图 3 和 4 描述。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) 的至少一些的 F_0 的子数被计算。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 经计算，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。

[0084] 回到图 1 和 2，在一些实施例中，静态编码器 210 从静态密钥发生器 130 接收一个或多个静态密钥 S_0，S_1，...。在这些实施例中，静态编码器 210 使用静态密钥以确定一些或所有的函数 F_0，F_1，...F_{j-1}。例如，静态 S_0 可以用于确定函数 F_0，静态密钥 S_1 可以被用于确定函数 F_1 等。或一个或多个静态密钥 S_0，S_1，... 可以被用于确定函数 F_0，一个或多个静态密钥 S_0，S_1，... 可以用于确定函数 F_1 等。在其他实施例中，不需要静态密钥，因此不需要静态密钥发生器 130。

[0085] 回到图 2 和 3，在一些实施例中，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。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，..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，...F_{R-1} 包括一些或所有它们对应的参变量的异或。如上描述，这些函数可以或实际上可能不能使用图 1 的静态密钥发生器 130 生成静态密钥。例如，在以下描述的一个特定实施例，第一几个函数实现汉明码且不使用静态密钥 S_0，S_1，...，其中剩余的函数实现低密度一致校验编码并显式使用静态密钥。

[0086] 动态编码器概览

[0087] 回到图 2，动态编码器 220 接收输入码元 IS(0)，...，IS(K-1) 以及冗余码元 RE(0)，...，RE(R-1) 以及要生成的每个输出码元的密钥 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 是动态编码器的一实施例的简化框图。该编码器类似于 Luby I 内描述的一实施例。Luby I 描述该种编码器的操作的进一步细节。

[0088] 动态编码器 500 包括加权选择器 510、关联器 515、值函数选择器 520 和计算器 525。如图 5 示出，K+R 个动态输入码元被存储在动态码元缓冲器 505 内。在一实施例中，动态码元缓冲器 505 是图 2 的输入码元缓冲器 205。在另一实施例中，动态码元缓冲器 505 与输入码元缓冲器 205 分开。动态密钥 I（由图 1 示出的动态密钥发生器 120 提供）是加权选择器 510、关联器 515 和值函数选择器 520 的输入。动态输入码元数 K+R 被提供给这些三个组件 510、515 和 520。计算器 525 经耦合以接收来自加权选择器 510、关联器 515 和值函数选择器 520 的输出并接收来自动态码元缓冲器 505 的码元。计算器 525 生成输出码元值。应理解可以使用图 5 内示出的元件的等价安排，且这里只是根据本发明的编码器的一个示例。例如，Luby I 和 Luby II 描述其他可以用于根据本发明其他实施例的编码器。

[0089] 在操作中，从输入码元缓冲器 205 接收 K+R 个动态输入码元，并存储在动态输入码

元缓冲器 505 内。如上解释的，每个动态输入码元有一位置（例如，输入码元的位置可以是其在输入文件内的原始位置）和值。动态输入码元不需要按其相应的顺序存储在动态输入码元缓冲器 505 内，只要能确定存储的动态输入码元的位置。

[0090] 使用密钥 I 和动态输入码元数目 K+R，加权选择器 510 确定是要与带有密钥 I 的输出码元的“关联”的动态输入码元数目 W(I)。使用密钥 I、加权 W(I) 以及动态输入码元数 K+R，关联器 515 确定与输出码元相关的动态输入码元位置列表 AL(I)。应理解如果关联器 515 能事先在不知道 W(I) 情况下生成 AL(I)，则 W(I) 不需要分开或显式地计算。一旦生成了 AL(I)，则 W(I) 可以被简单地确定，因为它是 AL(I) 内的关联的数目。

[0091] 并联器 515 是一映射，它接收输入密钥 I、数 N 以及数 t 并生成 0 到 N-1 间的整数的列表 X(0),...,X(t-1)。最好，这些整数不同且在其范围上均匀分布。例如，在图 5 的动态编码器 500 情况下，N 等于 K+R，t 等于 W(I)，且 AL(I) 是列表 X(0),...,X(t-1)。

[0092] 关联器 515 给出的映射可以采用各种形式。它可以访问真随机或伪随机比特源以使其输出随机。然而，对于相同密钥 I、相同 N 和相同 t，它应被选择生成编码器和解码器相同的输出。为了满足该要求，可以使用以密钥 I 为种子的编码器和解码器生成伪随机序列。取代伪随机序列，可以使用真随机序列以计算输出，但为了使之有用，用于生成输出的随机序列需要被传递到解码器。

[0093] 再回到图 5，一旦已知 I、W(I) 和 A(I)，输出码元的值 B(I) 由计算器 525 基于值函数 F(I) 经计算。合适值函数的一特点是它允许由 AL(I) 指明的关联的值从输出码元值 B(I) 和由 AL(I) 指明的其他 W(I)-1 个关联的值而被确定。 在该步骤内使用的一个最优值函数是 XOR 值函数，因为它满足该特性，它容易计算并容易求反。然而，还可以使用其他合适的值函数。例如 Luby II 描述了其他可以使用的合适值函数。

[0094] 如果被使用，值函数选择器 520 从密钥 I 和 K+R 确定值函数 F(I)。在一变体中，值函数 F(I) 对于所有 I 是相同的值函数 F。在该变体，值函数选择器器 520 不需要，且计算器 525 可以用值函数 F 经配置。例如，值函数可以对于所有 I 为 XOR，即输出码元值是所有其关联的值的 XOR（异 OR）。

[0095] 对于每个密钥 I，加权选择器 510 从 I 和 K+R 确定加权 W(I)。在一变体中，加权选择器 510 通过使用密钥选择 W(I)，以首先生成随机查找数，然后使用该数字在存储其中或由加权选择器 510 可访问的分布表格内查找 W(I) 的值。如何形成和访问该种分布表格在以下进行详细描述。一旦加权选择器 510 确定了 W(I)，该值被提供给关联器 515 和计算器 525。

[0096] 使用列表 AL(I)、加权 W(I) 和值函数选择器 520 提供的值函数 F(I) 或预选的值函数 F，计算器 525 访问动态输入码元缓冲器 505 内由 AL(I) 引用的 W(I) 个动态输入码元以为当前输出码元计算值 B(I)。计算 AL(I) 的过程示例如下，但可以使用其他的合适过程。最好，过程给予每个输入码元大致相等的机会被选择为给定输出码元的关联，且选择方式使得如果解码器没有可用的 AL(I)，则解码器可以复制。

[0097] 动态解码器 500 然后输出 B(I)。实际上，动态解码器 500 实现图 6 内说明的行为，即生成输出码元值 B(I) 作为选择的输入码元的一些值函数。在示出的示例中，值函数为 XOR，输出码元的加权 W(I) 为 3，关联联的动态输入码元（关联）在位置 0、2 和 K+R-2 且有相应的值 IS(0)、IS(2) 以及 RE(R-2)。因此对于 I 的值，输出码元被计算为：

[0098] $B(I) = IS(0) \oplus IS(2) \oplus RE(R-2)$

[0099] 在使用 XOR 值函数情况下，可以理解冗余码元有与原始码元 IS(0)，...，IS(K-1) 相同的比特数，且这些依次也与输出码元有相同的比特数。

[0100] 生成的输出码元然后被发送并如上述被接收。在此，假设可能丢失一些输出码元或可能排序出错，或它们可能由一个或多个编码器生成。然而假设被接收到的输出码元还接收到其密钥和其值 B(I) 准确度保证的指示。如图 1 示出，这些接收到的输出码元，连同从其动态密钥发生器 160、值 K+R 以及静态密钥发生器 163 生成的静态密钥 S_0,S_1,... 的指示重建的对应密钥是解码器 155 的输入。

[0101] 静态编码器

[0102] 静态编码器的主要功能是向原始数据加入冗余信息，使得在有擦除情况下有可能恢复原始数据。该种冗余信息可以帮助解码器恢复动态解码器不能恢复的输入码元。在一般的应用中，静态编码器在有擦除情况下保证能恢复到期望准确度需要的冗余码元数目的意义上是有效的，以及 / 或在编码过程和 / 或解码过程的计算代价上是有效的。例如，对于给定目标擦除率 p，该擦除率在应用中由动态解码器的性能决定，目标是使得冗余码元数 R 尽可能地小，而同时保证如果最多丢失数据的 p 部分时能快速恢复原始数据。满足该要求的一类码为 LDPC 码，这对于领域内的技术人员是已知的。虽然这些码可以在许多情况下恢复原始数据，但在不经常的情况下，这些码能恢复除了两三个原始输入码元外的所有码元。因此在一些实施例中，在 LDPC 编码前，输入数据首先使用在有两三个擦除情况下可以恢复原始数据的编码。第一编码生成第一组冗余码元。在第一编码后，多个原始码元和第一组冗余码元使用 LDPC 编码器经编码。扩展的汉明码对于本领域内的技术人员是众知的，且在以下简要描述，适合于第一层的编码的目的，因为它能在有两三个擦除的情况下恢复原始数据，且通过加入很小量的冗余码元而完成。可以理解还可以使用其他类型的编码。例如，在一些应用中，可以接受恢复了除两三个输入码元外的所有码元。因此，在该种应用中，LDPC 码单独就足够了。另外，其他类型的编码对于特定应用可能也是合适的，诸如 Reed-Solomon、Tornado 等。因此，可以理解根据本发明的其他实施例可以单独使用许多类型的编码或组合地使用。

[0103] 图 7 是根据本发明的静态编码器的一特定实施例简化框图。静态编码器 600 包括参数计算器 605、汉明编码器 610 以及低密度一致校验 (LDPC) 编码器 620。参数计算器 605 接收 K 个输入码元和要生成的 R 个冗余码元，并生成参数 D 和 E。D 指示由汉明编码器 610 生成的冗余码元数，E 指示由 LDPC 编码器 620 生成的冗余码元数。参数 D 提供给汉明编码器 610，参数 E 被提供给 LDPC 编码器 620。

[0104] 汉明编码器 610 经耦合以从输入码元缓冲器 625 接收输入码元 IS(0)，...，IS(K-1)、输入码元数 K 以及参数 D。作为响应，汉明编码器 610 根据汉明码生成 D+1 个冗余码元 HA(0)，HA(q)，...，HA(D)。在一实施例中，输入码元缓冲器 625 是图 2 的输入码元缓冲器 205。汉明编码过程向原始 K 个输入码元加入 D+1 个冗余码元，其中 D 是使得 $2^D-D-1 \geq K$ 的最小数。如本领域内的技术人员众知的，冗余码元的选择是使得对于所有的输入码元的可能设置，使得它们不是全为零，至少在多个输入码元以及对应的冗余码元中的至少四个不为零。该特性保证了至少能纠正三个擦除。汉明编码器 610 可以以任何领域内的技术人员已知的差错纠正和擦除纠正编码的多种方式实现。

[0105] LDPC 编码器 620 经耦合以接收输入码元 IS(0)，...，IS(K-1)、输入码元数 K+D+1 和经汉明编码的冗余码元、参数 E 以及静态密钥 S_0，S_1，...。作为响应，LDPC 编码器 620 根据 LDPC 编码生成 E 个冗余码元。LDPC 编码器计算的冗余码元的数目 E 等于 R-D-1，其中 R 是冗余码元数目。如本领域内的技术人员已知的，有多种使用 LDPC 码对信息编码的方式。LDPC 编码用图形结构的方式描述，该图形结构包括消息节点集合、校验节点集合以及连接消息节点到校验节点的边。有效 LDPC 码字集合是使得对于每个校验节点，相邻消息节点的 XOR 为零的消息节点的这些设置的集合。在一些应用中，最好消息节点有相同的度，即连接到相同数目的校验节点，这样简化了编码器的实现，且使得解码器的差错概率计算变得更简单。在本发明的一特定实施例中，每个消息节点连接到的校验节点的数目为四。已经知道该数目提供在编码器的运行时间 / 计算负载以及解码器的失败概率间可接受的折衷。而且，已知较佳地是随机地在校验节点集合间随机选择与给定消息节点相邻的校验节点。LDPC 编码器 620 可以以领域内技术人员已知的差错纠正和擦除纠正编码的任何一种方式实现。

[0106] 图 8 说明使用图 7 示出的静态编码器的本发明一实施例操作。尤其是，汉明编码器 610 从输入码元缓冲器 205（图 2）接收输入码元，并生成 D+1 个经汉明编码的冗余码元，这些码元被存储在输入码元缓冲器 205 内。然后，LDPC 编码器 620 从输入码元缓冲器 205 接受输入码元和 D+1 个经汉明编码的冗余码元，并生成 E 个 LDPC 经编码的冗余码元，这些码元被存储在输入码元缓冲器 205 内。

[0107] 如上所述，在一些实施例中，LDPC 编码器 620 接收图 1 的静态密钥发生器 130 生成的静态密钥 S_0，S_1，...。在一实施例中，静态密钥发生器 130 是随机数发生器，它在接收到种子 (seed) 后生成随机查询数序列（静态密钥 S_0，S_1，...）。种子可以取各形式。例如，可以是真随机数发生器的值。作为另一例，种子可以是从 CPU 时钟以确定方式获得的字符串。无论种子是什么，它应被传递到解码器，使得相同的静态密钥序列可以由解码器生成。在许多应用中，比较有利的是种子不太大。在许多应用中，种子是 32 比特整数，或 64 比特整数。

[0108] 在图 6 内说明的静态编码器 600 的一特定实施例中，参数 D 作为使得 2^D-D-1 大于或等于输入码元数 K 的最大整数而被计算。另外，参数 E 被计算为 R-D-1。图 9 是说明参数计算器的一实施例的简化流程图，诸如图 7 的参数计算器 605，该计算器如上所述计算参数 D 和 E。首先，在步骤 705，参数 D 被初始化为一。然后在步骤 710，确定 2^D-D-1 是否小于 K。如果不是，则流程进行到步骤 730。如果是，则流程进行到步骤 720，其中递增参数 D。然后，流回到步骤 710。一旦 D 经确定，则在步骤 730，参数 E 被计数为 R-D-1。

[0109] 再回到图 1，在一些特定应用中，在信道 145 上要发送的文件或流相当小。例如，输入文件可以是较短的音频消息或包括几万字节的 Web 网页内容。则上述的静态编码器的特定实施例可能在该种情况下性能次于最优。例如，一些上述的实施例会导致存储器和处理器速度的低效使用，因此减缓了数据的重建。而且，上述的一些实施例可能需要更大的接收开销以在由系统的用户设定的可靠性参数内重建数据。另外，一些上述的实施例可能导致重建的数据的可靠性小于所期望的。

[0110] 已知解码器的失败概率当输入码元数目减少时增加。而且还已知这很大程度上是因为如果原始内容相对较小，则编码过程没有建立关于原始内容的足够信息。因此，可以使

用编码器的另一实施例，它生成可以传递更多关于原始码元的信息的冗余码元。图 10 是根据本发明的一实施例的该种编码器的简化流程，以下将描述。

[0111] 首先在步骤 805，变量 i 被初始化为零。变量 i 跟踪已经生成的冗余码元数。在步骤 810，数字 t 作为大于或等于 K/2 的最小奇数而经计算。在步骤 815，值 $P_1, P_2, .., P_t$ 基于 K、t 和静态密钥 S_t 而经生成。值 $P_1, P_2, ..., P_t$ 指明用于生成冗余码元的输入码元位置。在一特定实施例中，，诸如图 5 的关联器 515 的关联器被用于生成 $P_1, P_2, ..., P_t$。特别是，值 t 可以作为 W(I) 输入被提供，值 K 可以作为 K+R 个输入被提供，且静态密钥 S_i 可以作为密钥 I 输入被提供。值得注意的是许多不同的 t 的值会生成类似的编码效果，且因此特定的选择只是示例。

[0112] 在步骤 820，RE(i) 的值被计算为值 $IS(P)_1, IS(P_2), ..., IS(P_t)$ 的 XOR。在步骤 825，变量 i 递增一以准备计算下一冗余码元，且在步骤 830，确定是否计算了所有冗余码元。如果没有，流程返回步骤 815。

[0113] 解码器概述

[0114] 图 11 是说明根据本发明的解码器的一实施例的简化框图。解码器 900 可以例如用于实现图 1 的解码器 155。

[0115] 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 从图 1 内的接收模块 1 50 接收输出码元 $B(I_a), B(I_b), ...$ 且从动态密钥重新发生器 160 接收动态密钥 $I_a, I_b, I_c, ...$。在接收到这些数据时，动态解码器 905 试图重建输入码元 IS(0), ..., IS(K-1) 以及冗余码元 RE(0), ..., RE(R-1)。本发明的一些实施例的优势在于动态解码器 905 不需要完成所有输入码元的解码。而是静态解码器 910 可以用于对动态解码器 905 未恢复的输入码元进行解码。

