



(12)发明专利

(10)授权公告号 CN 104022786 B

(45)授权公告日 2017.09.01

(21)申请号 201410221533.2

(22)申请日 2014.05.21

(65)同一申请的已公布的文献号

申请公布号 CN 104022786 A

(43)申请公布日 2014.09.03

(73)专利权人 上海宏光经济信息发展中心青岛
电子技术部

地址 266034 山东省青岛市四方区抚顺路5
号

(72)发明人 张宝东 姜淑敏 孙伟 李钊
班荣峰 迟松 徐蕾 曹建军
郑建云 陈鉴全 张良胜

(74)专利代理机构 国防专利服务中心 11043
代理人 文伟能

(51)Int.Cl.

H03M 13/29(2006.01)

(56)对比文件

CN 1805293 A,2006.07.19,

CN 1555603 A,2004.12.15,

CN 1266555 A,2000.09.13,

CN 1688111 A,2005.10.26,

EP 1443657 A1,2004.08.04,

US 2013132794 A1,2013.05.23,

Ramesh PYNDIAH 等.NEAR OPTIMUM

DECODING OF PRODUCT CODES.《IEEE

Transactions on Communications》.1998,第46
卷(第8期),339-343.

审查员 刘蕾蕾

权利要求书2页 说明书10页 附图3页

(54)发明名称

乘积码译码方法

(57)摘要

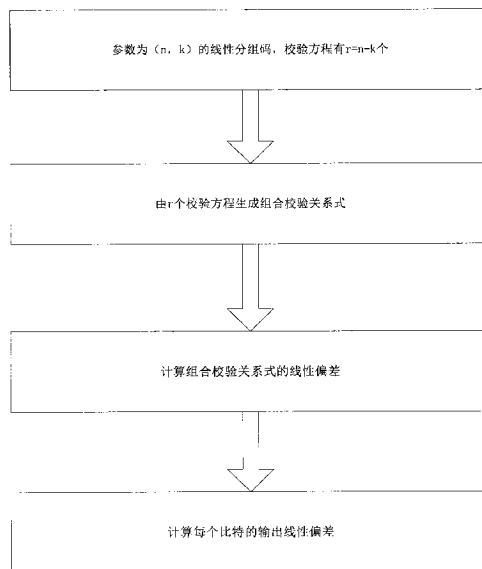
本发明涉及通信技术领域,涉及一种乘积码
译码方法,按照公式

$$\frac{p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=0) - p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=1)}{p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=0) + p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=1)} = \tilde{p}(\xi_i | L),$$

来计算每个比特满足r个线性校验关系式 $L_0=0, \dots,$
 $L_{r-1}=0$ 条件下的线性偏差 $\tilde{p}(\xi_i | L)$,同时提供一种

乘积码译码方法,按照方程式 $\tilde{p}(\xi_i) = \frac{\tilde{p}_1(\xi_i) + \tilde{p}_2(\xi_i)}{1 + \tilde{p}_1(\xi_i)\tilde{p}_2(\xi_i)}$

作为混合计算方程式计算更新的线性偏差。本发明
通过输入线性偏差和堆积引理计算校验全空间所有
组合的校验关系式的线性偏差,进而通过沃尔什逆
变换求出每个比特的输出线性偏差,和现有技术相
比,对相同数据进行译码,译码效果更好,比特错误
率低于现有译码方法;通过行、列迭代方法中的混
合运算方程式对数据中的错误比特进行修正,降低
了译码错误率,提高了译码性能。



1. 一种乘积码译码方法, 乘积码 $D = C_1 \otimes C_2$, 每行是参数为 (n_2, k_2, δ_2) 的线性分组码 C_2 , 每列是参数为 (n_1, k_1, δ_1) 的线性分组码 C_1 , n_i, k_i, δ_i ($i = 1, 2$) 分别表示码字长度、信息位个数、最小汉明距离; 对于乘积码 $D = C_1 \otimes C_2$ 采用行、列迭代方法进行译码, 其特征在于, 迭代译码方法为:

(一), 列译码: 乘积码共 n_2 列, 每列是参数为 (n_1, k_1, δ_1) 线性分组码, 采用线性分组码输出线性偏差计算方法计算出每个比特的输出线性偏差, 在此记作 $\tilde{p}_2(\xi_i)$; 将每个比特的输出线性偏差 $\tilde{p}_2(\xi_i)$ 与输入线性偏差 $\tilde{p}_1(\xi_i)$ 按方程式 $\tilde{p}(\xi_i) = \frac{\tilde{p}_1(\xi_i) + \tilde{p}_2(\xi_i)}{1 + \tilde{p}_1(\xi_i)\tilde{p}_2(\xi_i)}$ 进行混合运算, 得到更新的输出线性偏差 $\tilde{p}(\xi_i)$;

(二), 行译码: 乘积码共 n_1 行, 每行是参数为 (n_2, k_2, δ_2) 线性分组码, 将列译码得到的每个比特的更新的输出线性偏差 $\tilde{p}(\xi_i)$ 作为行译码的输入线性偏差, 记作 $\tilde{p}'_1(\xi_i)$, 采用线性分组码输出线性偏差计算方法计算出每个比特的输出线性偏差, 在此记作 $\tilde{p}'_2(\xi_i)$; 将每个比特的输出线性偏差 $\tilde{p}'_2(\xi_i)$ 与输入线性偏差 $\tilde{p}'_1(\xi_i)$ 按方程式 $\tilde{p}'(\xi_i) = \frac{\tilde{p}'_1(\xi_i) + \tilde{p}'_2(\xi_i)}{1 + \tilde{p}'_1(\xi_i)\tilde{p}'_2(\xi_i)}$ 进行混合运算, 得到更新的输出线性偏差 $\tilde{p}'(\xi_i)$; 一次迭代结束;

将上一次迭代中行译码得到的输出线性偏差 $\tilde{P}'(\xi_{i,0}), \dots, \tilde{P}'(\xi_{i,n_2-1})$ 作为下一次迭代列译码中的初始输入线性偏差;

(三), 对乘积码 $D = C_1 \otimes C_2$ 进行数次迭代, 当最终的输出线性偏差收敛到 +1 或 -1 时, 迭代结束;

其中, 乘积码由线性分组码构建, 对于参数为 (n, k) 的线性分组码, 线性分组码的每个码字为 n 比特向量 $\xi = (\xi_0, \xi_1, \dots, \xi_{n-1})$, 校验位有 $r = n - k$ 比特, 即有 r 个校验方程, r 个校验方程都为线性校验关系式, 记为 $L_0 = 0, \dots, L_{r-1} = 0$, $\tilde{p}(\xi_i)$ 为随机变量 $\xi_0, \xi_1, \dots, \xi_{n-1}$ 的输入线性偏差, 线性分组码输出线性偏差计算如下步骤:

步骤一、由 r 个校验方程生成 2^r 个组合校验关系式 $f_c(\xi)$:

每个校验方程都是对随机变量 $\xi_0, \xi_1, \dots, \xi_{n-1}$ 的一种线性组合, 得到 r 个随机变量 L_0, \dots, L_{r-1} , 对随机变量 L_0, \dots, L_{r-1} 进行校验全空间的线性组合, 共得到 2^r 个线性组合, 对应着 2^r 个组合校验关系式, 记为 $f_c(\xi)$, 其中 c 为线性系数状态, $c = (c_0, c_1, \dots, c_{r-1})$, $c \in [0, 2^r - 1]$;

步骤二、由输入线性偏差 $\tilde{p}(\xi_i)$ 和堆积引理计算 2^r 个组合校验关系式的线性偏差 $\tilde{p}_f(c)$, $\tilde{p}_f(c) = \tilde{p}_f(c_0, c_1, \dots, c_{r-1}) = \tilde{p}_f(c_0 L_0 \oplus c_1 L_1 \oplus \dots \oplus c_{r-1} L_{r-1})$

$\square \tilde{p}(d_0 \xi_0 \oplus d_1 \xi_1 \oplus \dots \oplus d_{n-1} \xi_{n-1}) = \prod_{d_i=1, i \in [0, n-1]} \tilde{p}(\xi_i), \tilde{p}_f(c)$ 的线性组合有 2^r 个, 记为 $\tilde{p}_f(0), \tilde{p}_f(1), \dots, \tilde{p}_f(2^r - 1)$; 其中 $\tilde{p}(\xi_i) = p(\xi_i = 0) - p(\xi_i = 1), i = 0, 1, \dots, n-1$; \square 表示随着线性系数状态 c 的变化, 随机变量 $\xi_0, \xi_1, \dots, \xi_{n-1}$ 的组合系数也将产生变化;

步骤三、计算每个比特的输出线性偏差:

(1) 由沃尔什逆变换得, $p(\mathbf{0}) = 2^{-r} \sum_{c=0}^{2^r-1} \tilde{p}_f(c)$;

其中, $p(\mathbf{0})$ 表示 $L_0=0, \dots, L_{r-1}=0$ 出现的概率,

$p(\mathbf{0}) = p(L_0=0, L_1=0, \dots, L_{r-1}=0)$;

(2) $p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=0)$ 、 $p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=1)$;

令 $\xi_i=0$ ($i=0, 1, \dots, n-1$), 代入 L_0, \dots, L_{r-1} , 计算出 2^r 个线性组合的线性偏差 $\tilde{p}_f(\mathbf{0}), \tilde{p}_f(\mathbf{1}), \dots, \tilde{p}_f(2^r-1)$, 将 2^r 个线性组合的线性偏差相加, 得到 $p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=0)$;

再令 $\xi_i=1$ ($i=0, 1, \dots, n-1$), 代入 L_0, \dots, L_{r-1} , 计算出 2^r 个线性组合的线性偏差 $\tilde{p}_f(\mathbf{0}), \tilde{p}_f(\mathbf{1}), \dots, \tilde{p}_f(2^r-1)$, 将 2^r 个线性组合的 2^r 个线性偏差相加, 得到 $p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=1)$;

其中, $p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=0)$ 和 $p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=1)$ 分别表示 $L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=0$ 和 $L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=1$ 出现的概率;

(3) 每个比特的输出线性偏差为每个比特满足 r 个线性校验关系式 $L_0=0, \dots, L_{r-1}=0$ 条件下的线性偏差 $\tilde{p}(\xi_i | L_0=0, \dots, L_{r-1}=0)$, 简记为 $\tilde{p}(\xi_i | L)$:

$$\tilde{p}(\xi_i | L) = \frac{p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=0) - p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=1)}{p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=0) + p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=1)}.$$

乘积码译码方法

技术领域

[0001] 本发明涉及通信技术领域,特别涉及乘积码译码方法。

背景技术

[0002] 在现有的信道编码技术中,线性分组码具有明显的数据结构,是许多性能良好编码的基础。对于乘积码,由两个线性分组码 C_1 、 C_2 构造的乘积码 $D = C_1 \otimes C_2$ 的构造如附图1所示。

[0003] 设 C_1 的编码参数为 (n_1, k_1, δ_1) 、 C_2 的编码参数为 (n_2, k_2, δ_2) ,其中 $n_i, k_i, \delta_i (i = 1, 2)$ 分别表示码字长度、信息位个数、最小汉明距离。在附图1中,将 $k_1 \times k_2$ 个信息比特排成 k_1 行、 k_2 列的矩阵。其中,前 k_1 行用 C_2 编码,每一列用 C_1 编码;后 $(n_1 - k_1)$ 行用 C_2 编码后 $(n_2 - k_2)$ 列用 C_1 编码。这样得到乘积码 D 的参数为 $n = n_1 \times n_2, k = k_1 \times k_2, \delta = \delta_1 \times \delta_2$,编码效率为 $R = R_1 \times R_2$,其中 R_1 为 C_1 的编码效率, R_2 为 C_2 的编码效率。

[0004] 显然,线性分组码的译码性能决定了乘积码的译码性能。当线性分组码的码长较大时,靠穷尽整个码字空间的最大似然译码方法(MLD)显然是不现实的。1972年,Chase提出了近似最大似然译码方法,计算复杂度大大降低。该方法是在先验概率已知的前提下,设定最不可信的点的范围,在缩小的码字范围内进行最大似然译码。1994年,法国科学家Ramesh Mahendra Pyndiah等人提出了一种乘积码的次优译码方法,该方法采用Chase译码思想,近似计算出每个比特位的可信度,通过乘积码行、列的多次迭代实现译码。此方法称为次优方法(Near-Optimum),是目前被广泛应用的乘积码译码方法。但是,该方法中每个比特的可信度是在一定范围内的近似计算,精确度不高,属于次优方法。

发明内容

[0005] 本发明需解决的技术问题是提供精度高的乘积码译码方法。

[0006] 为解决上述技术问题,本发明提供一种乘积码译码方法,乘积码由线性分组码构建,对于参数为 (n, k) 的线性分组码,线性分组码的每个码字为 n 比特向量 $\xi = (\xi_0, \xi_1, \dots, \xi_{n-1})$,校验位有 $r = n - k$ 比特,即有 r 个校验方程, r 个校验方程都为线性校验关系式,记为 $L_0 = 0, \dots, L_{r-1} = 0, \tilde{p}(\xi_i)$ 为随机变量 $\xi_0, \xi_1, \dots, \xi_{n-1}$ 的输入线性偏差,线性分组码输出线性偏差计算如下步骤:

[0007] 步骤一、由 r 个校验方程生成 2^r 个组合校验关系式 $f_c(\xi)$;

[0008] 每个校验方程都是对随机变量 $\xi_0, \xi_1, \dots, \xi_{n-1}$ 的一种线性组合,得到 r 个随机变量 L_0, \dots, L_{r-1} ,对随机变量 L_0, \dots, L_{r-1} 进行校验全空间的线性组合,共得到 2^r 个线性组合,对应着 2^r 个组合校验关系式,记为 $f_c(\xi)$,其中 c 为线性系数状态, $c = (c_0, c_1, \dots, c_{r-1}), c \in [0, 2^r - 1], ;$

