* 1

# 一种抑制 LDPC 码突发错误的软件方法

野晓东，　马林华，　王卫民，　于　瑞
(空军工程大学　工程学院　航空电子工程系，陕西　西安　710038)

**摘　要**：对低密度奇偶校验码(Low - Density Parity - Check Codes，LDPC)在AWGN(Additive White Gaussian Noise)信道下的译码算法进行了深入研究，分析了在译码过程中出现突发错误的原因，指出出现这种错误是由于在校验矩阵中存在环路，并提出了一种抑制突发错误出现的软件方法。在该方法中，只需对LDPC码的译码程序进行适当控制，就可有效抑制由于环路影响而出现的突发错误，进一步提高了LDPC码的译码性能和译码程序的稳定性，扩大了LDPC码的应用空间。

**关键词**：低密度奇偶校验码；环路；突发错误
**DOI**：10.3969/j.issn.1009-3516.2009.03.018
**中图分类号**：TN911.22　　**文献标识码**：A　　**文章编号**：1009-3516(2009)03-0082-04

低密度奇偶校验码(Low - Density Parity - Check Codes，LDPC)具有逼近香农(Shannon)限的优良性质[1]，其理论研究和实际应用研究都具有重要意义。在文献[2]中Sae - Young Chung设计的接近香农限的码，在二进制输入高斯信道上与香农限只相差0.004 5 dB。低密度码的理论研究中，多数采用和积译码算法(Sum - Product Decoding Algorithm)，而此种算法在程序仿真过程中容易出现突发错误(incidental error)，一旦出现突发错误，将严重影响LDPC码的译码性能，从而限制了LDPC码的应用空间。

本文具体分析了突发错误产生的原因，指出之所以产生突发错误是因为校验矩阵中存在环路，对于环路常用的方法是对校验矩阵进行优化，删去短环，从而提高LDPC码的译码性能[3-5]。但是校验矩阵优化到一定程度后就不容易再优化了，而且长环还存在于矩阵中，影响着LDPC码性能的提高和稳定性。

## 1　LDPC码的环路与突发错误

### 1.1　LDPC码环路分析

1981年，Tanner提出了采用Tanner二部图来表示LDPC码，二部图被称作Tanner图[6]。如图1(a)所示，由Tanner图中的某个节点出发，经过不同的连续的节点，最后可返回所选的起始节点，则该闭合路径即构成Tanner图中的一个环(cycle)，环上的节点数或边数称为环长，图中所有环的最小长度称为围长(girth)。图1(a)中虚线边和连接的节点构成长度为4的环：$x6 -> z1 -> x8 -> z4 -> x6$，4也是此Tanner图的围长。此围长至少为4，且必为偶数[7]。

Tanner图中长度为4的环是很容易从校验矩阵直接地检测出来，以校验矩阵的非零元素“1”为一个顶点，则校验矩阵中的一个矩形就是一个周长为4的环，如图1(b)中虚线画出了校验矩阵中的一个矩形，对应于Tanner图中虚线连接的一个环[8]。

由和积译码算法的迭代过程可知，当图中存在环路时，参与迭代运算的信息不满足独立性，环的存在使得译码不满足最优性。因此，Tanner图中的环是影响LDPC码性能的重要因素之一[9]。

* **收稿日期**：2008-06-18
**作者简介**：野晓东(1982-)，男，河北武安人，硕士生，主要从事抗干扰通信研究：E-mail：student_yxd@163.com
马林华(1965-)，男，陕西汉中人，教授，主要从事编码理论、抗干扰通信、宽带视频通信研究.

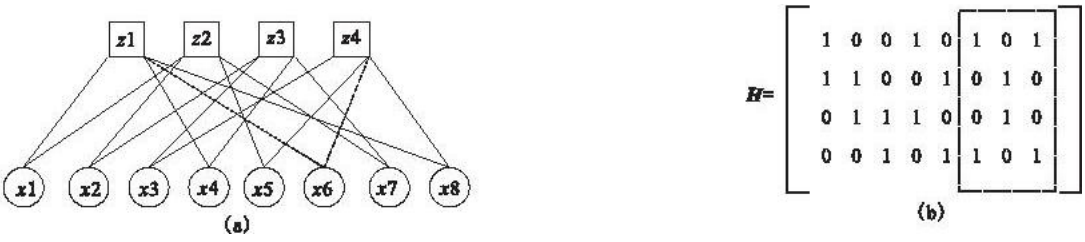


图 1　(8,2,4) LDPC 码的校验矩阵和 Tanner 图

Fig. 1　Parity – check matrix and Tanner graph for LDPC codes of (8, 2, 4)