[0116] 动态解码器 905 恢复的输入码元和冗余码元被存储在重建缓冲器 915 内。在完成动态解码后，静态解码器 910 试图恢复任何动态解码器 905 未恢复的输入码元，如果有的话。尤其是，静态解码器 910 从重建缓冲器 915 接收输入码元和冗余码元。另外，静态解码器 910 从静态密钥发生器 130（图 1）（如果使用的话）接收静态密钥 $S_0, S_1, ...$。回到图 1，在一特定实施例中，静态密钥可以通过将公共种子通过通信信道 145 传递到随机数发生器 135 而经重新生成，该发生器 135 驱动静态密钥发生器 130。恢复的输入码元被提供给输入文件组装器 165。

[0117] 图 12 是根据本发明说明用于解码的方法的一实施例简化流图。在步骤 1005，Q 个输出码元由解码器接收。Q 的值可以取决于输入码元数和使用的特定动态解码器。Q 的值还可以取决于解码器能恢复输入码元的期望准确度。例如，如果期望解码器以高概率恢复所有的输入码元，则 Q 应被选为大于输入码元数。特别是在一些应用中，当输入码元数很大，Q 可以比原始输入码元数大不到 3％。在其他应用中，当输入码元数较小，则 Q 可以比输入码元数至少大 10％。尤其是 Q 可以被选为输入码元的数 K 加上数 A，其中 A 被选择保证解码器可以以较高的概率重新生成所有的输入码元。确定数字 A 在以下描述。如果对于解码器不能对所有的输入码元解码（要么有时要么一直）是可接受的，则 Q 可以小于 K+A，等于 K 或甚至小于 K。很清楚地是，整个编码系统的一个目的经常是尽可能地减少 Q，而同时以期望的准确度维持较佳概率保证解码器过程的成功。

[0118] 在步骤 1010，动态解码器 905 从 Q 个接收到的输出码元重新生成输入码元和冗余

码元。可以理解，步骤 1005 和 1010 可以大致并发地实现。例如，动态编码器 905 可以在解码器接收 Q 个输出码元前开始重新生成输入码元和冗余码元。

[0119] 在动态解码器 905 处理了 Q 个输出码元后，可以确定输入码元是否被恢复到一定的准确度。期望的准确度可以是例如所有的输入码元或小于所有输入码元的一定数目、百分比等。如果是，则流程结束。如果不是，则流程进行到步骤 1020。在步骤 1020，静态解码器 910 试图恢复任何动态解码器 905 不能恢复的输入码元。在静态编码器 910 处理了动态编码器恢复的输入码元以及冗余码元后，流程结束。

[0120] 图 13 是根据本发明用于解码的方法的另一实施例的简化流图。该实施例类似于关于图 11 的描述，且包括相同的步骤 1005、1010、1015 和 1025。但是，在步骤 1025 后，流程进行到步骤 1030，其中确定输入码元是否被恢复到期望的准确度。如果是，则流程结束。如果不是，则流程进行到步骤 1035。在步骤 1035，接收一个或多个附加输出码元。然后，流程回到步骤 1010，使得动态解码器 905 和 / 或静态解码器 910 可以试图恢复剩余的未经恢复的输入码元。

[0121] 图 14 是根据本发明用于解码方法的另一实施例的简化流图。在步骤 1055，由解码器接收输出码元，且在步骤 1060，动态解码器 905 从接收到的输出码元重新生成输入码元和冗余码元。然后在步骤 1065，确定是否应中止动态解码。该确定是基于处理的输出码元数、恢复的输入码元数、附加输入码元正在被恢复的当前速率、处理输出码元花费的时间等中的一个或多个。

[0122] 需要理解的是步骤 1055、1060 和 1065 可以大致并发地实现。例如，动态解码器 905 可以在解码器继续接收输出码元时开始重新生成输入码元和冗余码元。另外，评估是否停止动态解码过程可以在正在接收输出码元和 / 或输出码元正在被动态解码器 905 处理时周期性地实现。

[0123] 在步骤 1065，如果确定不停止动态解码，则流程回到步骤 1055。但是，如果在步骤 1065，确定结束动态解码，则流程进行到步骤 1070。在步骤 1070，确定输入码元是否被恢复到期望的准确度。如果是，则流程结束。如果不是，则流程继续进行到步骤 1075。在步骤 1075，静态解码器 910 试图恢复任何动态解码器 905 不能恢复的输入码元。在静态解码器 910 处理了由动态编码器 905 恢复的输入码元和冗余码元后，流程结束。

[0124] 动态解码器

[0125] 图 15 根据本发明示出动态解码器一实施例。动态解码器 1100 包括如图 5 内示出的动态解码器 500 类似的元件。解码器 1100 类似于 Luby I 和 Luby II 内描述的连锁解码器的实施例。动态解码器 1100 包括加权选择器 510、关联器 515，值函数选择器 520、输出码元缓冲器 1105、缩减器 1115、重建器 1120 以及重建缓冲器 1125。如同编码器，值函数选择器 520 以及在输出码元缓冲器 1105 内分配给存储值函数的描述的空间是可选的，且如果值函数对于所有的输出码元相同，则可以不使用。示出重建缓冲器 1125 的几项，一些输入码元经重建，其他仍未知，用问号表示。例如，在图 15 内，在位置 0、2、5、6 和 K-1 的输入码元以及在位置 0 和 2 的冗余码元已经被恢复了，且在位置 1、3 和 4 以及位置 1 处的冗余码元仍要被恢复。

[0126] 在操作中，对于有密钥 I 和值 B(I) 的每个接收到的输出码元，解码器 1100 进行以下操作。密钥 I 被提供给值函数选择器 520、加权选择器 510 和关联器 515。使用 K+R 以及

动态密钥 I，加权选择器 510 确定加权 W(I)。使用 K+R、动态密钥 I 以及 W(I)，关联器 515 生成与输出码元相关联的输入和冗余码元的 W(I) 个位置的列表 AL(I)。可选地，使用 K+R 和 I，值函数选择器 520 选择值函数 F(I)。然后 I、B(I)、W(I) 和 AL(I) 以及可选的 F(I) 被存储在输出码元缓冲器 1105 的行内。值函数选择器 520、加权选择器 510 以及关联器 515 如同对动态编码器 220（图 2）的描述对解码器 1105 实现相同的操作。特别是，由图 15 内的值函数选择器 520、加权选择器 510 和关联器 515 生成的值函数 F(I)、加权 W(I) 和列表 AL(I) 对于相同的动态密钥 I 如同对于图 5 内示出的对应部分相同。如果 K 和 R 随不同的输入文件而不同，则它们可以以常规的方式从编码器传递到解码器，诸如将其包括在消息头部。

[0127] 重建器 1120 扫描输出码元缓冲器 1105 以寻找存储在那里的带有加权一即 W(I) ＝ 1 以及 AL(I) 只列出一个关联的输出码元。这些码元在此被称为“可解码集合”的成员。对于带有上述特性的值函数，加权一的输出码元在可解码集合内，因为动态输入码元值可以从该输出码元被确定。当然，如果使用值函数，则会使得动态输入码元在除了带有加权一的条件下经解码，该条件会被用于确定是否输出码元在可解码集合内。为了清楚起见，在此描述的示例假设可解码集合是那些带有加权一的输出码元，且这些示例扩展到其他值函数可解码条件从该描述中应很清楚。

[0128] 当重建器 1120 找到在可解码集合内的一个输出码元时，输出码元值 B(I) 以及可选的值函数 F(I) 用于重建在 AL(I) 内列出的动态输入码元，且重建后的动态输入码元放入重建缓冲器 1125 对该输入或冗余码元合适的位置。如果指示的输入或冗余码元已经被重建，则重建器 1120 可丢弃新重建的动态输入码元，覆盖现存的重建后输入或冗余码元，或者如果两者不同，比较两者并生成差错。在值函数是所有关联的 XOR 情况下，输入或冗余码元值简单地是输出码元值。重建器 1120 因此只从可解码集合内的输出码元重建输入和冗余码元。一旦来自可解码集合的输出码元被用于重建输入或冗余码元，则它可以被删除以节省在输出码元缓冲器 1105 内的空间。删除“使用过”的输出码元还保证了重建器 1120 不会连续地重新访问该输出码元。

[0129] 开始时，重建器 1120 等待直到至少接收了一个输出码元，该输出码元是可解码集合的成员。一旦使用了这一个输出码元，可解码集合会再次变空，除了一些其他输出码元可能只是这一个重建后的输入或冗余码元和一个其他的输入或冗余码元的函数。因此，从可解码集合的成员重建一个输入或冗余码元会引起其他输出码元被加入可解码集合。输出码元减少以将其加入可解码集合的过程由缩减器 1115 实现。

[0130] 缩减器 1115 扫描输出码元缓冲器 1105 和重建缓冲器 1125 以找到一些输出码元，这些码元的列表 AL(I) 列出已经被恢复的输入或冗余码元的位置。当缩减器 1115 找到该种带有密钥 I 的“可缩减”的输出码元，缩减器 1115 获得在位置 h 处的恢复后的动态输入码元的值 IS(h) 并修改 B(I)、W(I) 以及 AL(I)，如下：

[0131] B(I) 被设定为$B(I) \oplus IS(h)$

[0132] W(I) 被设定为 W(I)-1

[0133] AL(I) 被设定为除了 h 以外的 AL(I)

[0134] 在上述等式中，假设值函数是所有关联的值的 XOR。值得注意的是 XOR 是其本身的反－如果情况不是这样，且原来使用另一值函数以计算输出码元，则该值函数的反在此被缩减器 1115 使用。如应很明显的。如果已知多余一个关联的值，则以上等式的等价可以经

计算以使得 B(I) 只取决于任何未知的关联值（并相应地调整 W(I) 和 L(I)）。

[0135] 缩减器 1115 的行为减少了输出码元缓冲器 1105 内的输出码元的加权。当输出码元的加权被减少到一（或对其他值函数发生其他可解码条件），则该输出码元成为可解码集合的成员，可以由重建器 1120 对其进行操作。实际上，一旦接收到充分数量的输出码元，缩减器 1115 和重建器 1120 建立连锁解码，重建器 1120 对可解码集合解码以恢复更多的动态输入码元，缩减器 1115 使用这些刚恢复的输入或冗余码元以减少更多的输出码元，使得它们被加入可解码集合等，直到可解码集合为空。

[0136] 图 15 内示出的解码器部分以直接方式部分地重建输入和冗余码元，而不考虑存储器存储、计算周期或传输时间。当解码器存储、解码时间或传输时间（这限制了接收到的输出码元数）受限时，解码器可以经优化以更好地使用这些受限的资源。该种优化的示例在例如 Luby I 和 Luby II 内经描述。这些优化可以被用于多级编码的动态解码。另外，可以理解可以使用其他的变体和等价的解码器。

[0137] 静态解码器

[0138] 图 16 是说明静态解码器的一实施例的简化框图。该实施例可以在数据用诸如图 7 内描述的静态编码器编码时被使用。静态解码器 1200 包括 LDPC 解码器 1205 以及汉明解码器 1210。LDPC 解码器 1205 从重建缓冲器 1215 接收输入码元以及冗余码元，并试图重建这些在动态解码器的解码步骤之后未经恢复的重建缓冲器 1215 的码元。在一些实施例中，重建缓冲器 1215 是重建缓冲器 1125（图 15）。LDPC 解码器 1205 接收由静态密钥发生器 130 生成的静态密钥 S_0，S_1，...。另外，LDPC 解码器 1205 接收 K 个输入码元，D 个冗余汉明码元以及 E 个冗余 LDPC 码元。LDPC 解码器 1205 以本领域内技术人员已知的方式尽可能多地恢复输入和冗余码元，并将这些值写入重建缓冲器 1215 内对应的位置。

[0139] 汉明解码器 1210 还经耦合以从重建缓冲器 1215 接收输入码元和冗余码元。另外，汉明解码器 1210 接收输入码元数 K、数字 D，其中 D+1 是冗余汉明码元数。汉明解码器 1210 试图恢复这些未被动态解码器和 LDPC 解码器 2005 恢复的输入码元。虽然 LDPC 解码器 2005 的目的是恢复尽可能多的输入和冗余码元，汉明解码器 2010 只试图恢复输入码元 IS(0)，IS(1)，...，IS(K-1)。

[0140] LDPC 解码器和汉明解码器的许多变体对于本领域内的技术人员是熟知的，且可以用于本发明的各个实施例中。在一特定实施例中，汉明解码器使用高斯消去 (elimination) 算法实现。高斯消去算法的许多变体在本领域内已知，且可以根据本发明用于各个实施例。

[0141] 在一定应用中，更优地是使用图 1 内示出的不同于上述一个类型的解码器 155。例如，如果输入码元数 K 不很大，例如小于 1000，则输出码元的接收概率过程内涉及的方差可以强迫解码器 155 收集多个输出码元，这些输出码元远显著大于 K 为了能使动态和静态解码器纠正规定数目的擦除。在这些情况中，可以使用不同类型的解码器。该种使用高斯消去的解码器的实施例在以下参考图 17、18 和 19 进行描述。

[0142] 首先，回到图 1，解码器 155 从接收模块 150 接收输出码元 $B(I_a)$，$B(I_b)$，...，从动态密钥重新发生器 160 接收密钥 I_a，I_b，...，且从静态密钥发生器 130 接收密钥 S_0，S_1，...。另外，它接收输入码元的值 K 以及冗余码元的值 R。在接收到该输入后，它试图重建输入码元 IS(0)，...，IS(K-1)，这些码元被传递到输入文件组装器 165 以作进一步处理。

[0143] 现在参考图 17，解码器 1300 包括动态矩阵发生器 1305 和静态矩阵发生器 1310。

动态矩阵发生器 1305 接收输出码元 $B(I_a)$，$B(I_b)$，...、动态密钥 I_a，I_b，... 以及参数 K 和 R。另外，动态矩阵发生器 1305 接收其他参数 A，这些参数描述应收集多少输出码元（即收集的输出码元数为 K+A）。参数 A 的确定一般取决于用于动态和静态编码的方法，且在以下会经详述。在以下的描述中，收集的 K+A 个输出码元被称为 B(0)，B(1)，...，B(K+A-1)。在接收到这些参数后，具有形式为 C* 转置的线形方程组 (IS(0)，...，IS(K-1)，RE(0)，...，RE(R-1)) ＝转置 (B(0)，...，B(K+A-1))，该系统由动态矩阵发生器 1305 设定，其中 C 是形式为 (K+A)x(K+R) 的矩阵。由动态矩阵发生器 1305 的生成矩阵 C 在以下将详述。

[0144] 然后静态矩阵发生器 1310 从动态矩阵发生器 1305 接收矩阵 C，并使用密钥 S_0，S_1，... 以向矩阵 C 加入 R 行以获得方程组：

[0145] M* 转置 (IS(0)，...，IS(K-1)，RE(0)，...，RE(R-1)) ＝

[0146] 转置 (B(0)，...，B(K+A-1)，0，...0)

[0147] 其中右边向量的最后 R 项为零，且其中 M 为 (K+A+R)x(K+R) 格式。最终，使用线性方程组求解 1315 以求解该方程组 M 并获得输入码元 IS(0)，...，IS(K-1) 的一些或所有。在一特定实施例中，线性方程组求解器 1315 使用高斯消去算法以求解线性方程组。

[0148] 动态矩阵发生器 1305 和静态矩阵发生器 1310 现在参考图 5 的动态编码器 500 和图 2 内的静态编码器 205 进一步详细描述。图 18 是说明动态矩阵发生器 1305 使用的方法的实施例。在步骤 1405，动态矩阵发生器 1205 初始化形式为 (K+A)x(K+R) 的矩阵 C 全为零。下一步，在步骤 1410，密钥 I_a，I_b，... 连同加权选择器 510 和关联器 515 一起用于生成加权 W(0)，...，W(K+A-1) 以及相应的列表 AL(0)，...，AL(K+A-1)。每个列表 AL(k) 包括在范围 0，...，K+R-1 内的 W(k) 个整数。在步骤 1415 内，这些整数用于用 AL(k) ＝ (a(0)，...，a(W(k)-1) 计算 C(k，1)，且项 C(k，a(0))，...，C(k，a(W(k)-1)) 被设定为 1。如上所述，矩阵 C 生成未知 (IS(0)，...，IS(K-1)，RE(0)，...，RE(R-1)) 关于 (B(0)，...，B(K+A-1)) 的方程组。原因如下：一旦动态编码器选择加权 W(k) 以及关联列表 AL(k) ＝ (a(0)，...，a(W(k)-1)，则对应的输出码元 B(k) 可以获得为：