[0009] 步骤二、由输入线性偏差 $\tilde{p}(\xi_i)$ 和堆积引理计算 2^r 个组合校验关系式的线性偏差

$$\tilde{p}_f(c), \tilde{p}_f(c) = \tilde{p}_f(c_0, c_1, \dots, c_{r-1}) = \tilde{p}_f(c_0 L_0 \oplus c_1 L_1 \oplus \dots \oplus c_{r-1} L_{r-1})$$

□ $\tilde{p}(d_0\xi_0 \oplus d_1\xi_1 \oplus \dots \oplus d_{n-1}\xi_{n-1}) = \prod_{d_i=1, i \in [0, n-1]} \tilde{p}(\xi_i)$, 其中 $\tilde{p}(\xi_i) = p(\xi_i=0) - p(\xi_i=1)$, $i=0, 1, \dots, n-1$; □

表示随着线性系数状态c的变化, 随机变量 $\xi_0, \xi_1, \dots, \xi_{n-1}$ 的组合系数也将产生变化, $\tilde{p}_f(c)$ 的线性组合有 2^r 个, 记为 $\tilde{p}_f(0), \tilde{p}_f(1), \dots, \tilde{p}_f(2^r-1)$;

[0010] 步骤三、计算每个比特的输出线性偏差:

[0011] (1) 由沃尔什逆变换得, $p(0) = 2^{-r} \sum_{c=0}^{2^r-1} \tilde{p}_f(c)$;

[0012] 其中, p(0) 表示 $L_0=0, \dots, L_{r-1}=0$ 出现的概率, $p(0) = p(L_0=0, L_1=0, \dots, L_{r-1}=0)$;

[0013] (2) $p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=0)$ 、 $p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=1)$;

[0014] 令 $\xi_i=0$ ($i=0, 1, \dots, n-1$), 代入 L_0, \dots, L_{r-1} , 计算出 2^r 个线性组合的线性偏差 $\tilde{p}_f(0), \tilde{p}_f(1), \dots, \tilde{p}_f(2^r-1)$, 将 2^r 个线性组合的线性偏差相加, 得到 $p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=0)$;

[0015] 再令 $\xi_i=1$ ($i=0, 1, \dots, n-1$), 代入 L_0, \dots, L_{r-1} , 计算出 2^r 个线性组合的线性偏差 $\tilde{p}_f(0), \tilde{p}_f(1), \dots, \tilde{p}_f(2^r-1)$, 将 2^r 个线性组合的 2^r 个线性偏差相加, 得到 $p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=1)$;

[0016] 其中, $p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=0)$ 和 $p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=1)$ 分别表示 $L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=0$ 和 $L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=1$ 出现的概率;

[0017] (3) 每个比特的输出线性偏差为每个比特满足r个线性校验关系式 $L_0=0, \dots, L_{r-1}=0$ 条件下的线性偏差 $\tilde{p}(\xi_i | L_0=0, \dots, L_{r-1}=0)$, 简记为 $\tilde{p}(\xi_i | L)$;

[0018]
$$\tilde{p}(\xi_i | L) = \frac{p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=0) - p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=1)}{p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=0) + p(L_0=0, L_1=0, \dots, L_{r-1}=0, \xi_i=1)}$$

[0019] 同时, 提供一种乘积码译码方法, 乘积码 $D = C_1 \otimes C_2$, 每行是参数为 (n_2, k_2, δ_2) 的线性分组码 C_2 , 每列是参数为 (n_1, k_1, δ_1) 的线性分组码 C_1 ; 对于乘积码 $D = C_1 \otimes C_2$ 采用行、列迭代方法进行译码, 迭代译码方法为:

[0020] (一), 列译码: 乘积码共 n_2 列, 每列是参数为 (n_1, k_1, δ_1) 线性分组码, 采用所述的线性分组码输出线性偏差计算方法计算出每个比特的输出线性偏差, 在此记作 $\tilde{p}_2(\xi_i)$; 将每个比特的输出线性偏差 $\tilde{p}_2(\xi_i)$ 与输入线性偏差 $\tilde{p}_1(\xi_i)$ 按方程式 $\tilde{p}(\xi_i) = \frac{\tilde{p}_1(\xi_i) + \tilde{p}_2(\xi_i)}{1 + \tilde{p}_1(\xi_i)\tilde{p}_2(\xi_i)}$ 进行混合运算, 得到更新的输出线性偏差 $\tilde{p}(\xi_i)$;

[0021] (二), 行译码: 乘积码共 n_1 行, 每行是参数为 (n_2, k_2, δ_2) 线性分组码, 将列译码得到的每个比特的更新的输出线性偏差 $\tilde{p}(\xi_i)$ 作为行译码的输入线性偏差, 记作 $\tilde{p}'_1(\xi_i)$, 采用所述的线性分组码输出线性偏差计算方法计算出每个比特的输出线性偏差, 在此记作 $\tilde{p}'_2(\xi_i)$; 将每个比特的输出线性偏差 $\tilde{p}'_2(\xi_i)$ 与输入线性偏差 $\tilde{p}'_1(\xi_i)$ 按方程式 $\tilde{p}'(\xi_i) = \frac{\tilde{p}'_1(\xi_i) + \tilde{p}'_2(\xi_i)}{1 + \tilde{p}'_1(\xi_i)\tilde{p}'_2(\xi_i)}$ 进行混合运

算, 得到更新的输出线性偏差 $\tilde{p}'(\xi_i)$; 一次迭代结束;

[0022] (三), 对乘积码 $D = C_1 \otimes C_2$ 进行数次迭代, 当最终的输出线性偏差收敛到+1或-1时, 迭代结束。

[0023] 本发明通过输入线性偏差和堆积引理计算校验全空间所有组合的校验关系式的线性偏差, 进而通过沃尔什逆变换求出每个比特的输出线性偏差, 和现有技术相比, 对相同数据进行译码, 译码效果更好, 比特错误率低于现有译码方法; 通过行、列迭代方法中的混合运算方程式对数据中的错误比特进行修正, 降低了译码错误率, 提高了译码性能。

附图说明

- [0024] 图1为乘积码 $D = C_1 \otimes C_2$ 的构造示意图；
 [0025] 图2为本发明基于线性分组码的乘积码译码方法的流程框图；
 [0026] 图3为本发明基于迭代的乘积码译码方法的流程框图；
 [0027] 图4为本发明基于线性分组码的乘积码译码方法与次优方法译码性能比较曲线图；
 [0028] 图5为本发明基于迭代的乘积码译码方法与次优方法译码性能比较曲线图。