和积(Sum – Product)算法是 LDPC 码最常用的译码算法,其“和(Sum)”指每个校验节点向相邻变量节点发送的消息,是对该校验节点的其他相邻变量节点所发送消息的求和运算;“积(Product)”指每个变量节点向相邻校验节点发送的消息,是对该变量节点的其他相邻校验节点所发送消息的求积运算[10]。

### 1.2　突发错误原因分析

对接收到的信号按概率测度下的和积译码算法进行迭代译码时,会不定时地出现一些突发错误,这些突发错误往往是集中在一个码组中,这个码组基本上是全部错误。刚遇到突发错误的时候,认为是高斯噪声模型不合格,但经过多次仿真验证,证明噪声模型是完全正确的,最后确定错误是由校验矩阵中的环路引起的。通过详细研究译码程序和反复的仿真捕捉,发现出现突发错误的码组在迭代译码的过程中都呈现相同的变化规律,见图 2。

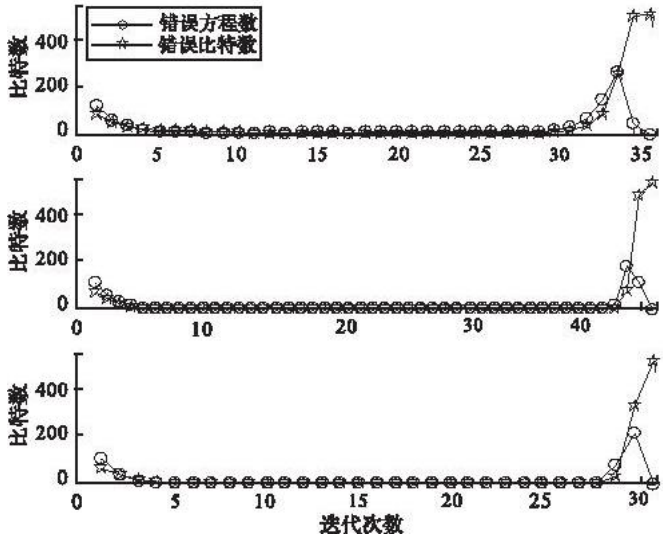


图 2　$N=1\ 024$ 非正则码突发错误产生示意图

Fig. 2　Graph for incidental error appearing of $N=1\ 024$ irregular codes

由图 2 可以看出,发生突发错误的 3 个码组,它们都是在跳出迭代循环的时候 $\boldsymbol{H}_{M\times N}\hat{x}=0$ 是成立的,但是最终的结果却是每个码组都错了几百个比特位,有一半信息都是错误的。正常的码组在迭代几次以后基本上就全部正确了,而出现突发错误的码组都是迭代三四十次才能跳出循环。在它们跳出循环前,错误的校验方程数有个最小值,错误的校验方程数在此最小值附近振荡一段时间后,就会突然增加,然后陡然降至 0 结束迭代循环,但是译码后的信息有一半左右是错误的。

因为本文所用的校验矩阵是消除了短环,经过优化而选出来的性能比较好的矩阵,所以 LDPC 码的去环优化算法对这种突发错误往往无法解决。

用 Tanner 图来解释突发错误产生的过程很好理解,如图 3 所示,$z$ 代表校验消息,$x$ 代表变量消息。假设校验矩阵中存在一个 4 环,如图 3 中 $z1->x1->z2->x2->z1$,由于噪声的干扰,使 $x1$ 和 $x2$ 都判决错误了,因为它们在一个环中,所以它们所在的校验方程都是成立的。随着迭代次数的增加,更加确信了 $x1$ 和 $x2$ 的错误值,因为 $x2$ 和 $z3$ 相连,所以 $x2$ 的错误值又影响了 $z3$ 值的判定。为了使 $z3$ 这个方程成立,经过几轮迭代运算,将与 $z3$ 相连的置信度最低的 $x4$ 的值判错了。由于 $x4$ 值的错判,又影响了 $z4$ 方程的成立,为了使 $z4$ 成立,又经过几轮迭代运算,将与 $z4$ 相连的置信度最低的 $x9$ 的值也判错了。

如此进行下去,随着迭代次数的增加,引起了一连串的链锁反应,使错误的信息不断传播,最终在错误信息的基础上达到了一种新的错误平衡,使 $\boldsymbol{H}_{M\times N}\hat{x}=0$ 成立,跳出迭代循环。但实际译码结果却有一半信息是错误的,可以说这个码组的传输彻底失败了。这就是突发错误产生的整个过程。