[0149] $$B(k) = L(a(0)) \oplus L(a(1)) \oplus \ldots \oplus L(a(W(k)-1)),$$

[0150] 其中 L(j) 表示在位置 j 处的重建缓冲器 1925 的未知值。这些方程为 0 到 K+A-1 间的所有 k 值累加，形成期望的方程组。

[0151] 图 19 是说明静态矩阵发生器 1310 使用的方法的一实施例简化流图。该实施例参考图 1 0 说明。在图 10 的步骤 820 内，值得注意的是冗余码元 RE(i) 作为 $RE(i)=IS(P_1)\oplus\ldots\oplus IS(P_t)$ 经计算，且 P_1，P_2，...，P_t 如在步骤 815 内在接收到密钥 S_i 后经计算。这意味着 $IS(P_1)\oplus\ldots\oplus IS(P_t)\oplus RE(i)=0$。用重建缓冲器的位置说明，这意味着 $L(P_1)\oplus\ldots\oplus L(P_t)\oplus L(i+k)=0$。将 M 的 (i，$P_1$)，...，(i，$P_t$)，(i，i-A) 项设定为 1，其中 i 从 K+A 到 K+A+R-1，获得矩阵 M，该矩阵描述未知的 (IS(0)，...，IS(K-1)，RE(0)，...，RE(R-1)) 关于 (B(0)，...，B(K+A-1)) 的线性方程组，如上所述。

[0152] 在步骤 1505，格式为 (K+A+R)x(K+R) 的矩阵 M 通过使得 M 的前 K+A 行等于动态矩阵发生器 1305 计算的矩阵 C 而经初始化。M 的剩余行经初始化为零。下一步在步骤 1510，变量 i 经初始化为 K+A。该变量跟踪 M 的最后 R 行。在步骤 1512，计算与冗余码元 i-K-A 的相关联的数 t。该步骤类似于图 8 的步骤 810。特别是如果在图 8 给出的静态编码过程中偏好另一 t 选择，则还可以为步骤 1512 内计算的变量 t 采取该选择。在步骤 1515，关联

器 414 从静态密钥 S_i、输入码元数 K 以及整数 t 计算在 0 到 K-1 之间的索引 $P_1, P_2, \ldots, P_t$。然而，矩阵 M 的对应位置在步骤 1530 内被设定为 1。步骤 1540 内的递增和步骤 1550 内的测试保证了 M 的所有最后 R 行被访问且经计算。

[0153] 在一些实施例中，图 17、18 和 19 内示出的实施例可以比在此描述的其他实施例更优，因为比起其他实施例，它允许相对收集更少的输出码元以进行正确解码。解码器的选择很大程度上取决于应用，且取决于例如收集的输出码元数是否是关键资源。

[0154] 关联器实现

[0155] 回到图 5，关联器 515 的一实施例在 Luby I 内描述。数 N 应是质数。在操作中，当该实施例用于计算 AL(I) 时，输入大小 K+R 可以经调整使得它是质数。在本发明中，冗余码元数目被选得充分大使得 K+R 为质数。在一些应用中，输入 N 是质数的条件非常有限制性。

[0156] 另一用于实现关联器 520 的方法的实施例在图 20 内示出，其中 N 不需要为质数。首先在步骤 1805，变量 k 被初始化为零。然后，在步骤 1810 处，生成随机整数 Y。在一特定实施例中，输出码元的密钥被用作随机数发生器的种子。然后，在步骤 1815，对整数 Y 与进行模数 N 运算以生成 0 到 N-1 间的数。在步骤 1820，候选数 Y 与其它先前生成的 Y 个 (X(0), X(1),...) 相比经测试。如果 Y 已经先前被生成，则流程回到步骤 1810。否则，在步骤 1825，它被包括在列表 X(0), X(1), ... 内。则在步骤 1830，确定是否生成了 W(I) 个数。如果没有，则流程回到步骤 1810。图 8 内说明的流程结果是 W(I) 个数 X(0), X(1), ..., X(W(I)-1) 的列表，其中列表内的每个数 X 是 0 到 N-1 间的唯一整数。然后，在步骤 835 内，列表 AL(I) 被设定为数 X(0), X(1), ..., X(W(I)-1)。

[0157] 加权选择器实现

[0158] 编码器 / 解码器的性能的效率取决于图 2 内示出的动态编码器 220 生成的输出码元的加权分布，且一些分布优于其他分布。尤其是参数 A 的选择主要受到加权分布选择的影响，该参数 A 描述与输入码元数 K 相比收集到数据码元数超出情况。加权选择的操作方面在以下描述，接着是一些重要的加权分布的描述。图 21 的框图和图 22 的流图用于说明这些概念。

[0159] 图 5 内示出的加权选择器 510 的任务如下 :在接收到密钥 I 和长度 K+R 后，加权选择器输出在范围 0 到 K+R-1 内的整数 W(I)，该整数被称为加权。不同于关联器 515，加权选择器 510 的输出希望不是均匀，而是偏斜的，更集中于一些加权，而关联器则理想地用均匀随机分布均匀地生成整数。

[0160] 如图 21 示出，加权选择器 510 包括两个处理过程 WT_INIT 1905 和 WT_CALC1910，以及两个表格 WT_RBITS 1915 和 WT_DISTRIB 1920。处理过程 WT_INIT 1905 只在当第一密钥被传递以初始化表格 WT_DISTRIB 1920 时被调用。WT_DISTRIB1920 的设计是系统的重要方面，且之后将更详细描述。过程 WT_CALC 1910 在每次被调用时被启用以基于密钥 I 生成加权 W(I)。如在图 22 的流程图内示出的，WT_CALC 1910 使用存储在表格 WT_RBITS 1915 内的密钥和随机比特以生成随机数 T(2005)。然后，T 的值被用于在表格 WT_DISTRIB 1920 内选择行号 N。

[0161] 如图 21 示出，WT_RBITS 1920 内的 RANGE 列内的项是递增的正整数序列，结束于值 MAX_VAL，且 WT 列是递增的正整数序列，结束于值 MAX_WT。T 的可能值集合是在零到 MAX_VAL-1 间的整数。期望的特性是 T 等概率地可能是可能值范围内的任何值。N 的值是通过搜

索RANGE列直到找到一N而确定的，该N满足RANGE(N-1) ≤T＜RANGE(N)(2010)。一旦找到N，值W(I)被设定为WT(N)即表格WT_DISTRIB的WT列的第N项，且这是返回的加权(2015，2020)。在图21内，对于示出的示例表格，如果T等于38500，则N被确定为4，且因此W(I)被设定为WT(4) ＝8。在最优实施例中，WT_DIST 1920的行经组织使得RANGE(N)-RANGE(N-1)随N递增时值递减。这最小化了通过WT_DIST 1920的平均搜索时间，即当使用从第一行开始的顺序搜索时对应值T的加权的时间。在其他实施例中，其他行的组织方式可能是更优的，且可以使用其他的搜索方法，诸如二分搜索。

[0162] 选择加权分布

[0163] 应为给定的编码过程选择加权分布，使得输入文件可以完整地被重建，使用a)尽可能少的输出码元，b)尽可能少的操作，以及c)尽可能高的可靠性。一般，这些最优准则可以通过准确地为输出码元选择正确的加权分布（即在所有I上的W(I)的分布）以及在输出码元上的关联的分布（即在所有I上的AL(I)的成员）而达到。要强调的是虽然可以不管加权重分布和关联选择上的分布而应用解码过程，但最优实施例会使用为接近最佳性能而选择的加权分布和关联选择上的分布。实际上，许多分布性能很好，因为选择的分布内的较小变化只会导致性能内较小的变化。

[0164] 现在描述一种用于在最优实施例中确定分布的方法。使用的实际加权分布取决于输入码元的数目K。分布给出如下，连同范围(K_{min}，K_{max})、因子β以及相对开销α。这有以下意义：给定K，其中K_{min} ≤K≤K_{max}，则冗余码元数目R是用大于或等于β*K的最小整数计算的，收集的输出码元数应至少为(1+α)*K，即上述的参数A是大于或等于α*K的最小整数。在使用第一个关联器520的版本情况下，R应附加地满足K+R是质数的条件，即R是大于等于β*K的最小质数。如果应用不需要K+R为质数，则R可以被选择为大于或等于β*K的最小整数。

[0165] 分布本身以以下形式的表格给出

[0166]

加权 1	P1
加权 2	P2
加权 3	P3
…	…

[0167] 其中P1是加权1的对应概率，P2是加权2的对应概率等，且其中P1、P2，... 的和为一。这意味着图21的表格WT_DISTRIB 1920有以下形式

[0168]

加权 1	MAX_VAL*P1
加权 2	MAX_VAL*(P1+P2)
加权 3	MAX_VAL*(P 1+P2+P3)

…	…

[0169] 将描述用于计算在此的表格的一般规则。设计的一个目标是使得目前的具有减少加权码元的一个非零数的输出码元尽可能地在进入动态解码过程。最好，该数目能在到动态解码结束前的整个过程中一直大于零。然而，数学分析示出这只在如果输出码元的平均加权至少与输入码元数 K 的对数成正比时才可能，且这是 Luby I 内描述的几种加权分布的设计。本发明的一些实施例将该平均加权大大减少到固定的独立于 K 的恒量。结果，减少加权码元的输出码元数不能期望在整个动态解码过程期间大于零。

[0170] 设计加权分布的初始步骤是获得期望的动态输入码元的数目表达式，其值在动态输入码元的部分 x 尚未被恢复时，可以从在动态解码过程中当前可解码集合内的输出码元获得。该表达式是加权 1,2,...,k 的输出码元的部分 P1,P2,..,Pk 以及收集的输出码元数和输入和冗余码元数之比的函数。以下，该比用 γ 表示。可见 $\gamma = (1+\alpha)/(1+\beta)$。该量的数学分析示出该种动态输入码元的期望数目可以表示为

[0171] $K*(1+\beta)*(x-e^{-(1+\gamma)*\omega(1-x)})$, (1)

[0172] 其中 x 表示在动态解码过程中尚未被恢复的动态输入码元部分，且 $\varpi(x)$ 为多项式：

[0173] $P1+2*P2*x+3*P3*x^2+...+k*Pk*x^{k-1}$. (2)

[0174] 由于该数目只是一个量的期望，这个量是统计性质的，所以会有变化。该变化的分析显示它与还未恢复的输入码元的期望数目即与 $x*K*(1+\beta)$ 的方根成正比。为了合理地保证是与减少加权码元的输出码元的相邻的输入码元数总为正，则 P1,...,Pk 和 γ 的选择应使得：

[0175] $K*(1+\beta)(x-e^{-(1+\gamma)*\omega(1-x)}) > c*sqrt(x*K*(1+\beta))$, (2)

[0176] 其中该不等性对于在给定正实数 ε 和 1 间的所有 x 值均成立，且 c 是大于一的正实数。c 越大，越能保证解码过程的成功。ε 越小，则在动态解码过程最后会有更少的未经恢复输入码元。$\varpi(1)$ 越小，输出码元的平均加权越小。给定这些限制，可以为给定的 ε 和给定的 c 计算多项式 $\varpi(x)$，其中所有的系数非负，这满足了 ε 和 1 之间的所有 x 值的上述不等性，且对此 $\varpi(1)$ 尽可能地小。该最优化可以由各种方法完成，例如在适当地操作上述不等性后使用单纯形算法。

[0177] 现在实际表格根据以上描述被给出。以上描述内的恒量 c 的选择是为了保证总解码器擦除概率小于 10^{-10}。对于 $K > 49251$，差错概率小于 10^{-12}。这些表格仅作为可以被使用的加权分布示例而被提供。可以理解还可以使用其他加权分布。

[0178] 表格 1

[0179] K 范围：9900-14800，$\beta = 0.0081$，$\alpha = 0.1187$

[0180]

1	0.018235
2	0.477562
3	0.153565

4	0.102006
5	0.034651
7	0.048352
8	0.06084
18	0.058325
19	0.008401
70	0.008451
71	0.029613

[0181] 表格 2

[0182] K 范围 :14801-19680，β ＝ 0.00121，α ＝ 0.084

[0183]

1	0.019314
2	0.483582
3	0.160754
4	0.081631
5	0.067541
8	0.094528
18	0.041968
19	0.019462
66	0.007987
67	0.023233

[0184] 表格 3

[0185] K 范围 :19681-29510，β ＝ 0.0151，α ＝ 0.0769

[0186]

1	0.013531
2	0.488250
3	0.164810
4	0.070953
5	0.084243
8	0.050093
9	0.042547
19	0.055060
62	0.00501988
63	0.025491

[0187] 表格 4

[0188] K 范围 :29511-49250，β ＝ 0.0161，α ＝ 0.0674

[0189]

1	0.013876
2	0.489087
3	0.162276
4	0.081638
5	0.069880
8	0.081339
9	0.014424
18	0.017712
19	0.040774
66	0.014680
67	0.014314

[0190] 表格 5

[0191] K 范围 :49251-64780，β ＝ 0.015，α ＝ 0.0558

[0192]

1	0.009117
2	0.492843
3	0.165983
4	0.072707
5	0.082303
8	0.056347
9	0.036917
19	0.055616
65	0.022195
66	0.005972

[0193] 表格 6

[0194] K 范围 :64781-79080，β ＝ 0.0114，α ＝ 0.05

[0195]

1	0.007969
2	0.493570
3	0.166220
4	0.072646
5	0.082558
8	0.056058
9	0.037229
19	0.055590
65	0.025023
66	0.003135

[0196] 表格 7

[0197] K 范围 :79081-98623，β ＝ 0.01134，α ＝ 0.047

[0198]

1	0.007544
2	0.49361
3	0.166458
4	0.071243
5	0.084913
8	0.049633
9	0.043365
19	0.045231
20	0.010157
66	0.010479
67	0.017365

[0199] 表格 8

[0200] K 范围 :98624-118349，β ＝ 0.01377，α ＝ 0.0424

[0201]

1	0.006495
2	0.495044
3	0.168010
4	0.067900
5	0.089209
8	0.041731
9	0.050162
19	0.038837
20	0.015537
66	0.016298

67	0.010777

[0202] 表格 9

[0203] K 范围 :118350- ∞, β = 0.01579, α = 0.0393

[0204]

1	0.004807
2	0.496472
3	0.166912
4	0.073374
5	0.082206
8	0.057471
9	0.035951
18	0.001167
19	0.054305
65	0.018235
66	0.009100

[0205] 例如,如果 K = 33000,则冗余码元的数目可以是大于 K*0.0161 = 531.3 的最小整数 R,使得 K+R 为质数。即 R = 533。收集的输出码元数应至少为 (1+0.0674)*K 即 35225。

[0206] 表格 1 的平均加权大致为 6.75,且表格 2 到 9 的平均加权大致为 6。这些平均加权与先前描述的 Luby I 的一实施例相比大大减少,先前的情况在 K 等于 60000 时平均加权为 28.68。

[0207] 较少输入码元的编码

[0208] 对于任何大小的输入文件最好是相对较低的开销。如以上表格中可见,随着输入码元数 K 变得较小,相对开销 α 增加。例如,如果输入文件有 10000 字节,且 K 被选为 10000 使得每个码元包括一个字节,则开销大致为 11%,这对于一些应用是不期望的。一种减少开销的方法是增加输入码元数 K,其代价是减少输入码元大小。例如,在期望大致 7%的开销时,K 被选为 40000。在该情况下,输入码元的大小会是 2 比特。然而使用非常小大小的输入码元实际上会导致计算不足。

[0209] 该问题的解决方法是使用基于高斯消去法的解码器,诸如图 17 内描述的实施例。即使与其他实施例内描述的任何静态编码器组合的连锁解码器相比,该解码器计算效率没有那么有效,与解码器的非常小的失败概率和开销的组合使得该解决方法在一些应用内是期望的。

[0210] 尤其是，当输入码元数 K 在 800 到 9899 之间时，以下的加权分布可以用于动态编码器 220：

[0211] 表格 10

[0212] K 范围：800-1600，β ＝ 0.08，α ＝ 0.05

[0213]

2	0.39
3	0.095
4	0.095
5	0.095
10	0.095
19	0.095
30	0.095
130	0.04

[0214] 根据诸如图 10 内描述的静态编码器的操作生成 0.08*K 冗余码元。解码使用诸如图 17 内描述的解码器完成。

[0215] 作为一示例，假设输入文件大小为 16000 字节。输入码元的选择使得它们每个包括 16 个字节，使得输入码元的数目 K 为 1000。图 10 内的静态编码器用于建立 80 个冗余码元。下一步，动态编码器 220 与上述的加权分布一起用于生成输出码元。接收机收集 (1+α)*K ＝ 1050 个输出码元并将其提供给图 17 的解码器。动态矩阵发生器 1305 建立格式为 1050x1080 的矩阵 C。静态矩阵发生器 1330 建立格式 1130x1080 的矩阵 M，并将其送到线性方程组求解器 1340 的系统，该系统试图对原始 1000 个输入码元（即输入文件）解码。