具体实施方式

[0029] 下面结合附图和具体实施例对本发明的内容做进一步详细解释与描述。

[0030] 结合图2,一种乘积码译码方法,乘积码由线性分组码构建,对于参数为 (n, k) 的线性分组码,线性分组码的每个码字为 n 比特向量 $\xi = (\xi_0, \xi_1, \dots, \xi_{n-1})$,校验位有 $r = n - k$ 比特,即有 r 个校验方程, r 个校验方程都为线性校验关系式,记为 $L_0 = 0, \dots, L_{r-1} = 0$, $\tilde{p}(\xi_i)$ 为随机变量 $\xi_0, \xi_1, \dots, \xi_{n-1}$ 的输入线性偏差,线性分组码输出线性偏差计算如下步骤:

[0031] 步骤一、由 r 个校验方程生成 2^r 个组合校验关系式 $f_c(\xi)$:

[0032] 每个校验方程都是对随机变量 $\xi_0, \xi_1, \dots, \xi_{n-1}$ 的一种线性组合,得到 r 个随机变量 L_0, \dots, L_{r-1} ,对随机变量 L_0, \dots, L_{r-1} 进行校验全空间的线性组合,共得到 2^r 个线性组合,对应着 2^r 个组合校验关系式,记为 $f_c(\xi)$,其中 c 为线性系数状态, $c = (c_0, c_1, \dots, c_{r-1})$, $c \in [0, 2^r - 1]$;

[0033] 步骤二、由输入线性偏差 $\tilde{p}(\xi_i) = p(\xi_i = 0) - p(\xi_i = 1), i = 0, 1, \dots, n-1$ 和堆积引理计算 2^r 个组合校验关系式的线性偏差 $\tilde{p}_f(c)$, $\tilde{p}_f(c) = \tilde{p}_f(c_0, c_1, \dots, c_{r-1}) = \tilde{p}_f(c_0 L_0 \oplus c_1 L_1 \oplus \dots \oplus c_{r-1} L_{r-1})$

$\square \tilde{p}(d_0 \xi_0 \oplus d_1 \xi_1 \oplus \dots \oplus d_{n-1} \xi_{n-1}) = \prod_{d_i=1, i \in [0, n-1]} \tilde{p}(\xi_i)$,其中 \square 表示随着线性系数状态 $c = (c_0, c_1, \dots, c_{r-1})$ 的变化,随机变量 $\xi_0, \xi_1, \dots, \xi_{n-1}$ 的组合系数也将产生变化, $\tilde{p}_f(c)$ 的线性组合有 2^r 个,记为 $\tilde{p}_f(0), \tilde{p}_f(1), \dots, \tilde{p}_f(2^r - 1)$;

[0034] 步骤三、计算每个比特的输出线性偏差:

[0035] (1) 由沃尔什逆变换得, $p(0) = 2^{-r} \sum_{c=0}^{2^r-1} \tilde{p}_f(c)$;

[0036] 其中, $p(0)$ 表示 $L_0 = 0, \dots, L_{r-1} = 0$ 出现的概率, $p(0) = p(L_0 = 0, L_1 = 0, \dots, L_{r-1} = 0)$;

[0037] (2) $p(L_0 = 0, L_1 = 0, \dots, L_{r-1} = 0, \xi_i = 0)$ 、 $p(L_0 = 0, L_1 = 0, \dots, L_{r-1} = 0, \xi_i = 1)$;

[0038] 令 $\xi_i = 0 (i = 0, 1, \dots, n-1)$,代入 L_0, \dots, L_{r-1} ,计算出 2^r 个线性组合的线性偏差 $\tilde{p}_f(0), \tilde{p}_f(1), \dots, \tilde{p}_f(2^r - 1)$,将 2^r 个线性组合的线性偏差相加,得到 $p(L_0 = 0, L_1 = 0, \dots, L_{r-1} = 0, \xi_i = 0)$;

[0039] 再令 $\xi_i = 1 (i = 0, 1, \dots, n-1)$,代入 L_0, \dots, L_{r-1} ,计算出 2^r 个线性组合的线性偏差 $\tilde{p}_f(0), \tilde{p}_f(1), \dots, \tilde{p}_f(2^r - 1)$,将 2^r 个线性组合的 2^r 个线性偏差相加,得到 $p(L_0 = 0, L_1 = 0, \dots, L_{r-1} = 0, \xi_i = 1)$;

[0040] 其中, $p(L_0 = 0, L_1 = 0, \dots, L_{r-1} = 0, \xi_i = 0)$ 和 $p(L_0 = 0, L_1 = 0, \dots, L_{r-1} = 0, \xi_i = 1)$ 分别表示 $L_0 = 0, L_1 = 0, \dots, L_{r-1} = 0, \xi_i = 0$ 和 $L_0 = 0, L_1 = 0, \dots, L_{r-1} = 0, \xi_i = 1$ 出现的概率;

[0041] (3) 每个比特的输出线性偏差为每个比特满足 r 个线性校验关系式 $L_0 = 0, \dots, L_{r-1}$

=0条件下的线性偏差 $\tilde{p}(\xi_i | L_0 = 0, \dots, L_{r-1} = 0)$, 简记为 $\tilde{p}(\xi_i | L)$:

$$[0042] \quad \tilde{p}(\xi_i | L) = \frac{p(L_0 = 0, L_1 = 0, \dots, L_{r-1} = 0, \xi_i = 0) - p(L_0 = 0, L_1 = 0, \dots, L_{r-1} = 0, \xi_i = 1)}{p(L_0 = 0, L_1 = 0, \dots, L_{r-1} = 0, \xi_i = 0) + p(L_0 = 0, L_1 = 0, \dots, L_{r-1} = 0, \xi_i = 1)}.$$

[0043] 一种乘积码译码方法, 乘积码 $D = C_1 \otimes C_2$, 每行是参数为 (n_2, k_2, δ_2) 的线性分组码 C_2 , 每列是参数为 (n_1, k_1, δ_1) 的线性分组码 C_1 ; 对于乘积码 $D = C_1 \otimes C_2$ 采用行、列迭代方法进行译码, 迭代译码方法为:

[0044] (一), 列译码: 乘积码共 n_2 列, 每列是参数为 (n_1, k_1, δ_1) 线性分组码, 采用分组码输出线性偏差计算方法计算出每个比特的输出线性偏差, 在此记作 $\tilde{p}_2(\xi_i)$; 将每个比特的输出线性偏差 $\tilde{p}_2(\xi_i)$ 与输入线性偏差 $\tilde{p}_1(\xi_i)$ 按方程式 $\tilde{p}(\xi_i) = \frac{\tilde{p}_1(\xi_i) + \tilde{p}_2(\xi_i)}{1 + \tilde{p}_1(\xi_i)\tilde{p}_2(\xi_i)}$ 进行混合运算, 得到更新的输出线性偏差 $\tilde{p}(\xi_i)$:

[0045] (二), 行译码: 乘积码共 n_1 行, 每行是参数为 (n_2, k_2, δ_2) 线性分组码, 将列译码得到的每个比特的更新的输出线性偏差 $\tilde{p}(\xi_i)$ 作为行译码的输入线性偏差, 记作 $\tilde{p}'_1(\xi_i)$, 采用所述的线性分组码输出线性偏差计算方法计算出每个比特的输出线性偏差, 在此记作 $\tilde{p}'_2(\xi_i)$; 将每个比特的输出线性偏差 $\tilde{p}'_2(\xi_i)$ 与输入线性偏差 $\tilde{p}'_1(\xi_i)$ 按方程式 $\tilde{p}'(\xi_i) = \frac{\tilde{p}'_1(\xi_i) + \tilde{p}'_2(\xi_i)}{1 + \tilde{p}'_1(\xi_i)\tilde{p}'_2(\xi_i)}$ 进行混合运算, 得到更新的输出线性偏差 $\tilde{p}'(\xi_i)$; 一次迭代结束;

[0046] (三), 对乘积码 $D = C_1 \otimes C_2$ 进行数次迭代, 当最终的输出线性偏差收敛到+1或-1时, 迭代结束。

[0047] 进一步, 为便于理解, 结合图1和图3, 对图2中计算出来的输出线性偏差进行迭代译码, 由线性分组码构建的乘积码 $D = C_1 \otimes C_2$, 每行是参数为 (n_2, k_2, δ_2) 的线性分组码 C_2 , 每列是参数为 (n_1, k_1, δ_1) 的线性分组码 C_1 。

[0048] 第一步、列译码:

[0049] 每列是参数为 (n_1, k_1, δ_1) 的线性分组码 C_1 , 共 n_2 列, 设第 j 列为 $C_j = (\xi_{0,j}, \xi_{1,j}, \dots, \xi_{n_1-1,j})$, 按 $j=0, 1, \dots, n_2-1$ 的顺序进行下列操作:

[0050] 由每个比特的输入线性偏差 $\tilde{P}(\xi_{0,j}), \dots, \tilde{P}(\xi_{n_1-1,j})$, 根据分组码输出线性偏差计算方法计算出每个比特的输出线性偏差 $\tilde{P}(\xi_{0,j}|L), \dots, \tilde{P}(\xi_{n_1-1,j}|L)$, 然后将 $\tilde{P}(\xi_{0,j}), \dots, \tilde{P}(\xi_{n_1-1,j})$ 代替 $\tilde{p}_1(\xi_i)$, $\tilde{P}(\xi_{0,j}|L), \dots, \tilde{P}(\xi_{n_1-1,j}|L)$ 代替 $\tilde{p}_2(\xi_i)$, 根据列译码的混合运算方程式 $\tilde{p}(\xi_i) = \frac{\tilde{p}_1(\xi_i) + \tilde{p}_2(\xi_i)}{1 + \tilde{p}_1(\xi_i)\tilde{p}_2(\xi_i)}$ 计算出新的输出线性偏差 $\tilde{p}(\xi_i)$, $\tilde{p}(\xi_i)$ 记作 $\tilde{P}^*(\xi_{0,j}), \dots, \tilde{P}^*(\xi_{n_1-1,j})$, $j=0, 1, \dots, n_2-1$ 。

[0051] 第二步、行译码:

[0052] 每行是参数为 (n_2, k_2, δ_2) 的线性分组码 C_2 , 共 n_1 行, 设第 i 行为 $C_i = (\xi_{i,0}, \xi_{i,1}, \dots, \xi_{i,n_2-1})$, 按 $i=0, 1, \dots, n_1-1$ 的顺序进行操作:

[0053] 将由列译码得到的输出线性偏差 $\tilde{P}^*(\xi_{0,j}|L), \dots, \tilde{P}^*(\xi_{n_1-1,j}|L)$, $j=0, 1, \dots, n_2-1$ 作为行译码的输入线性偏差, 记作 $\tilde{P}(\xi_{i,0}), \dots, \tilde{P}(\xi_{i,n_2-1})$, 根据分组码输出线性偏差计算方法计算出每个比

	$C_7^0 = 1$ 个: $0 \cdot L_0 \oplus 0 \cdot L_1 \oplus \dots \oplus 0 \cdot L_6 = 0$	线性系数状态: $(0, \dots, 0)$
	$C_7^1 = 7$ 个: $L_i, i=0,1,\dots,6$	线性系数状态: 第 i 位置为 1, 其它为 0
[0089]	$C_7^2 = 21$ 个: $L_i \oplus L_j, i, j \in [0,6], i \neq j$	线性系数状态: 第 i, j 位置为 1, 其它为 0
	$C_7^3 = 35$ 个: $L_i \oplus L_j \oplus L_k, i, j, k \in [0,6], i \neq j \neq k$	线性系数状态: 第 i, j, k 位置为 1, 其它为 0

	$C_7^6 = 1$ 个: $L_0 \oplus L_1 \oplus \dots \oplus L_6$	线性系数状态 $(1, \dots, 1)$

[0090] 每种线性组合对应一个线性系数状态,所有线性系数状态构成一个7维空间。设线性系数状态为 (c_0, c_1, \dots, c_6) ,由堆积引理可以将这128个线性组合的线性偏差计算出来,分别记为 $\tilde{p}_f(c_0, c_1, \dots, c_6)$ 。

[0091] 由沃尔什变换定义得知, $\tilde{p}_f(c_0, c_1, \dots, c_6), (c_0, c_1, \dots, c_6) \in [0, 127]$ 为函数 $f(L_0, \dots, L_6) = N(L_0, \dots, L_6)$ 的沃尔什谱。

[0092] 因为向量 (L_0, L_1, \dots, L_6) 与二进制数 $L_0L_1 \dots L_6$ 一一对应,所以 $p(L_0, L_1, \dots, L_6)$ 可以表示为 $p(L_0L_1 \dots L_6), L_0L_1 \dots L_6 \in [0, 127]$, $\tilde{p}_f(c_0, c_1, \dots, c_6)$ 表示为 $\tilde{p}_f(c_0c_1 \dots c_6), (c_0c_1 \dots c_6) \in [0, 127]$ 。

[0093] $p(0), p(1), \dots, p(127)$ 的值可以由 $\tilde{p}_f(0), \tilde{p}_f(1), \dots, \tilde{p}_f(127)$ 经沃尔什逆变换得到:

$$[p(0), p(1), \dots, p(127)] = 2^{-7} \begin{bmatrix} \tilde{p}_f(0), \tilde{p}_f(1), \dots, \tilde{p}_f(127) \end{bmatrix} \cdot H_{127} \quad (10)$$

[0095] 即通过堆积引理,计算出所有线性系数对应的线性线性偏差表,由线性线性偏差表通过逆Hadamard变换,得到状态概率分布表。

[0096] 因为每个码字都要满足7个线性校验关系式,即 $L_0=0, L_1=0, \dots, L_6=0$,所以只需要求出 $p(L_0=0, L_1=0, \dots, L_6=0) = p(0)$ 的值。

[0097] 利用Hadamard矩阵第1行第1列为全‘1’的特点,容易推导出:

$$p(0) = 2^{-7} \sum_{c=0}^{127} \tilde{p}_f(c) \quad (11)$$

[0099] 利用方程式(11),可以快速计算出 $p(0)$ 的值。

[0100] 有了 $p(0)$ 值,就可以精确计算出BCH码中每个比特的新线性偏差,具体步骤如下:对每个比特 $\xi_i, i=0, 1, \dots, 63$