实际迭代运算时,可能同时将多个变量消息判错。一旦出现突发错误,误码率就会陡然上升,在短时间内很难降下来,严重影响着 LDPC 码译码性能的进一步提高。

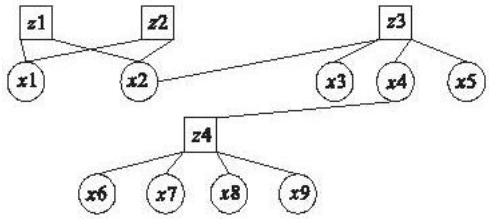


图 3　突发错误产生示意图

Fig. 3　Tanner graph for incidental error appearing

## 2 抑制突发错误的程序设计

由前一节的分析可知,正常的码组在迭代几次以后基本上就全部正确了,而出现突发错误的码组都是迭代三四十次才能跳出循环。经过研究发现,在它们跳出循环前,错误的校验方程数有个最小值,在此最小值下,码组错误的比特数也比较少。由于现有的 LDPC 码矩阵优化算法对此种突发错误也束手无策,所以就设想是否可以通过软件方法来对译码算法的程序进行优化,使其能够有效抑制突发错误的出现。大量的仿真结果表明,用软件方法来抑制突发错误的出现,是非常有效的。

在译码模块中,对接收到的每一个码组都进行迭代运算,在每次迭代运算后进行一次译码尝试,并检验 $\boldsymbol{H}_{M\times N}\hat{x}=0$ 是否成立。每次译码尝试都用一个变量 countx 来记录校验方程错误的个数,数组 vhat[ ]用来保存译码尝试后得到的码组。若 countx = =0 则跳出循环,宣布译码成功,否则进行新一轮的迭代运算,直至译码成功或达到最大迭代次数时 $\boldsymbol{H}_{M\times N}\hat{x}\neq 0$,强制跳出循环,宣布译码失败。迭代译码结束后,有用的信息存储在数组 vhat[ ]中。

软件方法抑制突发错误的基本思想是:在 LDPC 码的编译码程序中设计一段程序加在译码模块的迭代译码循环中,变量 countx_min 记录 countx 的最小值,countx_min 的初值是校验方程的个数,数组 vhat_min[ ]记录与 countx_min 相对应的码组。每次译码尝试后,都要让 countx 与 countx_min 进行比较,若 countx 小于 countx_min,则将 countx 的值赋给 countx_min,相应的码组 vhat[ ]的值赋给 vhat_min[ ]。设定一个差值 num,用来表示在迭代运算过程中出现错误方程最小个数后,每次译码尝试后 countx 与 countx_min 的最大差值。令 num = 100,若 countx > = countx_min + num,则认为此组码字在迭代译码过程中出现了突发错误,将 vhat_min[ ]的值赋给此次译码尝试后得到的码组 vhat[ ],跳出循环,宣布译码结束,否则认为未出现突发错误,按正常的译码规则进行新一轮的迭代运算。

对于差值 num 的取值要根据实际的情况而定。对于码长 $N=1\ 024$,码率 $R=1/2$ 的 LDPC 码,num = 100 是比较合适的。num 取值过大,则迭代循环所用时间比较长;num 取值太小,可能不能有效抑制突发错误的出现。在实际应用中需要在权衡时间和精度之后设定一个适当的 num 值。

## 3 性能仿真与分析

未对突发错误进行抑制的和积译码算法在较高信噪比下的译码性能比较好,而且性能比较稳定,误码率都可达到 $10^{-6}$ 数量级,有时可达到 $10^{-7}$ 数量级,甚至在 10M 数据情况下能够完全正确译码。但是此种算法在较低信噪比下的译码性能却不太稳定,例如在信噪比为 2.5 dB 时,有时误码率可达到 $10^{-6}$ 数量级,一旦出现突发错误,误码率就降到了 $10^{-5}$ 数量级。对和积译码算法程序加上抑制突发错误的程序后,其译码性能无论是在高信噪比下还是在低信噪比下都可取得比较好的结果,并且译码性能比较稳定。本文取码率 $R=1/2$,码长 $N=1\ 024$,度序列分布为:

$$\begin{aligned}\lambda(x) &= 0.276\ 84x + 0.283\ 42x^2 + 0.439\ 74x^8\\ \rho(x) &= 0.015\ 68x^5 + 0.852\ 44x^6 + 0.131\ 88x^7\end{aligned} \tag{1}$$

非规则 LDPC 码进行研究,在平稳无记忆 AWGN 信道,BPSK 调制下,对概率测度下的和积译码算法进行仿真。和积译码算法的最大迭代次数为 50,每个信噪比下统计 10 000 个分组,图 4 为误码率比较结果。