[0216] 一些多级编码的一些特性

[0217] 如上所述的大多数的示例，输入和输出码元对相同数量的比特进行编码，且每个输出码元被放入一个分组（分组是或者整个被接收到或者整个被丢失的传输单元）。在一些实施例中，通信系统经修改使得每个分组包含几个输出码元。输出码元值大小被设定到基于多个因子将文件初始分成输入码元时的输入码元值大小所确定的大小。解码过程保持基本不变，除了输出码元在接收到每个分组时成串到达。

[0218] 输入码元和输出码元大小的设置一般由文件的大小和输出码元要在其上发送的通信系统决定。例如，如果通信系统将数据比特分组成所定义的大小的分组或以其他方式分组比特，则码元大小的设计开始于该分组进行分或组的大小。此后，设计者可以确定在一个分组或一组内可以携带多少输出码元，并确定输出码元大小。为了简洁，设计者可能会使得输入码元大小设置为等于输出码元大小，且如果输入数据使得不同的输入码元大小更方

便，则也可以使用不同大小。

[0219] 上述的编码过程基于原始文件生成包含输出码元的分组流。流内的每个输出码元独立于所有其他的输出码元经生成，且在可以建立的输出码元数上没有上下界。一密钥与每个输出码元关联。该密钥以及输入文件的一些内容确定输出码元值。接连生成的输出码元不需要连续的密钥，且在一些应用中，最好能随即生成密钥序列或伪随机地生成该序列。

[0220] 多级解码的特性是如果原始文件可以被分成 K 个等大小的输入码元，且每个输出码元值长度与输入码元值的长度相同，则文件可以从平均 K+A 个输出码元中以很高的概率恢复，其中 A 与 K 相比较小。例如，对于上述的加权分布，如果 K 大于 19681，则 A 值超过 $\alpha * K$ 的概率最多为 10^{-12}，对于 K 的任何值，最多为 10^{-10}。由于特定的输出码元以随机或伪随机顺序生成，且传输中的特定输出码元的丢失被假设为随机，则在恢复输入文件需要的输出码元的实际数量上存在某些小方差。在一些情况下，其中特定的 K+A 分组的集合不足以对整个输入文件解码，如果接收机可以从一个或多个输出分组源收集到更多的分组，则输入文件仍可以被恢复。

[0221] 由于输出码元数只受到 I 的分辨率的限制，则可以生成大大多于 K+A 个输出码元。例如，如果 I 为 32 比特数，则可以生成四十亿不同的输出码元，然而文件仅可以包括 K = 50000 个输入码元。在一些应用中，这些四十亿个输出码元只有一小部分被生成且被发送，且几乎肯定输入文件可以用很小部分的可能输出码元以及极佳的概率被恢复，输入文件可以用稍微多于 K 个输出码元被恢复（假设输入码元大小与输出码元大小相同）。

[0222] 在一些应用中，可能可以接受不能对所有的输入码元解码，或以相对较低的概率对所有输入码元解码。在该种应用中，接收机可以在接收到 K+A 个输出码元后停止试图对所有的输入码元解码。或接收机可以在接收到少于 K+A 个输出码元后停止接收输出码元。在一些应用中，接收机可能只能接收到 K 个或更少的输出码元。因此，可以理解在本发明的一些实施例中，期望的准确度可能不是所有输入码元的完整恢复。

[0223] 而且，在一些不完整恢复可接受的应用中，数据可以经编码使得所有的输入码元都不能被恢复，或使得完整的输入码元需要接收到比输入码元数更多的输出码元。该种编码一般要求的计算代价较少，且因此是减少编码计算代价的可接受方法。

[0224] 可以理解的是，上述图内的各种功能框可以用硬件和 / 或软件的组合实现，且在特定实现中，可以组合一些框的一些或所有功能。类似地，可以理解，这是描述的各个方法可以用硬件和 / 或软件的组合而实现。

[0225] 以上描述是说明性的，而不是限制型的。本发明的许多变体对于领域内的技术人员在阅读了本揭示后会变得明显。本发明的范围因此不是由描述确定，而是由以下所附的权利要求书以及其等价的完整范围确定。

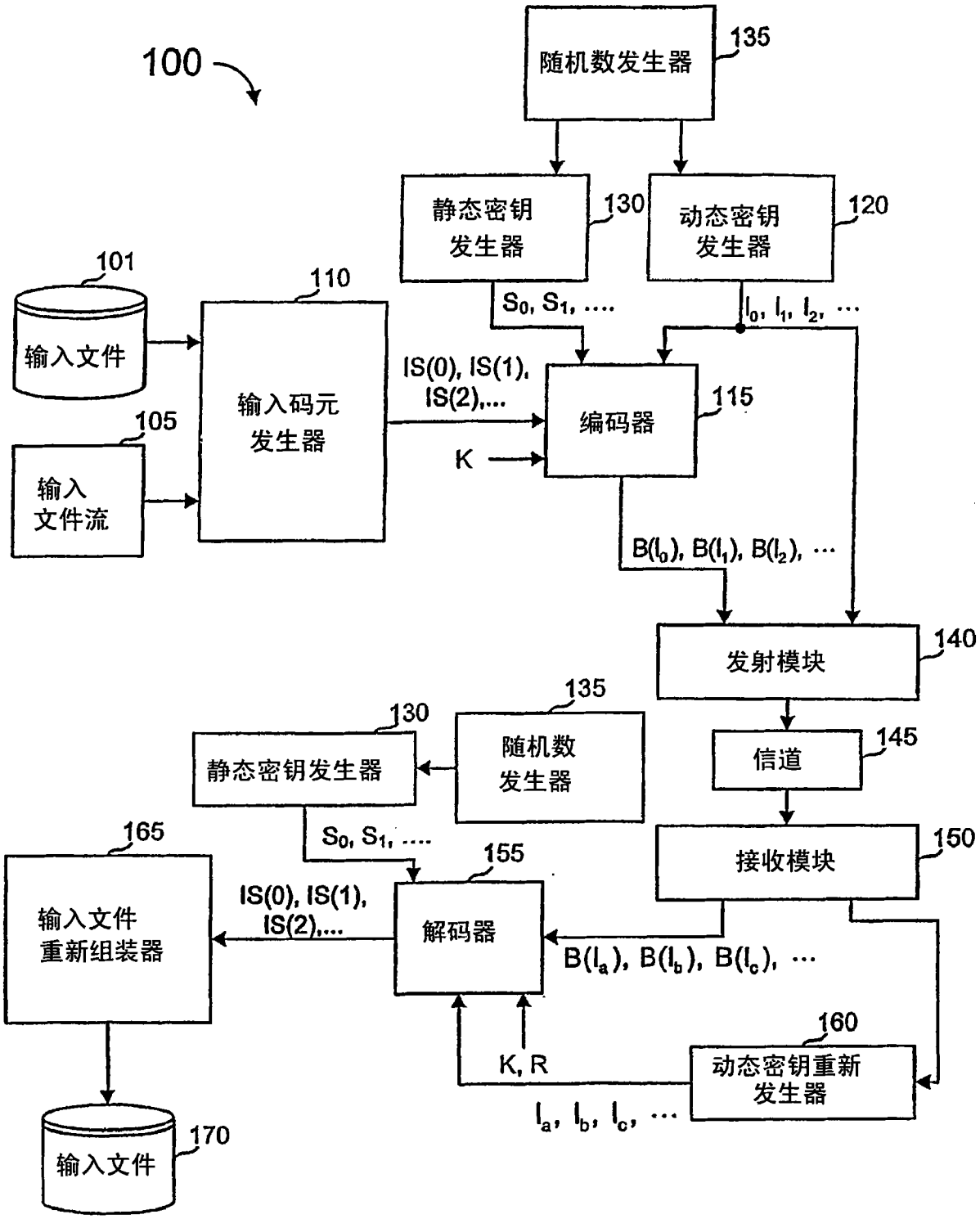

100
随机数发生器
135
静态密钥发生器
130
动态密钥发生器
120
101
输入文件
110
输入码元发生器
105
输入文件流
S0, S1, ...
I0, I1, I2, ...
IS(0), IS(1), IS(2),...
K
编码器
115
B(I0), B(I1), B(I2), ...
发射模块
140
135
随机数发生器
130
静态密钥发生器
信道
145
S0, S1, ...
165
输入文件重新组装器
155
解码器
IS(0), IS(1), IS(2),...
接收模块
150
B(Ia), B(Ib), B(Ic), ...
160
K, R
动态密钥重新发生器
Ia, Ib, Ic, ...
170
输入文件

图 1

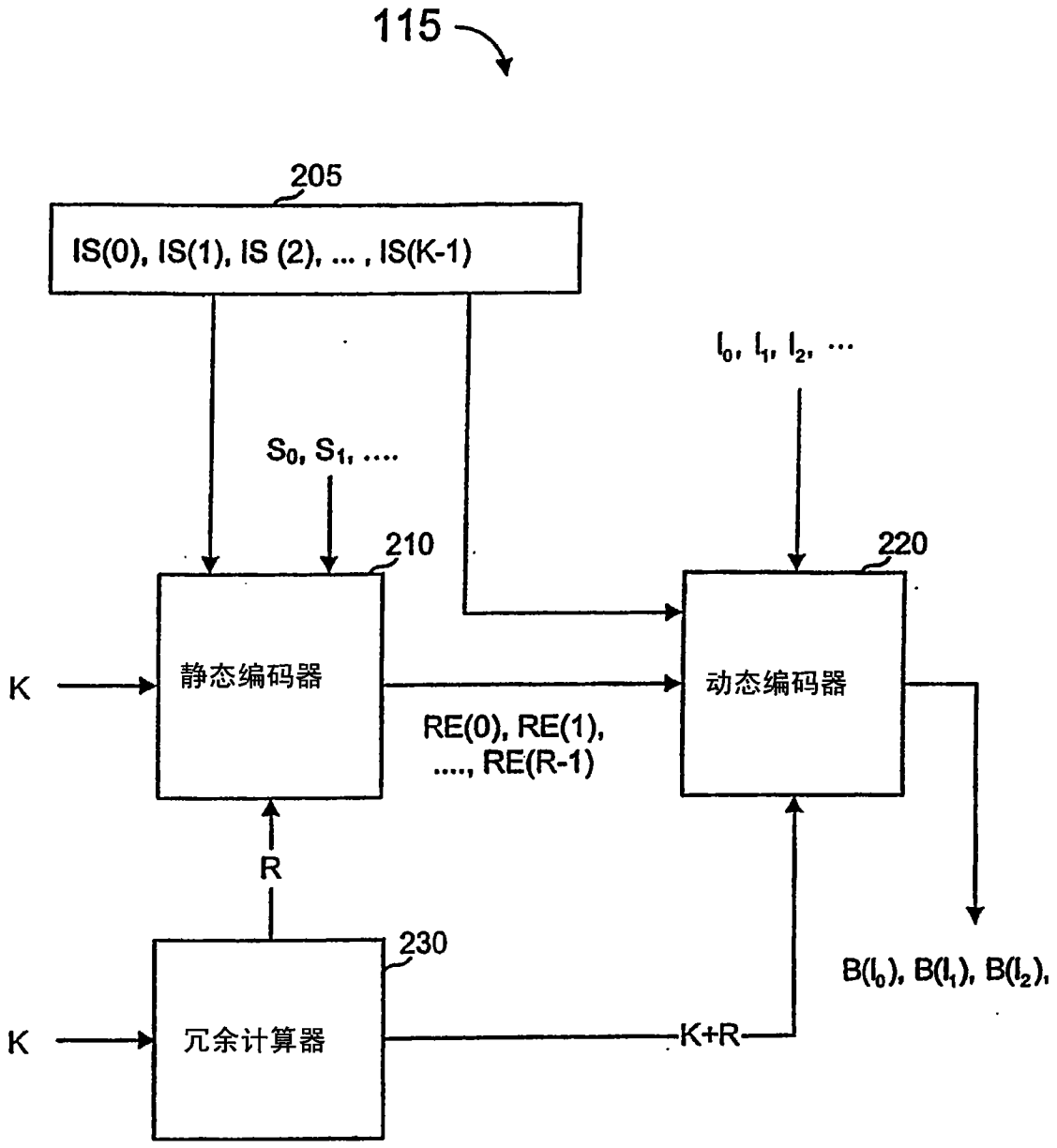

115
205
IS(0), IS(1), IS (2), ... , IS(K-1)
I_0, I_1, I_2, ...
S_0, S_1, ...
210
220
K
静态编码器
动态编码器
RE(0), RE(1),
...., RE(R-1)
R
230
$B(I_0)$, $B(I_1)$, $B(I_2)$,
K
冗余计算器
K+R

图 2

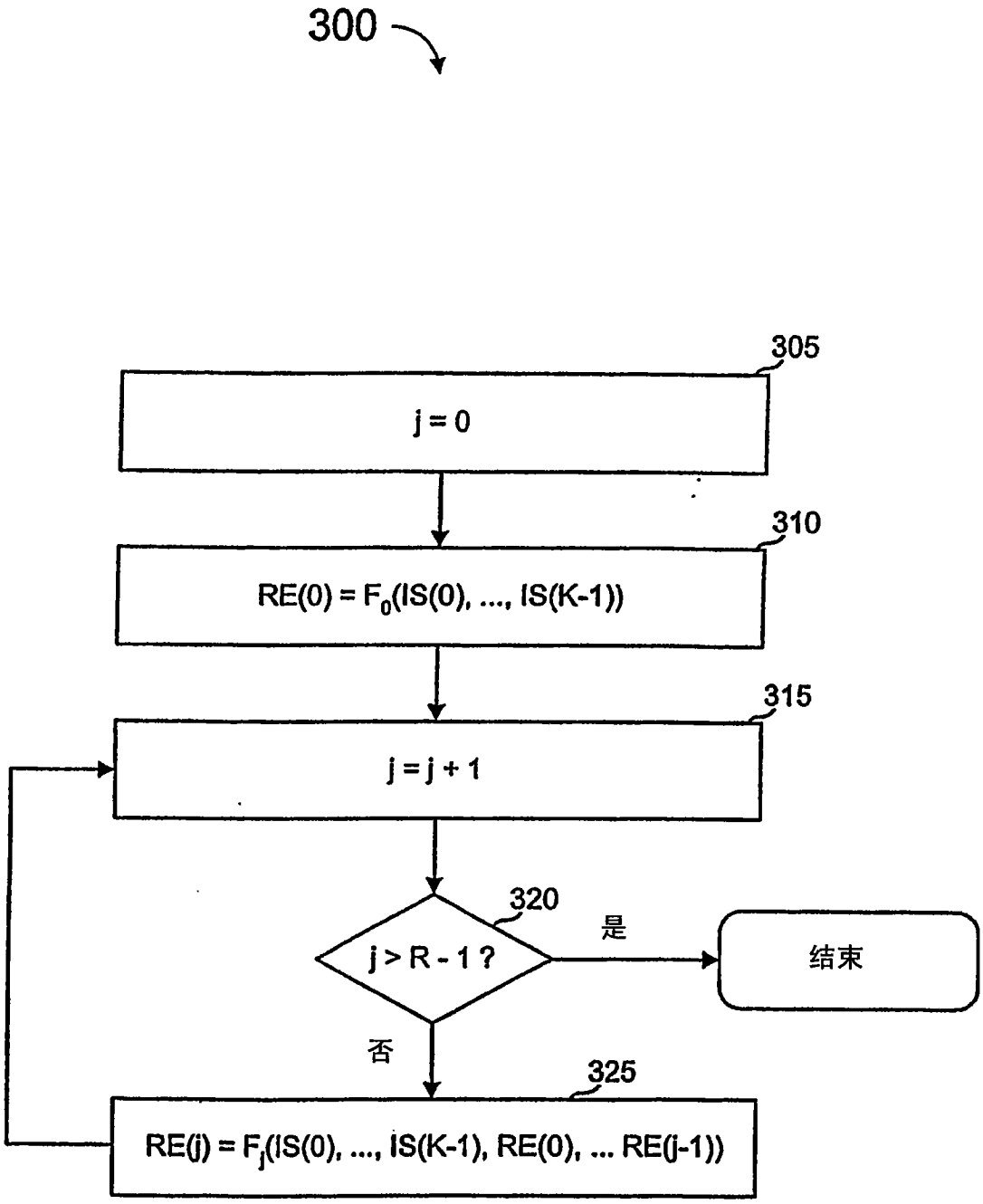

300
305
j = 0
310
RE(0) = F0(IS(0), ..., IS(K-1))
315
j = j + 1
320
j > R - 1 ?
是
结束
否
325
RE(j) = Fj(IS(0), ..., IS(K-1), RE(0), ... RE(j-1))