[0101] (1) 先令 $\xi_i=0$,代入 L_0, \dots, L_6 ,计算出128种线性组合的线性偏差 $\tilde{p}_f(0), \tilde{p}_f(1), \dots, \tilde{p}_f(127)$, 128个线性偏差相加,得到 $p(L_0=0, L_1=0, \dots, L_{127}=0, \xi_i=0)$ 。

[0102] (2) 再令 $\xi_i=1$,代入 L_0, \dots, L_6 ,计算出128种线性组合的线性偏差 $\tilde{p}_f(0), \tilde{p}_f(1), \dots, \tilde{p}_f(127)$, 128个线性偏差相加,得到 $p(L_0=0, L_1=0, \dots, L_{127}=0, \xi_i=1)$ 。

[0103] (3) $\xi_i, i=0, 1, \dots, 63$ 在满足线性校验关系条件下的线性偏差为

$$[0104] \tilde{p}(\xi_i | L) = \frac{p(L_0=0, L_1=0, \dots, L_{127}=0, \xi_i=0) - p(L_0=0, L_1=0, \dots, L_{127}=0, \xi_i=1)}{p(L_0=0, L_1=0, \dots, L_{127}=0, \xi_i=0) + p(L_0=0, L_1=0, \dots, L_{127}=0, \xi_i=1)}$$

[0105] 下述提供一种简便算法计算来每个比特的输出线性偏差:

[0106] 根据每个比特的原始输入偏差,首先将所有线性组合校验关系式的线性偏差计算出来,分别记为 $\tilde{p}_f(0), \dots, \tilde{p}_f(2^r - 1)$ 。

[0107] 计算第*i*个比特的线性偏差时,不含有 ξ_i 的组合校验关系式的线性偏差保持不变,含有 ξ_i 的组合校验关系式的线性偏差除以 $\tilde{p}(\xi_i)$,再求 2^r 个组合校验关系式的线性偏差的均值,就得到 $p(L_0=0, \dots, L_{r-1}=0, \xi_i=0)$,记作sum0。

[0108] 类似地,得到 $p(L_0=0, \dots, L_{r-1}=0, \xi_i=1)$,记作sum1。

[0109] 每个比特的输出线性偏差为每个比特满足 r 个线性校验关系式 $L_0=0, \dots, L_{r-1}=0$

条件下的线性偏差 $\tilde{p}(\xi_i | L_0=0, \dots, L_{r-1}=0)$,计算公式为:
$$\tilde{p}(\xi_i | L_0=0, \dots, L_{r-1}=0) = \frac{\text{sum0} - \text{sum1}}{\text{sum0} + \text{sum1}}$$

$$= \frac{p(L_0=0, \dots, L_{r-1}=0, \xi_i=0) - p(L_0=0, \dots, L_{r-1}=0, \xi_i=1)}{p(L_0=0, \dots, L_{r-1}=0, \xi_i=0) + p(L_0=0, \dots, L_{r-1}=0, \xi_i=1)}$$
 简便算法的益处是,计算*n*个比特

的输出线性偏差,即: $i=0, \dots, n-1$,每次计算时,仅在原始组合校验关系式的线性偏差上稍微进行修改,不必重新计算每个组合校验关系式的线性偏差,大大减少了计算复杂度,提高了效率。

[0110] 对于TPC2944,每行每列都是一个BCH码。每次迭代,先对列进行操作,共64列,每列为46比特BCH码,分别按照上述步骤计算出每个比特的输出线性偏差;输出线性偏差与原始输入线性偏差进行混合运算,得到更新的线性偏差。再对行进行操作,列更新后的线性偏差作为行输入线性偏差,共46行,每行为64比特BCH码,分别按照上述步骤计算出行输出线性偏差;每个比特的输出线性偏差与输入线性偏差进行混合运算,得到更新的线性偏差,第一次迭代结束。第一次迭代结束后的更新的线性偏差作为第二次迭代的输入线性偏差,进入第二次迭代。迭代几次后,输出线性偏差收敛到+1或-1,迭代结束。输出线性偏差+1对应比特0,输出线性偏差-1对应比特1,实现了信道纠错。

[0111] 本发明译码方法与次优方法纠错能力比较

[0112] 步骤1:产生随机噪声源,按照固定的信噪比,使得一帧数据中每个比特位置按照预定的随机概率发生,即获得先验概率已知的试验数据;

[0113] 步骤2:分别用本发明译码方法和次优方法进行迭代纠错译码,迭代次数为6次;

[0114] 步骤3:译码后分别记录两种方法的比特错误率,具体结果如表1所示:

[0115]

信噪比(dB)	比特错误率	数据个数	次优方法	本发明的译码
2.400	0.03113990	990000	0.03019311	0.03333152
2.500	0.02965516	990000	0.02862351	0.02758300
2.600	0.02821347	990000	0.02684777	0.02139135
2.700	0.02681495	990000	0.02485885	0.01352875
2.800	0.02545960	990000	0.02302976	0.00804442
2.900	0.02414741	990000	0.02054338	0.00513512
3.000	0.02287829	990000	0.01564942	0.00246082
3.100	0.02165213	990000	0.01011910	0.00105870
3.200	0.02046874	990000	0.00654625	0.00059235
3.300	0.01932787	990000	0.00312209	0.00027132
3.400	0.01822922	990000	0.00156866	0.00016149
3.500	0.01717244	990000	0.00073890	0.00008776
3.600	0.01615715	990000	0.00030981	0.00004701

[0116]

3.700	0.01518286	990000	0.00010410	0.00002777
3.800	0.01424908	990000	0.00002726	0.00001590
3.900	0.01335524	990000	0.00001706	0.00001067
4.000	0.01250075	990000	0.00000371	0.00000305
4.100	0.01168492	990000	0.00000186	0.00000294
4.200	0.01090708	990000	0.00000062	0.00000165
4.300	0.01016647	990000	0.00000028	0.00000111
4.400	0.00946230	990000	0.00000000	0.00000092
4.500	0.00879375	990000	0.00000000	0.00000057
4.600	0.00815995	990000	0.00000000	0.00000000
4.700	0.00756000	990000	0.00000000	0.00000000
4.800	0.00699297	990000	0.00000000	0.00000000

[0117] 表1

[0118] 比较两种方法对同类数据的译码效果,如表1所示:当信噪比在2.5~4.0区间时,译码后,本发明的译码方法的比特错误率明显低于次优方法,即要达到相同的译码效果,本发明译码方法所需的信噪比更低。如译码后比特错误率要达到 2.7×10^{-5} ,次优方法需要信噪比为3.8dB,而本发明译码方法需要3.7dB,节省了0.1dB。

[0119] 结合图4,本发明译码方法的曲线在这个区间接近于一条直线,而次优方法呈现为

该直线外的弧线。当信噪比大于4.0dB时,次优方法收敛较快。当数据的含错率非常低(小于1.2%)时,次优方法译码效果较好,当数据的含错率较高(大于1.2%)且在信道编码的纠错能力范围内时,本发明译码方法效果更好。