由图 4 可以看出,经过抑制突发错误的和积译码算法在信噪比为 2 dB 时,误码率可达到 $10^{-5}$ 数量级,信噪比为 2.5 dB 时,误码率可达到 $10^{-6}$ 数量级,并且性能相当稳定。与未加抑制突发错误程序的和积译码算法相比,在低信噪比下译码性能提高了 0.5 dB 左右。大量仿真结果表明,在高信噪比下,经过抑制突发错误的和积译码算法有更好的性能。

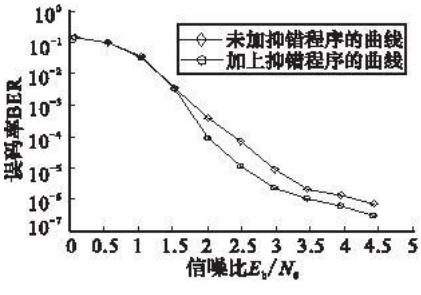


图 4 加抑错程序前后译码性能比较图

Fig. 4 Compare result for decoding performances

研究发现:对于似然比(LR)下的和积译码算法、对数似然比(LLR)下的和积译码算法,抑制突发错误的程序同样适用。

## 4　结束语

本文从软件方面出发,设计出了一种专门抑制突发错误的程序段,各类和积译码算法程序加上此段程序后,译码性能均得到了进一步提高,尤其在较低信噪比下,性能改善更加明显。这种抑制突发错误的软件设计思想,对于进一步优化 LDPC 码的译码性能具有一定的指导作用。对于更低信噪比下译码性能的优化,还需要进一步深入研究。

**参考文献:**


[1] MacKay D J C, Neal R M. Near Shannon Limit Performance of Low Density Parity Check Codes [J]. Electronics Letters, 1997, 33(6):457 – 458.

[2] Chung Sae – Young, Forney G D, Richardson T J, et al. On the Design of Low – density Parity – check Codes Within 0.004 5 dB of the Shannon Limit [J]. IEEE Commun Letters, 2001, 5(2):58 – 60.

[3] Kou Y, Lin S, Fossorier M P C. Low – Density Parity – check Codes Based on Finite Geometries: A Rediscovery and New Results [J]. IEEE Transactions on Information Theory, 2001, 47(7):2711 – 2736.

[4] MacKay David J C. Good Error – Correcting Codes Based on Very Sparse Matrices [J]. IEEE Transactions on Information Theory, 1999, 45(2):399 – 431.

[5] Lu J, Moura J M F. Structured LDPC Codes for High – Density Recording: Large Girth and Low Error Floor [J]. IEEE Transactions on Magnetics, 2006, 42(2):208 – 213.

[6] Tanner R M. A Recursive Approach to Low Complexity Codes [J]. IEEE Transactions on Information Theory, 1981, 27(5): 533 – 547.

[7] Moura Jose M F, Lu Jin, Zhang Haotian. Structured Low – Density Parity – Check Codes [J]. IEEE Signal Processing Magazine, 2004:42 – 55.

[8] 李水平,刘玉君,刑庆君. LDPC 码的环分析[J]. 信息工程大学学报,2003,4(4):82 – 84.
LI Shuiping, LIU Yujun, XING Qingjun. Analysis of LDPC's Loops [J]. Journal of Information Engineering University, 2003, 4(4):82 – 84. (in Chinese)

[9] Hu X, Fossorier M, Eleftheriou E. On the Computation of the Minimum Ddistance of Low – density Parity – check Codes [C]// Proc of Int Conf Commun. New Jersey: IEEE, 2004, 767 – 771.

[10] 马林华. 误码环境下的视频信源信道编码理论与技术研究[D]. 西安:西安电子科技大学,2006.
MA Linhua. Study on the Source and Channel Coding Theory and Techniques of Video Communication in Error – prone Environments [D]. Xi′an: Xidian University, 2006. (in Chinese)


(编辑:徐楠楠)

# A Software Method Restricting Incidental Error of LDPC Codes


YE Xiao – dong, MA Lin – hua, WANG Wei – min, YU Rui

(Engineering Institute, Air Force Engineering University, Xi′an 710038, China)



**Abstract:** This paper investigates the LDPC codes decoding algorithm over Additive White Gaussian Noise (AWGN) channels, analyzes the reason that incidental error appears in decoding process, and points out the reason why the incidental error appears is that there are loops in the parity – check matrix. Then a software method, which restricts incidental error, is proposed for decoding LDPC codes. In using this method the incidental error caused by loop's influence can be effectively restricted only by properly controlling the decoding program of LDPC. By this method, decoding performance of LDPC codes and decoding program stabilization are improved greatly, and the application of LDPC codes can enter more fields.



**Key words:** Low – Density Parity – Check Codes (LDPC); loops; incidental error