图 3

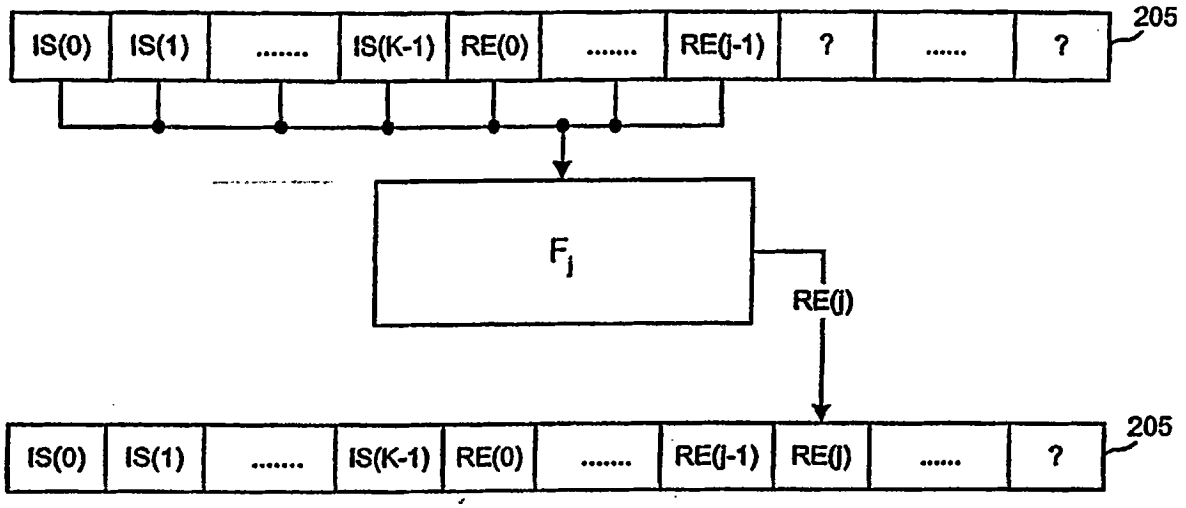

IS(0)
IS(1)
.....
IS(K-1)
RE(0)
.....
RE(j-1)
?
.....
?
205
F_j
RE(j)
IS(0)
IS(1)
.....
IS(K-1)
RE(0)
.....
RE(j-1)
RE(j)
.....
?
205

图 4

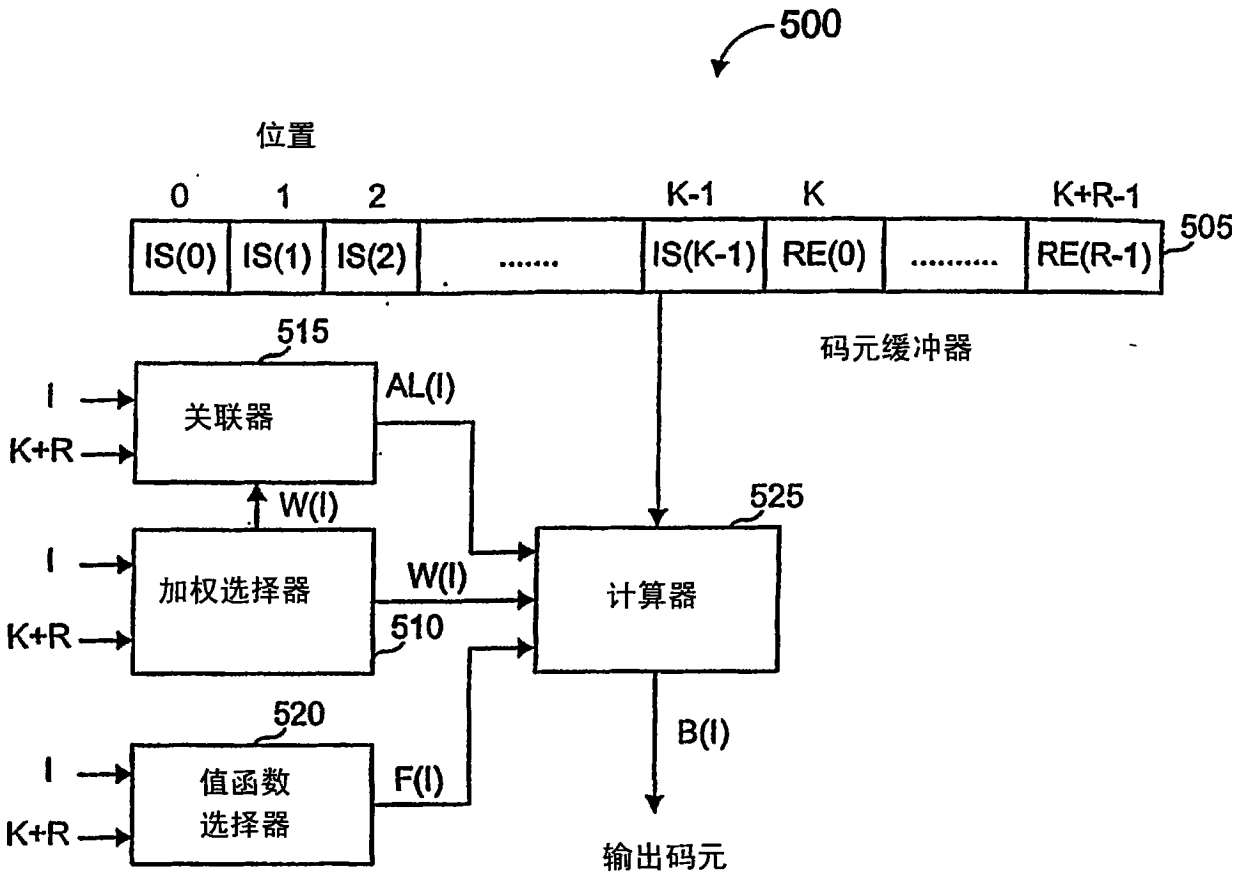

500
位置
0
1
2
K-1
K
K+R-1
IS(0)
IS(1)
IS(2)
.....
IS(K-1)
RE(0)
.....
RE(R-1)
505
码元缓冲器
515
关联器
I
K+R
AL(I)
W(I)
525
加权选择器
I
K+R
W(I)
510
计算器
520
值函数
选择器
I
K+R
F(I)
B(I)
输出码元

图 5

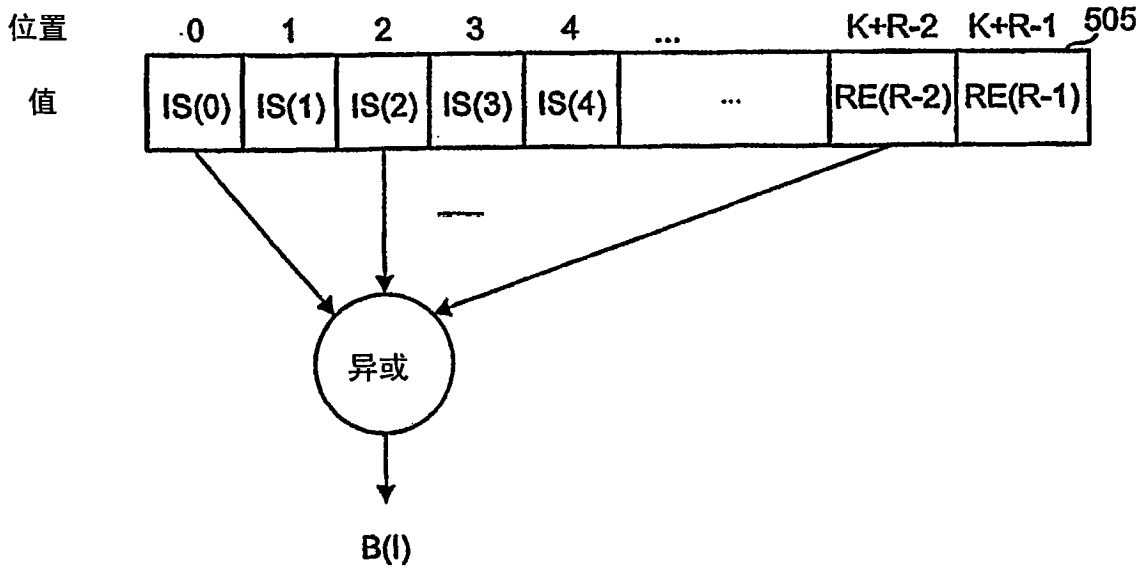


图 6

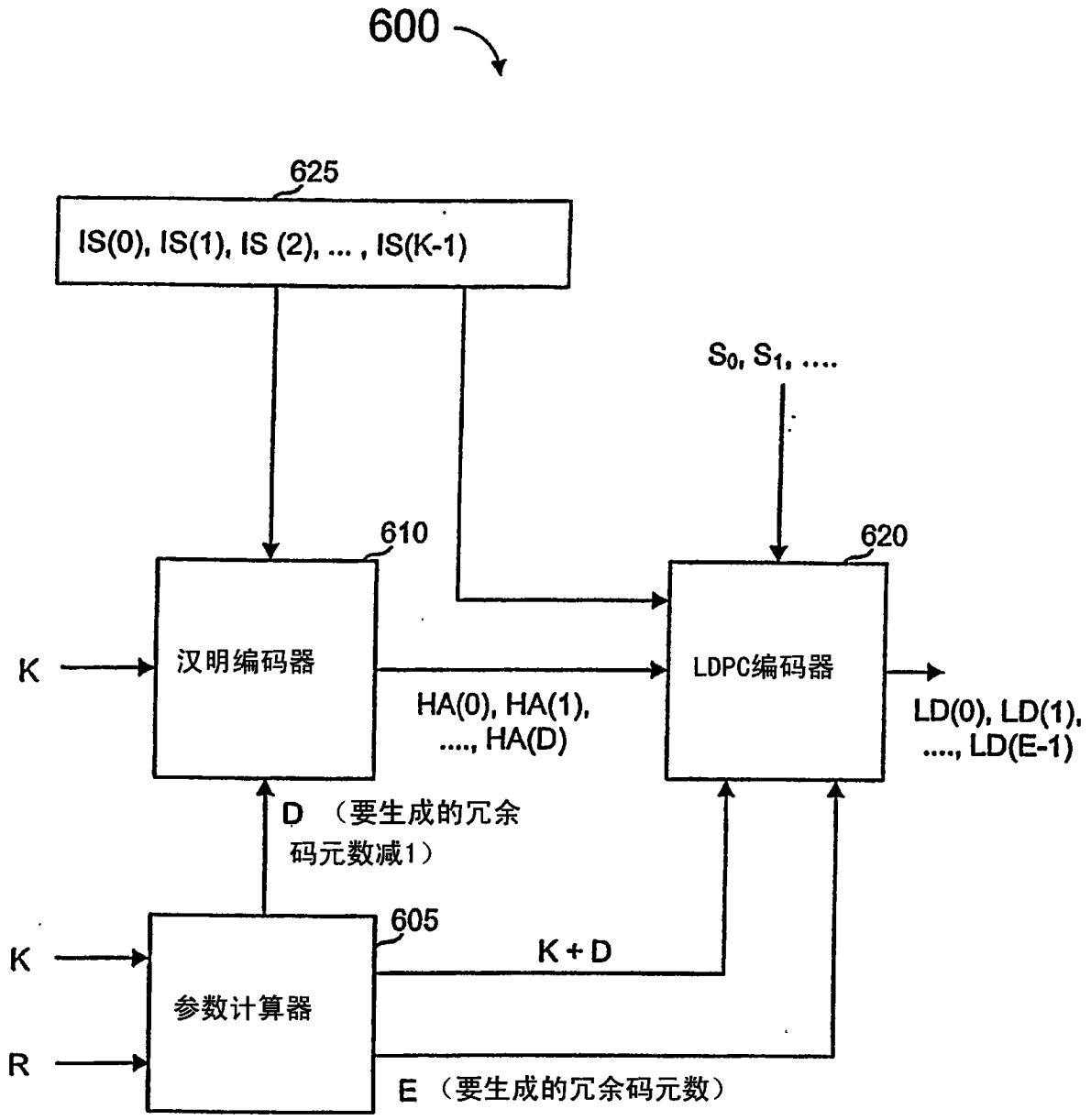

600
625
IS(0), IS(1), IS (2), ... , IS(K-1)
S0, S1, ...
610
620
K
汉明编码器
LDPC编码器
HA(0), HA(1), ..., HA(D)
LD(0), LD(1), ..., LD(E-1)
D （要生成的冗余码元数减1）
605
K
K + D
参数计算器
R
E （要生成的冗余码元数）

图 7

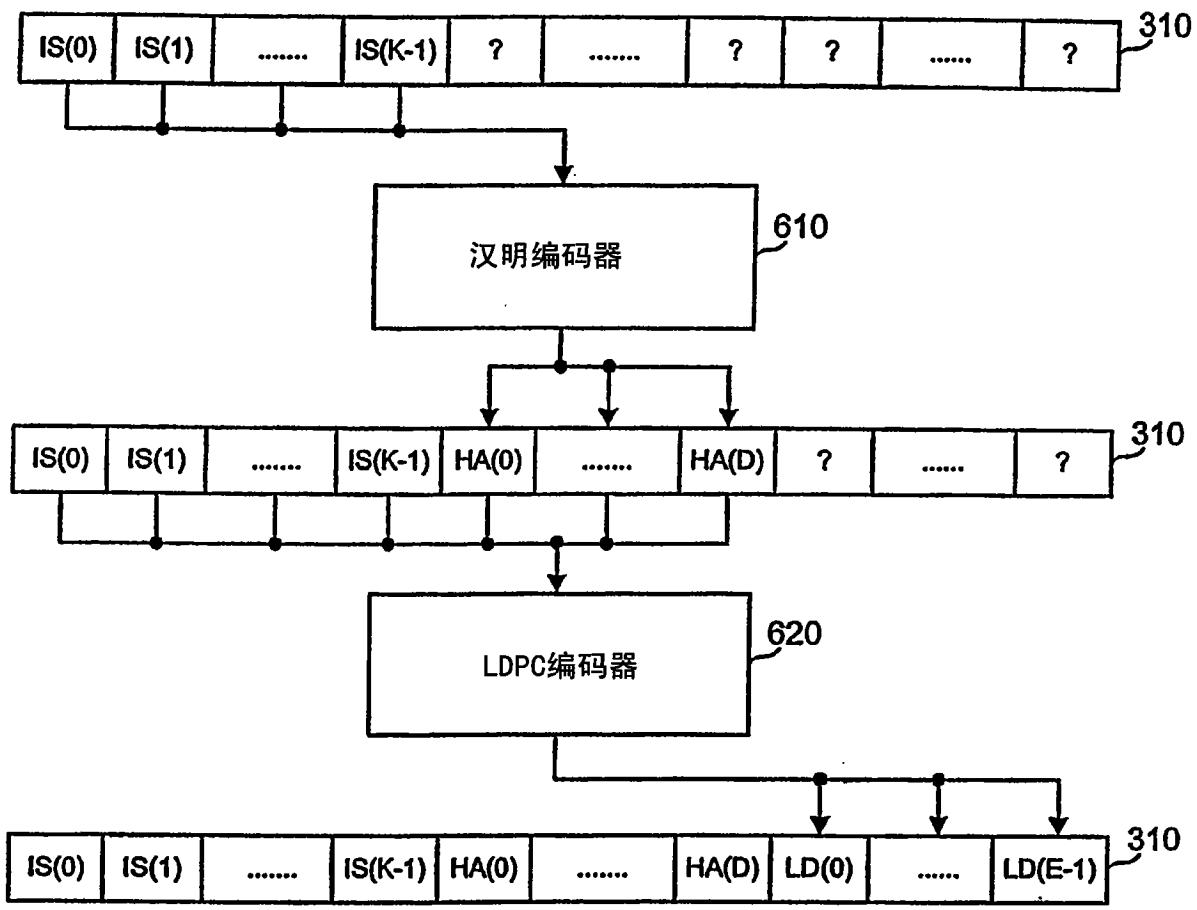

IS(0)
IS(1)
.....
IS(K-1)
?
.....
?
?
.....
?
310
汉明编码器
610
IS(0)
IS(1)
.....
IS(K-1)
HA(0)
.....
HA(D)
?
.....
?
310
LDPC编码器
620
IS(0)
IS(1)
.....
IS(K-1)
HA(0)
.....
HA(D)
LD(0)
.....
LD(E-1)
310