[0120] 本发明译码方法与次优方法收敛情况比较

[0121] 步骤1:设置初始先验错误概率分别为0.014、0.018、0.020的TPC编码数据作为译码数据;

[0122] 步骤2:用本发明和次优方法进行纠错译码;

[0123] 步骤3:比较每次迭代后TPC编码的比特错误率。

[0124]

错误率	先验错误率为 0.014		先验错误率为 0.018		先验错误率为 0.020	
	本发明 译码方法	次优 方法	本发明 译码方法	次优 方法	本发明 译码方法	次优 方法
0	0.014	0.014	0.018	0.018	0.020	0.020
1	0.001	0.009	0.004	0.014	0.006	0.016
2	0.000	0.000	0.000	0.003	0.001	0.005

[0125]

3	0.000	0.000	0.000	0.000	0.001	0.001
---	-------	-------	-------	-------	-------	-------

[0126] 表2

[0127] 从表2和图5可以看出,本发明译码方法迭代1次后,比特的正确率提升1%以上,次优方法迭代1次后,比特的正确率提升0.4%左右。可知本发明的收敛速度更快。

[0128] 本发明译码方法与次优方法的计算复杂度比较

[0129] 本发明中一次迭代都是指一次行/列译码紧跟着一次列/行译码,所以整体复杂度由一次迭代的复杂度决定,进而由一次行/列译码的复杂度决定。即线性分组码的计算复杂度决定了乘积码的复杂度。

[0130] 在发明内容中,由本发明的计算方法可知,在计算 2^r 个组合校验关系的线性偏差时,每个组合校验关系含有 $n/2$ 项,因此计算 2^r-1 个组合校验关系的线性偏差需 $(2^r-1) \cdot (n/2-1)$ 次乘法运算;在计算所有比特的输出线性偏差时,由于与 ξ_i 相关的组合校验关系一般为 $(2^r-1)/2$ 个,因此需要 $n \cdot 2^{r+1}$ 次加法运算、 $n \cdot 2^r$ 次乘法运算。整体计算复杂度为 $(3 \cdot 2^{r-1} \cdot n - 2^r - n/2 + 1)$ 次乘法运算, $n \cdot 2^{r+1}$ 次加法运算。而次优方法的复杂度与最不可信点的个数 p 相关,随着 p 值的增大,呈 2^p 增长。以TPC 3/4编码为例,当 $p=7$ 时,经实际数据测算得知:迭代相同的次数,本发明所需时间是次优方法的1/2。