图 8

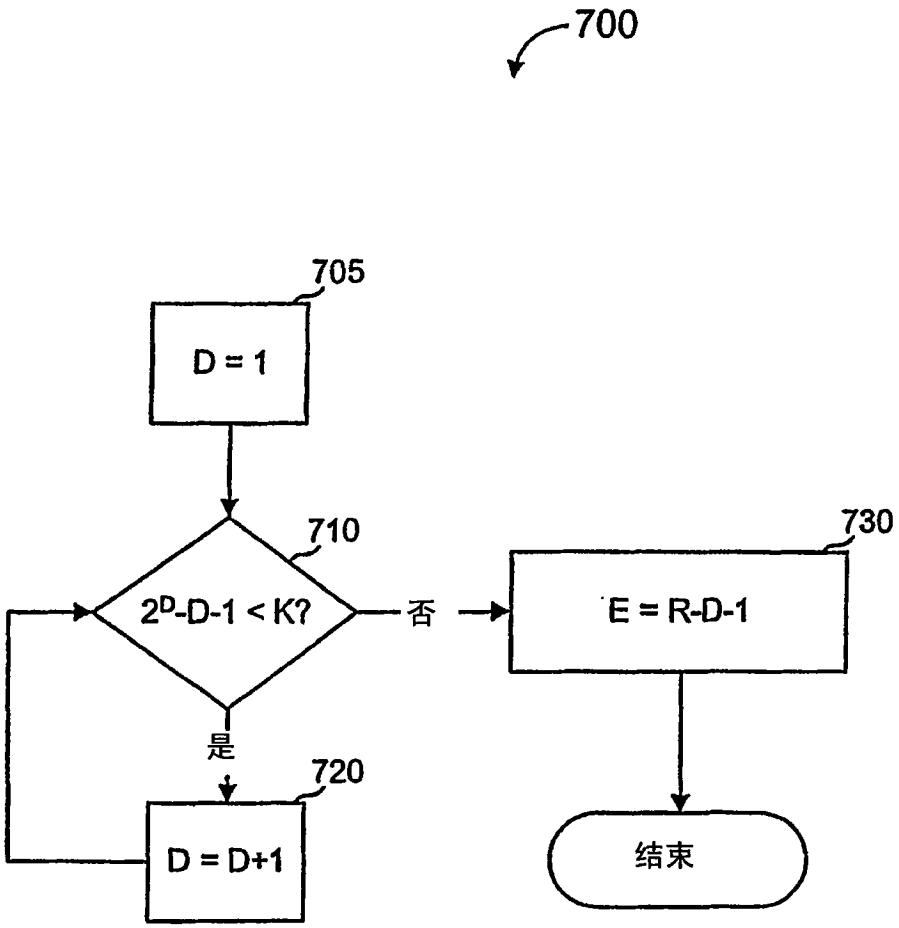


图 9

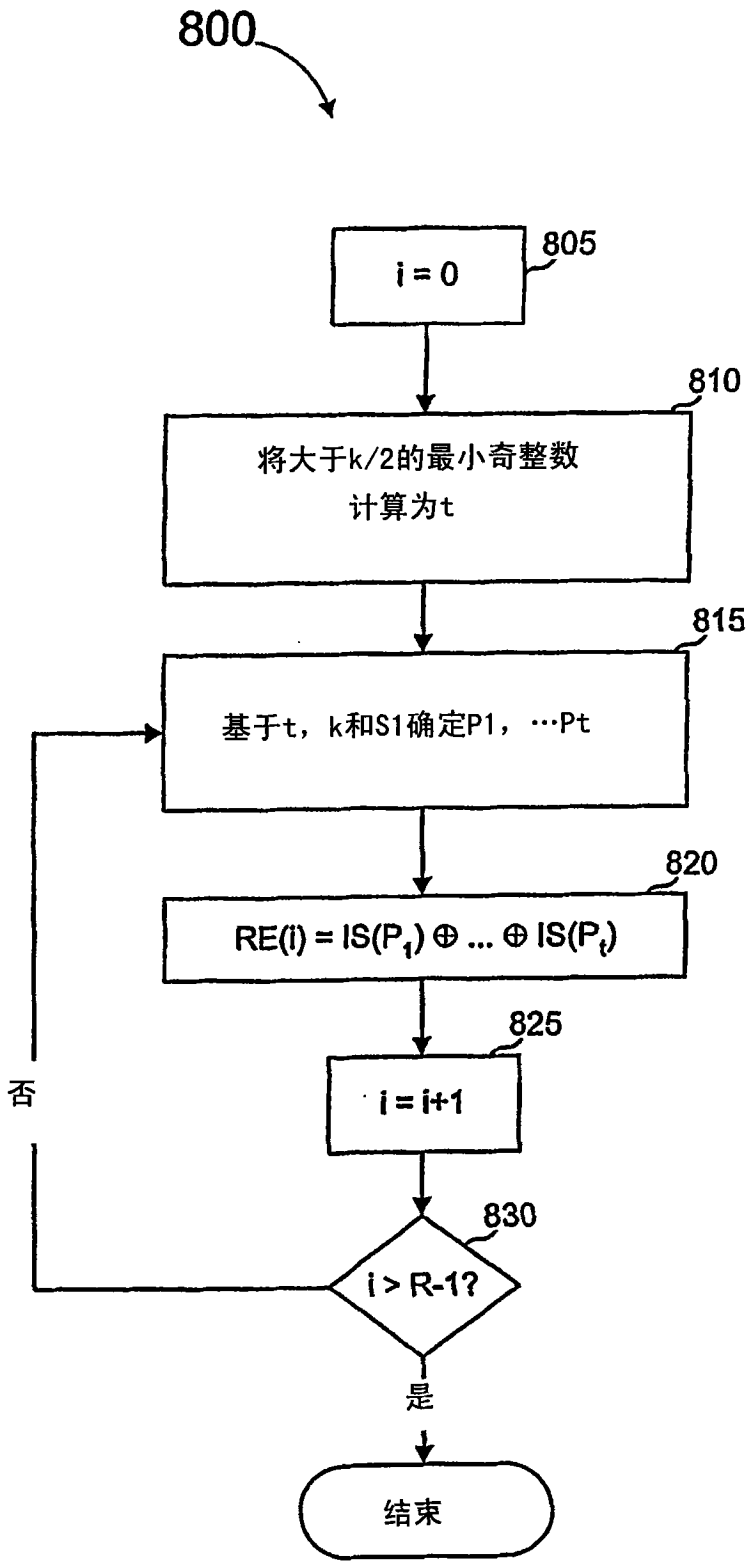

800
805
i = 0
810
将大于k/2的最小奇整数
计算为t
815
基于t，k和S1确定P1，…Pt
820
RE(i) = IS(P1) ⊕ ... ⊕ IS(Pt)
825
i = i+1
830
i > R-1?
否
是
结束

图 10

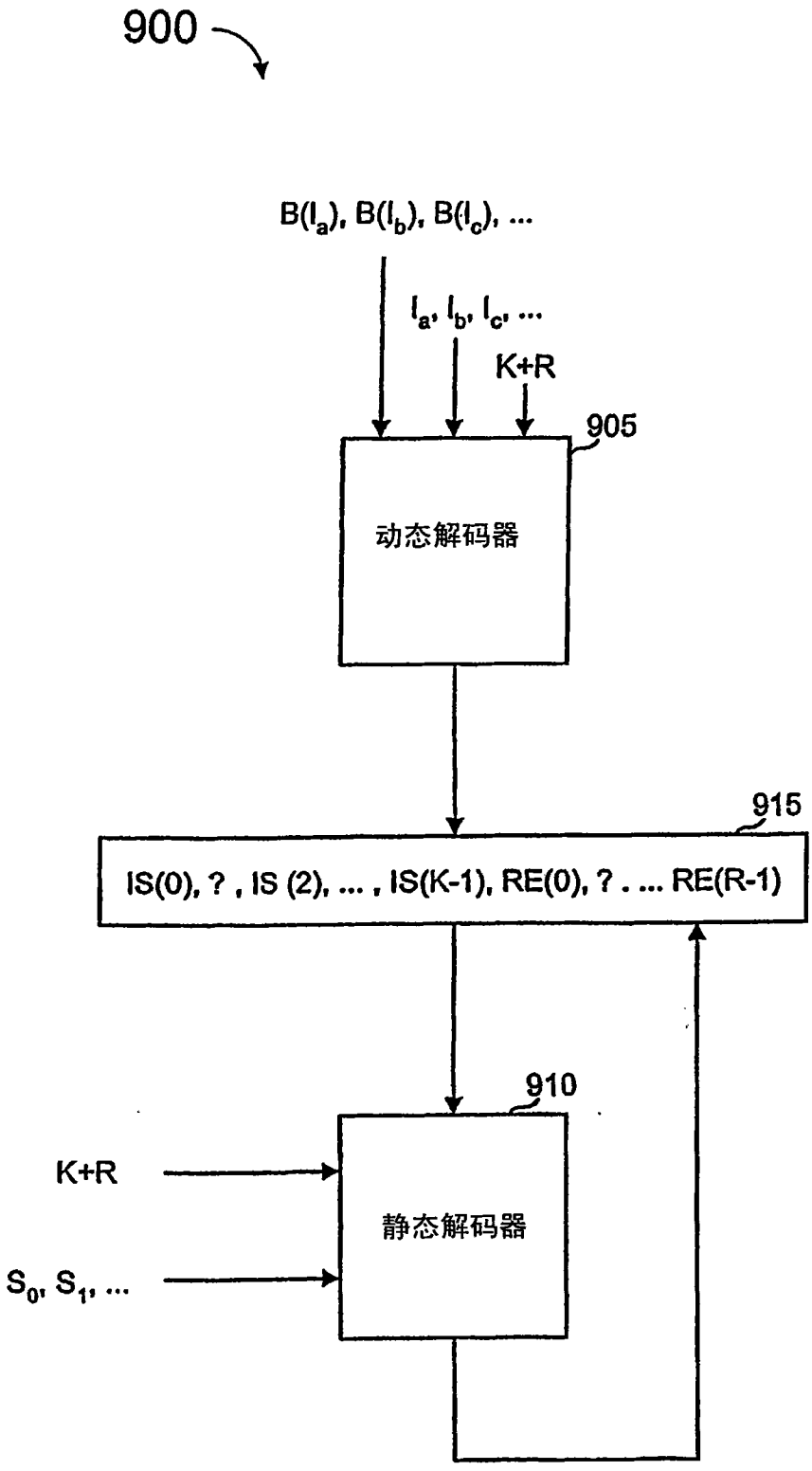

900
B(I_a), B(I_b), B(I_c), ...
I_a, I_b, I_c, ...
K+R
905
动态解码器
915
IS(0), ? , IS (2), ... , IS(K-1), RE(0), ? . ... RE(R-1)
910
K+R
静态解码器
S_0, S_1, ...

图 11

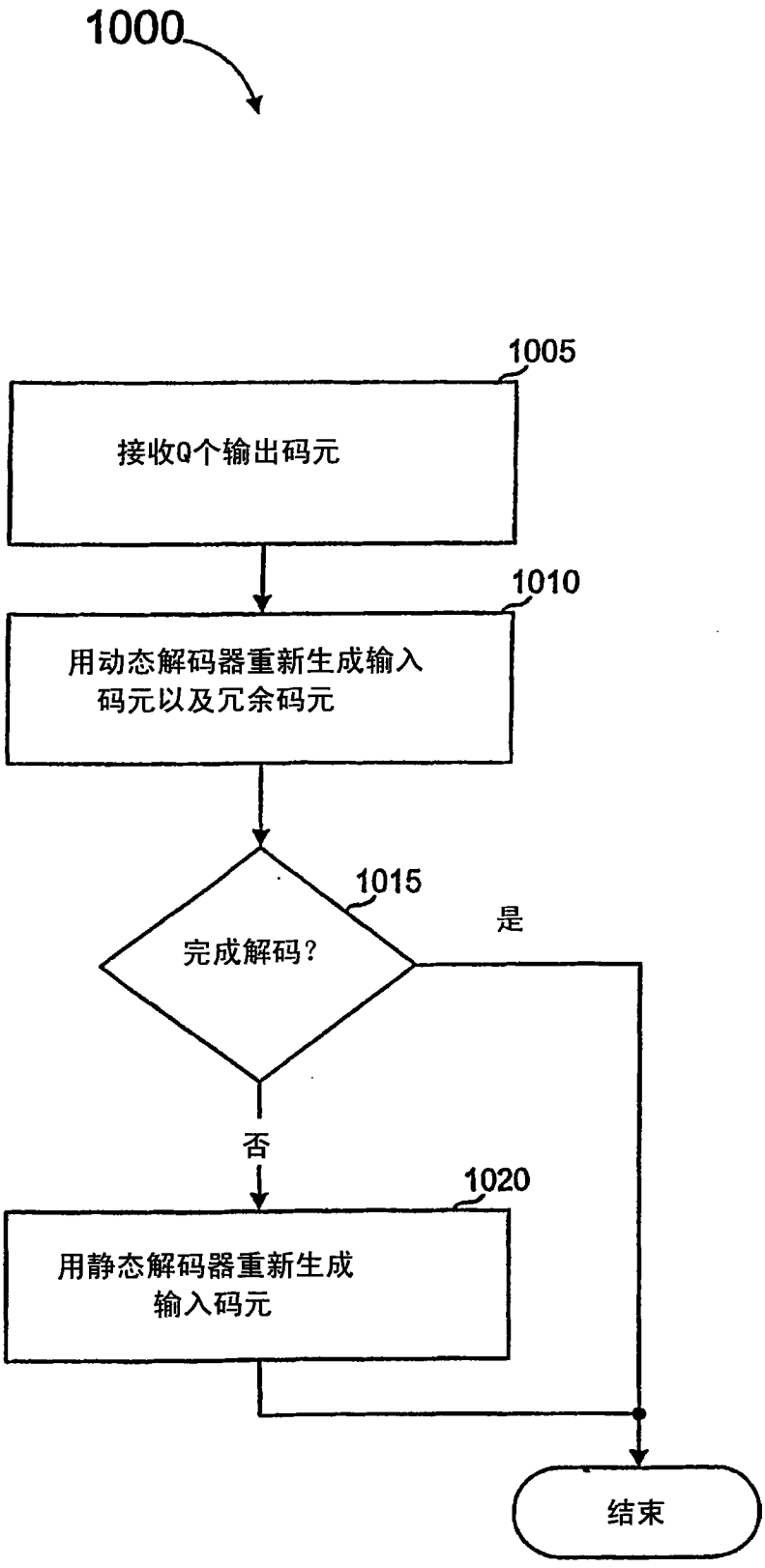

1000
1005
接收Q个输出码元
1010
用动态解码器重新生成输入码元以及冗余码元
1015
完成解码?
是
否
1020
用静态解码器重新生成输入码元
结束

图 12

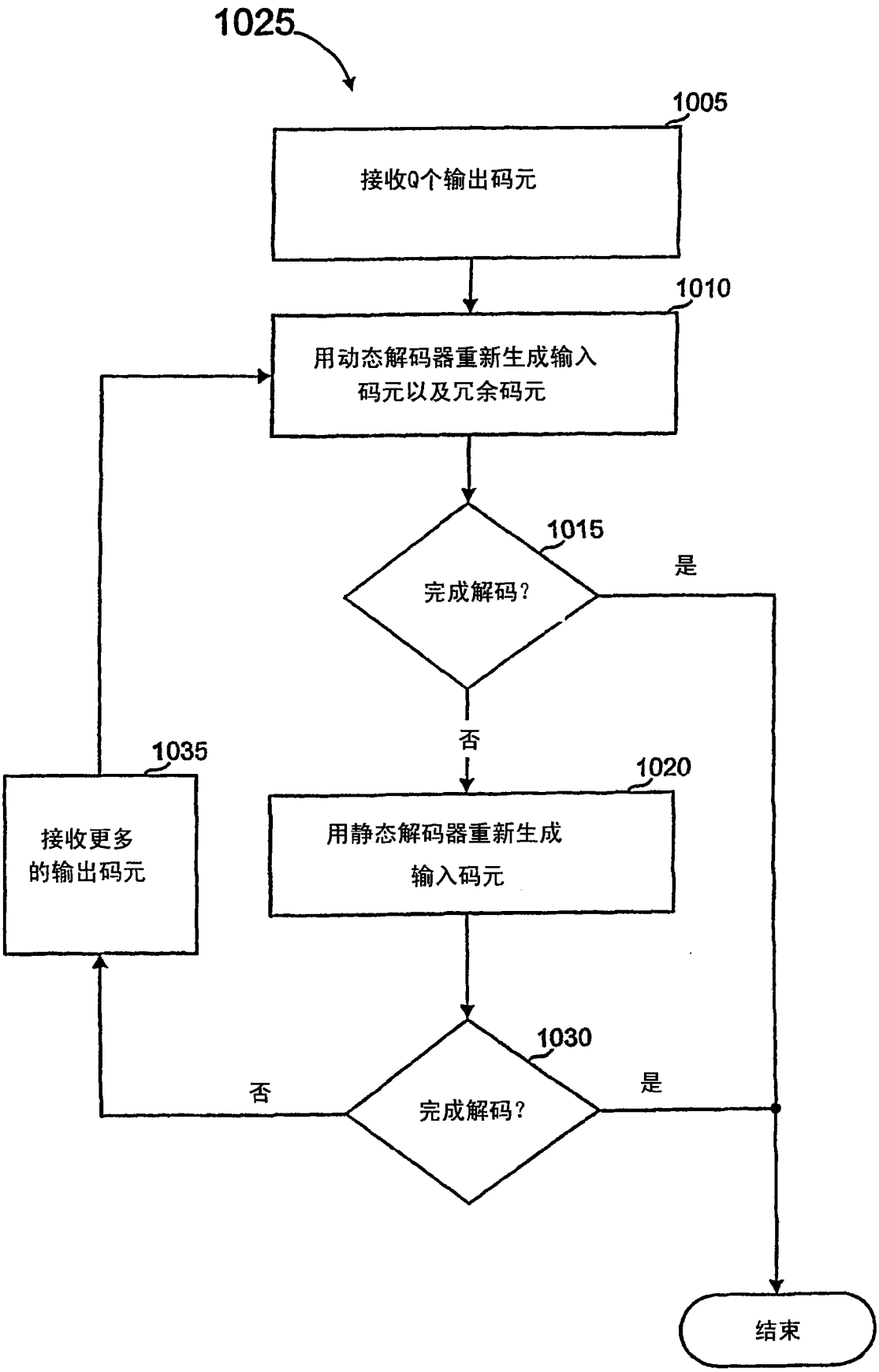

1025
1005
接收Q个输出码元
1010
用动态解码器重新生成输入码元以及冗余码元
1015
完成解码?
是
否
1035
接收更多的输出码元
1020
用静态解码器重新生成输入码元
1030
完成解码?
否
是
结束

图 13

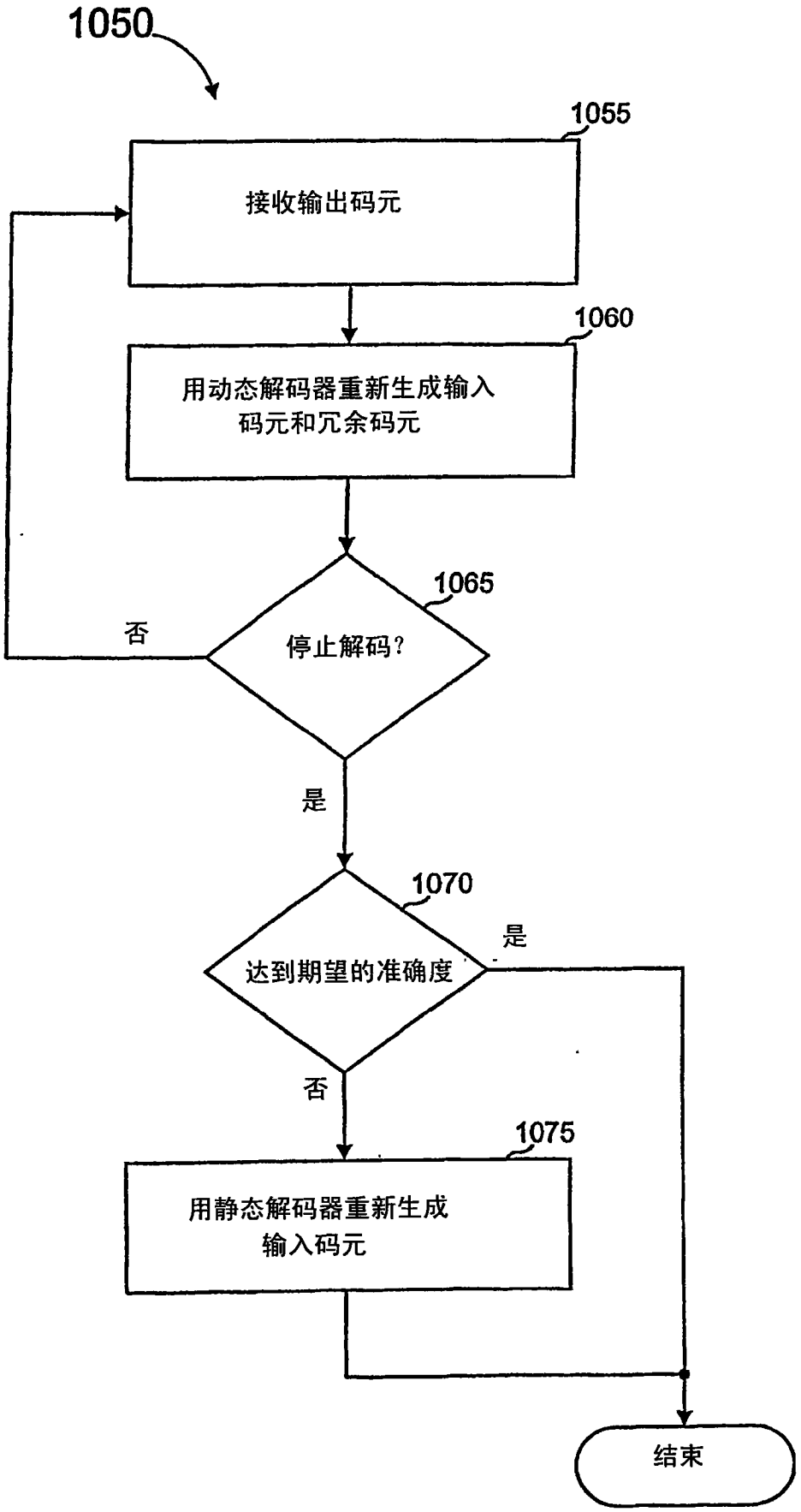

1050
1055
接收输出码元
1060
用动态解码器重新生成输入码元和冗余码元
1065
停止解码?
否
是
1070
达到期望的准确度
是
否
1075
用静态解码器重新生成输入码元
结束

图 14

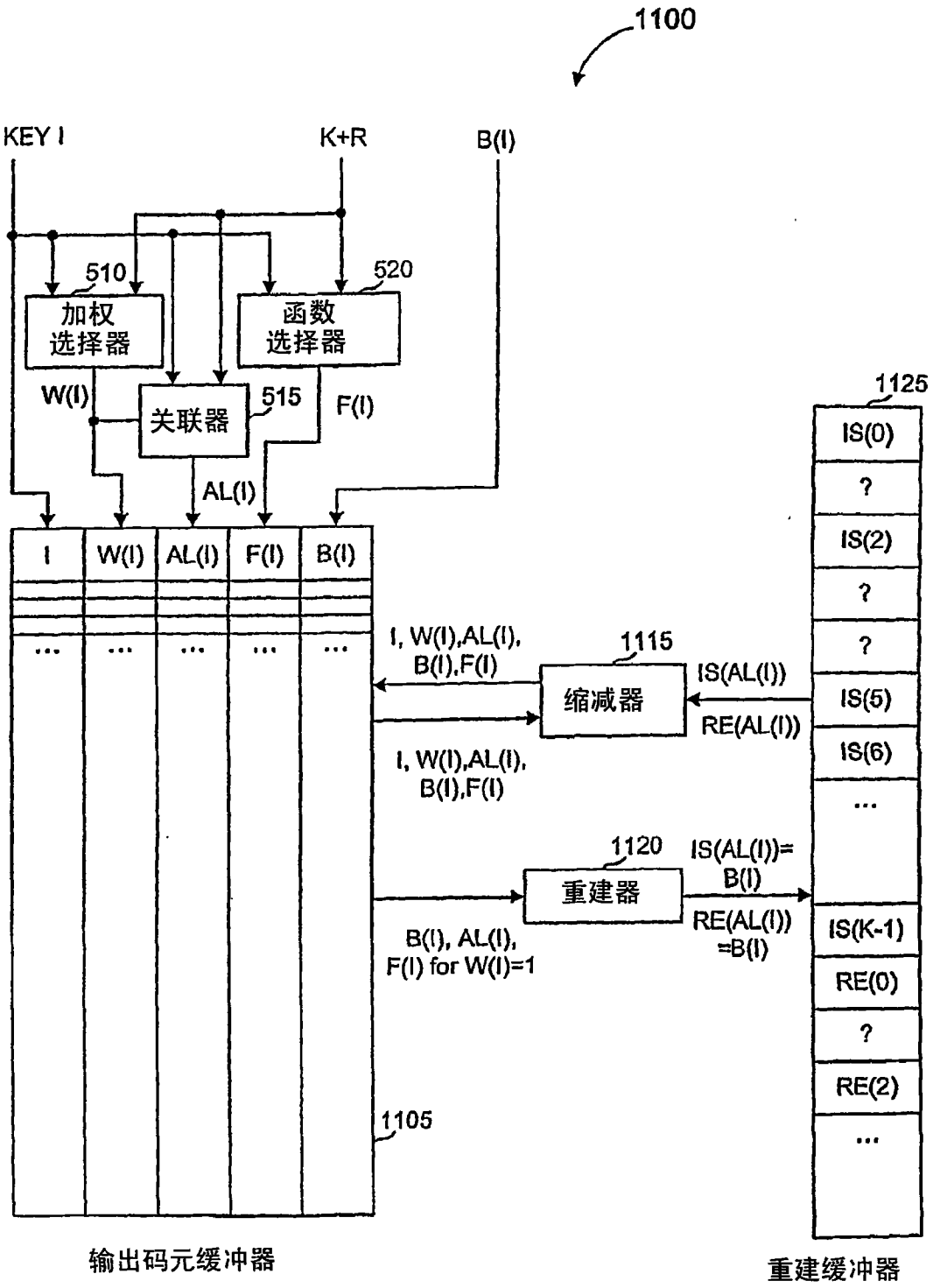

1100
KEY I
K+R
B(I)
510
加权
选择器
520
函数
选择器
W(I)
关联器
515
F(I)
AL(I)
I
W(I)
AL(I)
F(I)
B(I)
1125
IS(0)
?
IS(2)
?
?
IS(5)
IS(6)
...
IS(K-1)
RE(0)
?
RE(2)
...
I, W(I),AL(I),
B(I),F(I)
1115
缩减器
IS(AL(I))
RE(AL(I))
I, W(I),AL(I),
B(I),F(I)
1120
重建器
IS(AL(I))=
B(I)
RE(AL(I))
=B(I)
B(I), AL(I),
F(I) for W(I)=1
1105
输出码元缓冲器
重建缓冲器

图 15

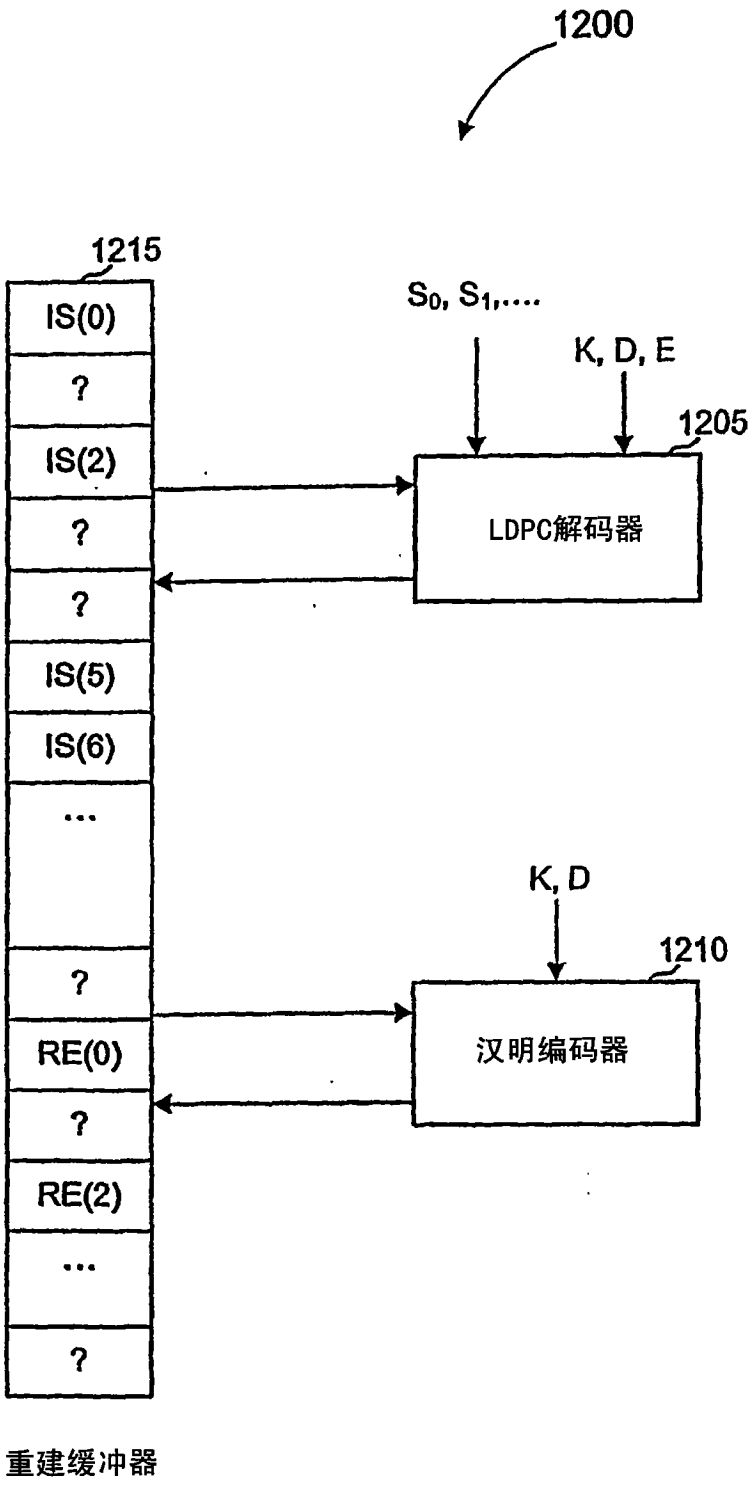

1200
1215
IS(0)
?
IS(2)
?
?
IS(5)
IS(6)
...
?
RE(0)
?
RE(2)
...
?
重建缓冲器
$S_0, S_1, \ldots$
K, D, E
1205
LDPC解码器
K, D
1210
汉明编码器

图 16

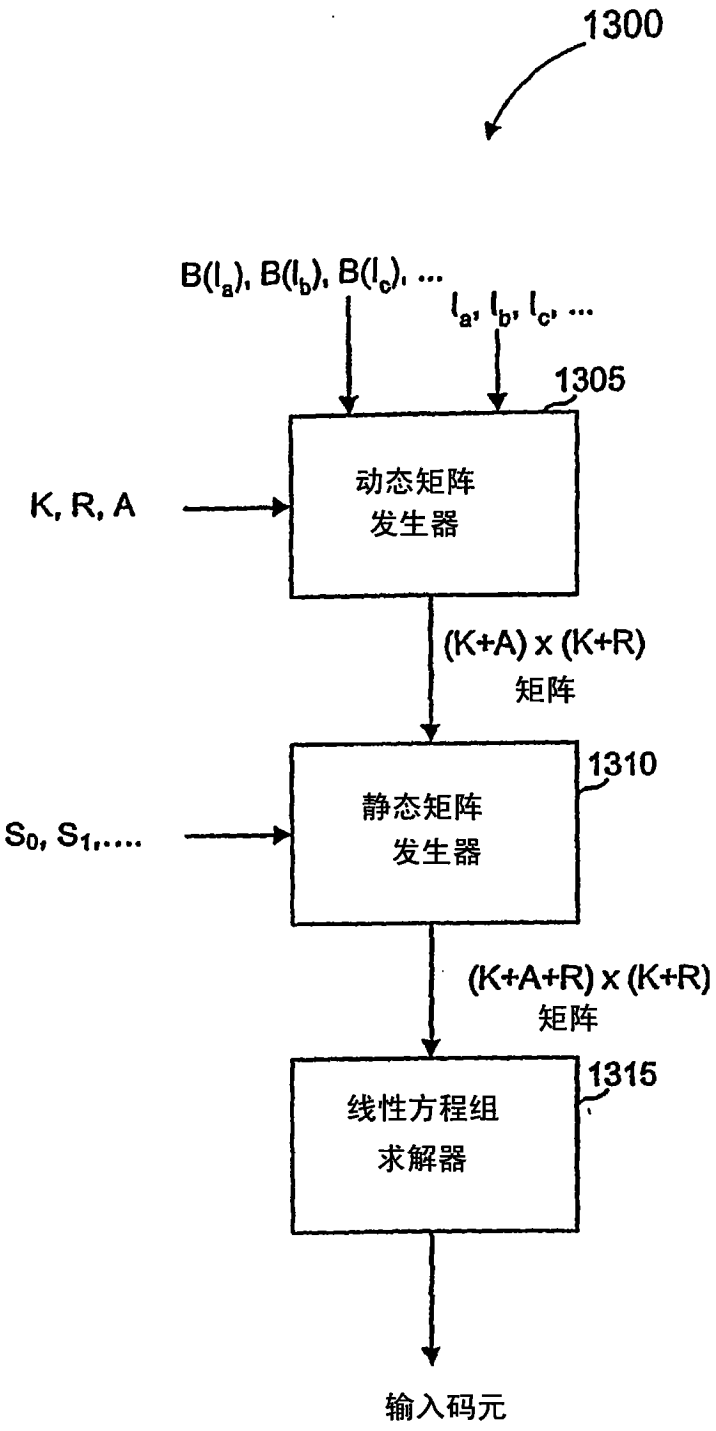

1300
B(I_a), B(I_b), B(I_c), ...
I_a, I_b, I_c, ...
1305
K, R, A
动态矩阵
发生器
(K+A) x (K+R)
矩阵
1310
S_0, S_1,....
静态矩阵
发生器
(K+A+R) x (K+R)
矩阵
1315
线性方程组
求解器
输入码元