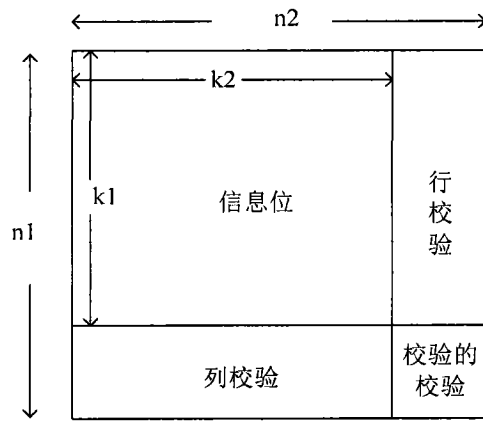


图1

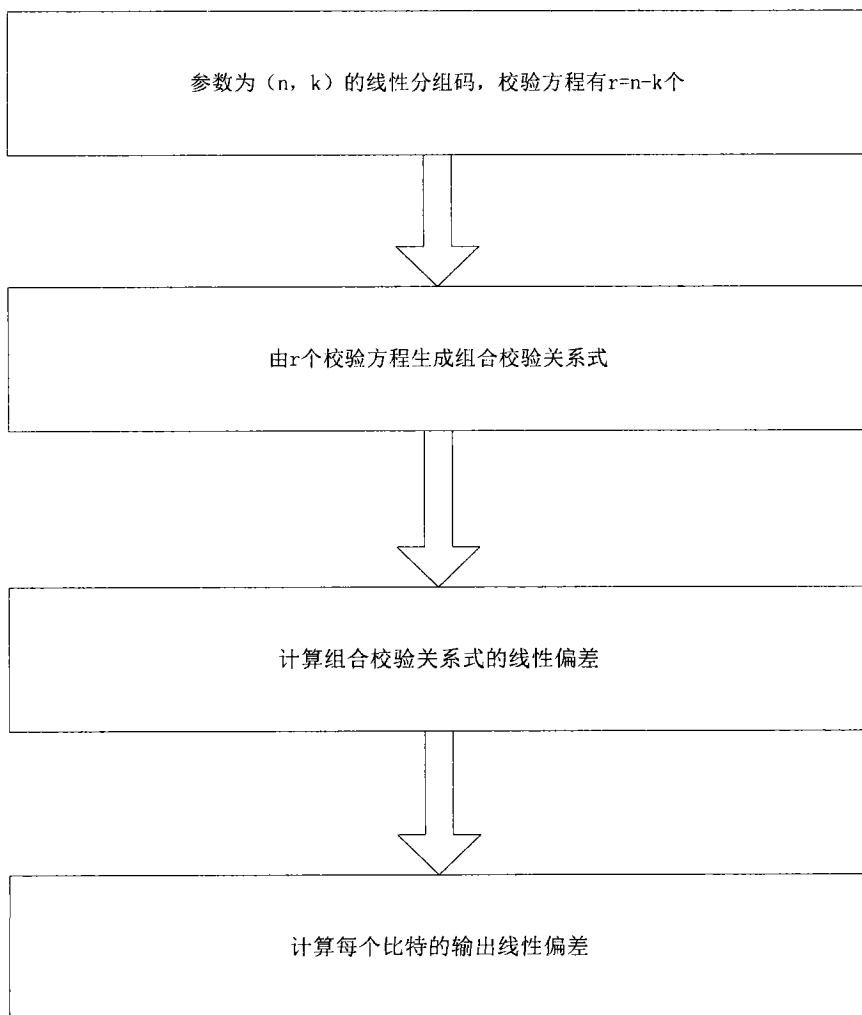


图2

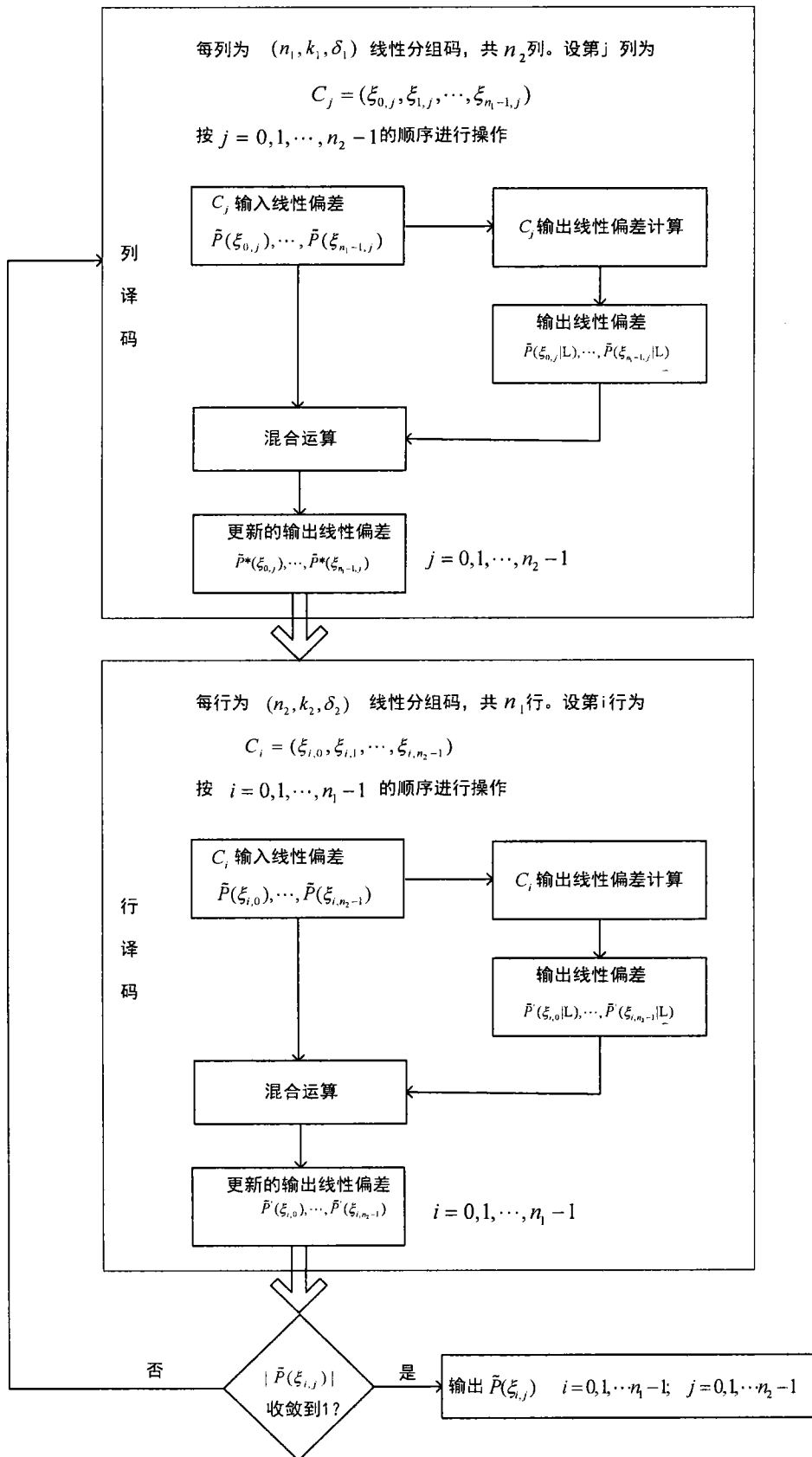


图3

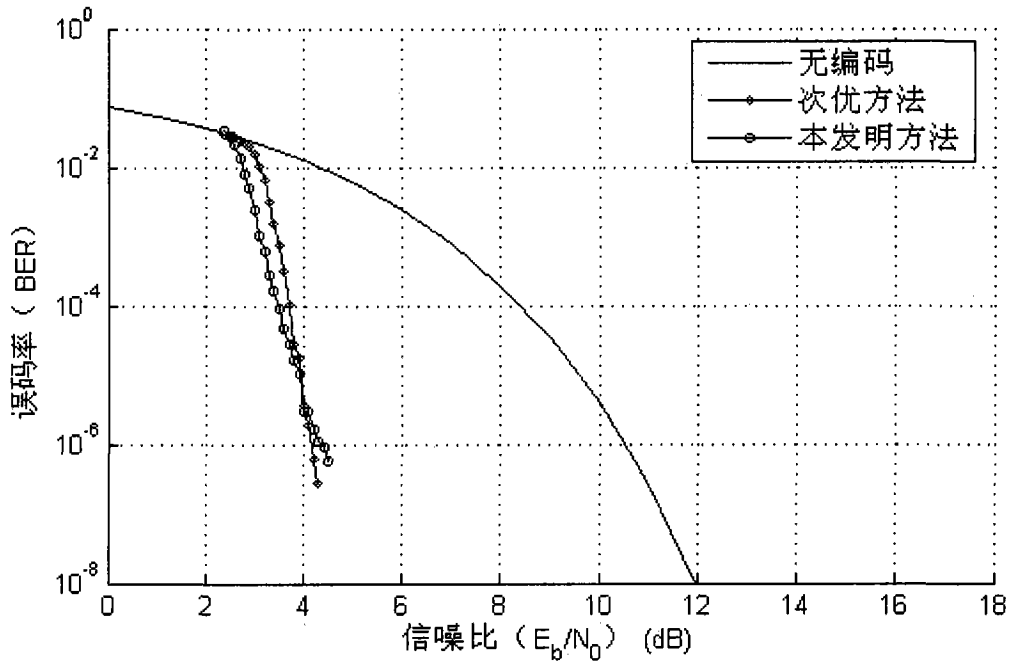


图4

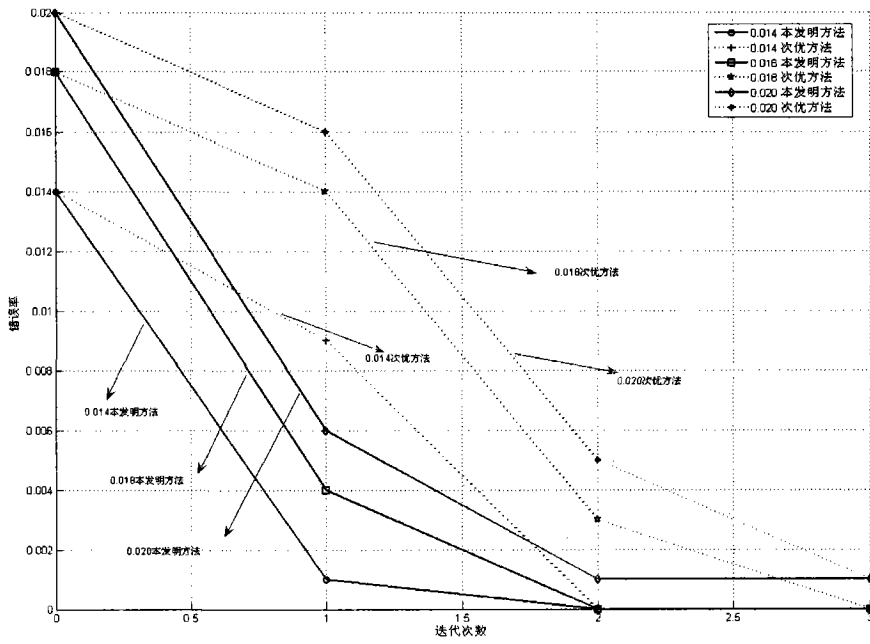


图5