图 17

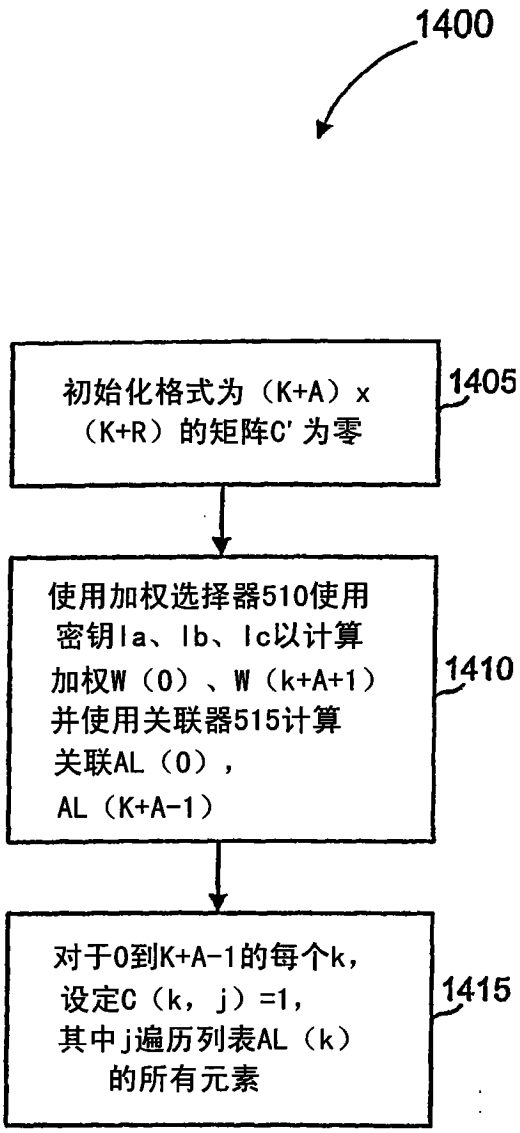

1400
初始化格式为（K+A）x（K+R）的矩阵C'为零
1405
使用加权选择器510使用密钥Ia、Ib、Ic以计算加权W（0）、W（k+A+1）并使用关联器515计算关联AL（0），AL（K+A-1）
1410
对于0到K+A-1的每个k，设定C（k，j）=1，其中j遍历列表AL（k）的所有元素
1415

图 18

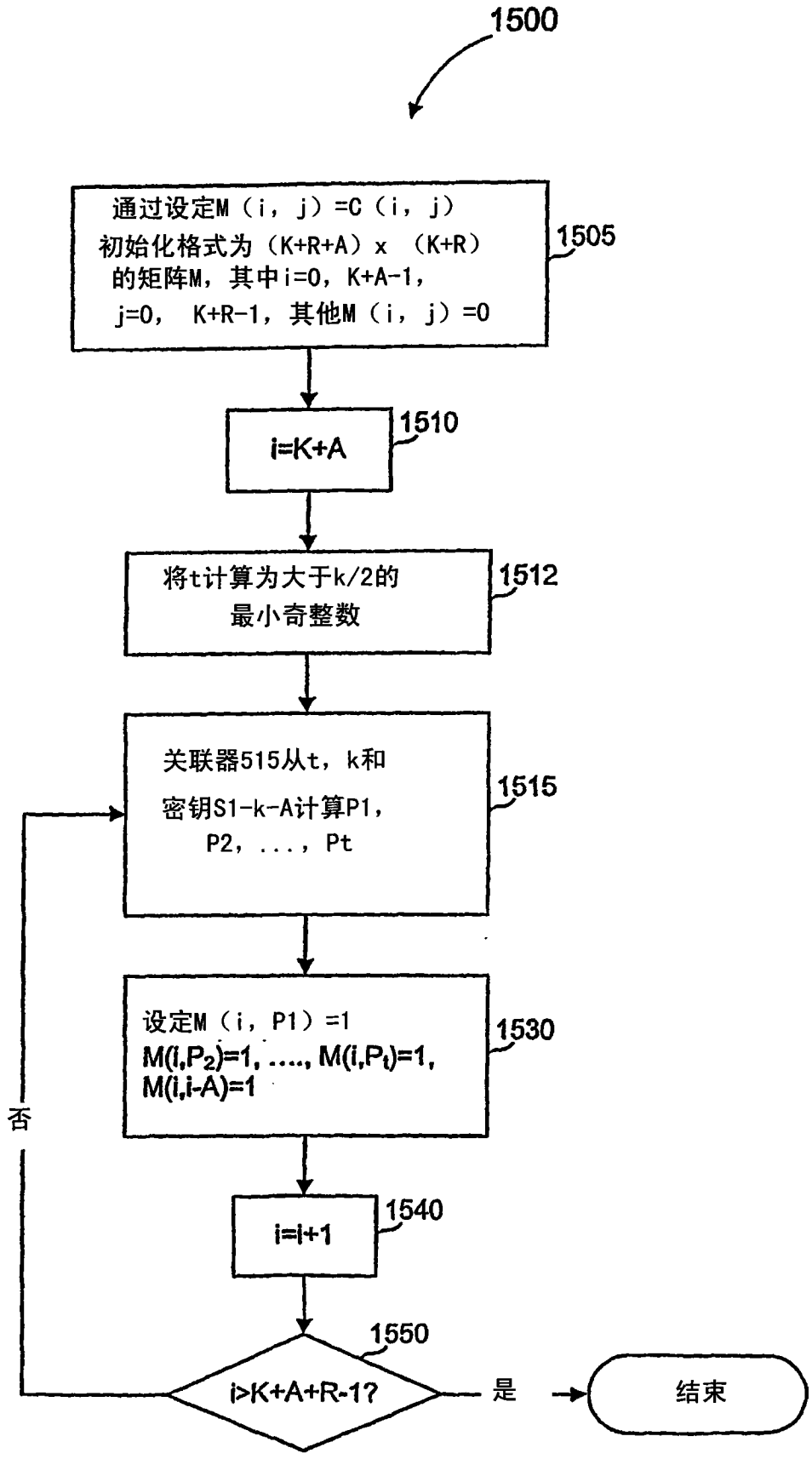

1500
通过设定M（i，j）=C（i，j）
初始化格式为（K+R+A）x （K+R）
的矩阵M，其中i=0，K+A-1，
j=0， K+R-1，其他M（i，j）=0
1505
i=K+A
1510
将t计算为大于k/2的
最小奇整数
1512
关联器515从t，k和
密钥S1-k-A计算P1，
P2，...，Pt
1515
设定M（i，P1）=1
M(i,P2)=1, ..., M(i,Pt)=1,
M(i,i-A)=1
1530
否
i=i+1
1540
1550
i>K+A+R-1?
是
结束

图 19

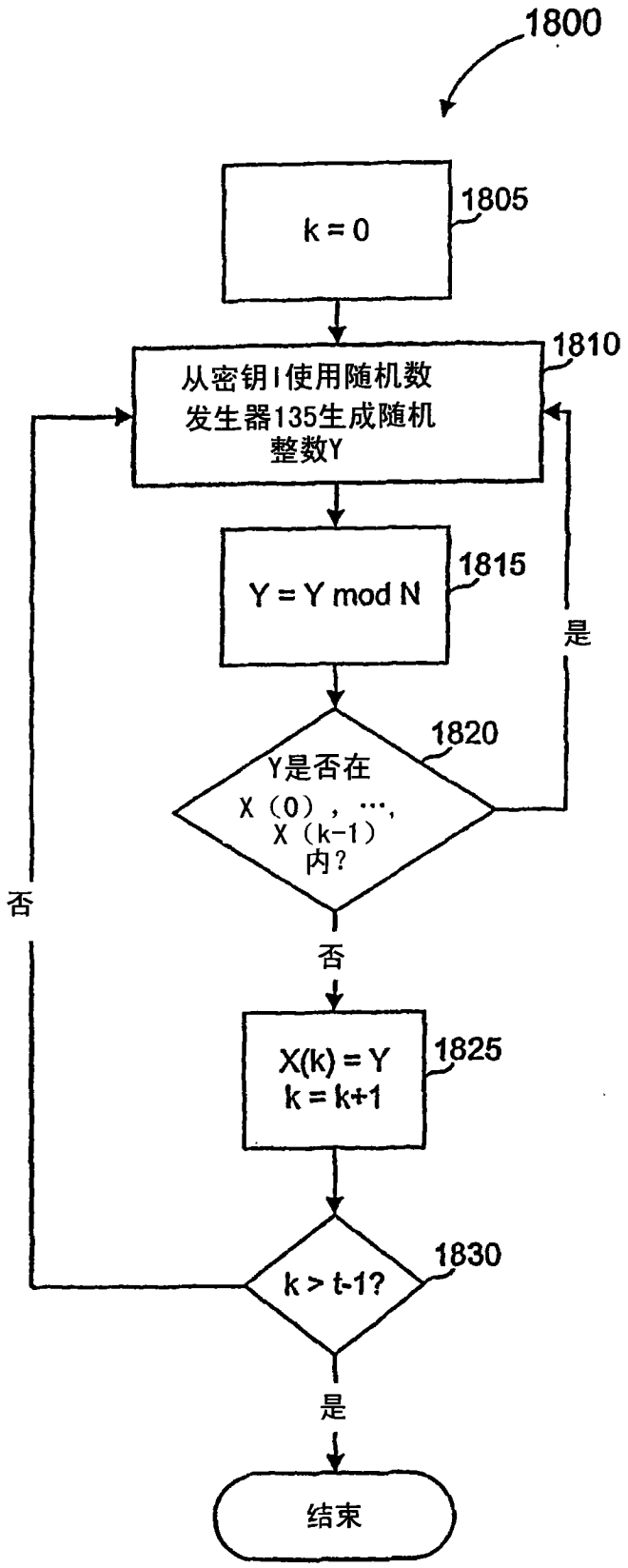

1800
k = 0
1805
从密钥I使用随机数发生器135生成随机整数Y
1810
Y = Y mod N
1815
Y是否在X（0），…，X（k-1）内？
1820
是
否
否
X(k) = Y
k = k+1
1825
k > t-1?
1830
是
结束

图 20

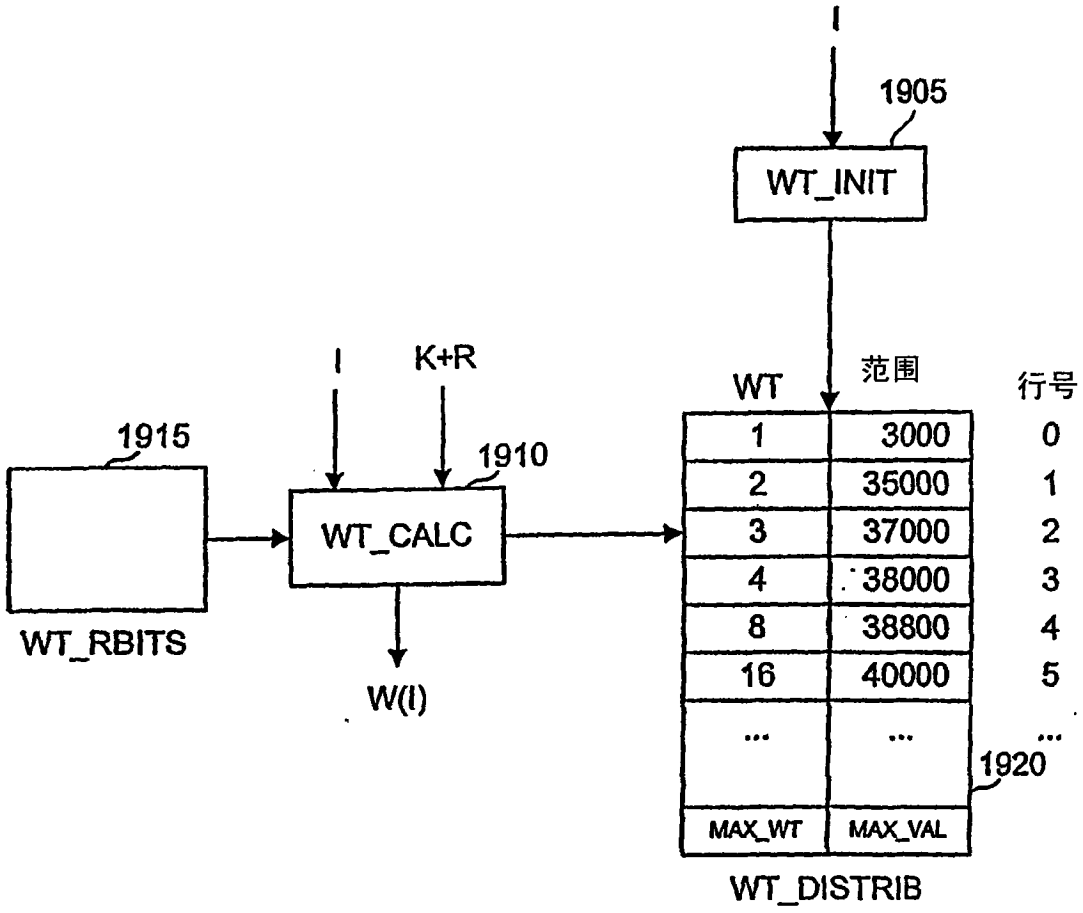

I
1905
WT_INIT
I
K+R
1915
1910
WT_CALC
WT_RBITS
W(I)
WT
范围
行号
1 3000 0
2 35000 1
3 37000 2
4 38000 3
8 38800 4
16 40000 5
... ...
1920
MAX_WT MAX_VAL
WT_DISTRIB

图 21

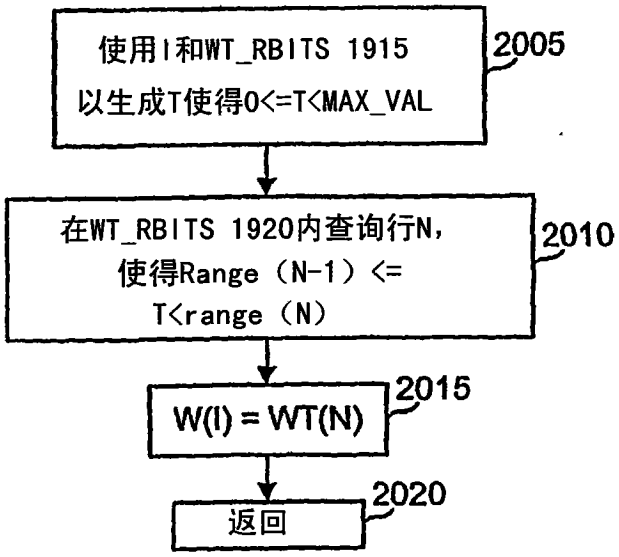

使用I和WT_RBITS 1915
以生成T使得0<=T<MAX_VAL
2005
在WT_RBITS 1920内查询行N，
使得Range（N-1）<=
T<range（N）
2010
W(I) = WT(N)
2015
返回
2020

图 22