计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.200800134

LDPC 自适应最小和译码算法及其 FPGA 实现

王登天[1]　周　华[1,2,3]　钱荷玥[1]
1 南京信息工程大学　南京 210044
2 江苏省大气环境与装备技术协同创新中心　南京 210044
3 江苏省气象探测与信息处理重点实验室　南京 210044

摘　要　低密度奇偶校验码(Low-density Parity-check,LDPC)置信传播算法性能优异且被证明接近香农极限,但需要极其复杂的对数和三角函数运算,不利于现实使用。尽管最小和算法提高了计算的方便性和适用性,但也削弱了其译码能力。为了减少误码率(Bit Error Rate,BER)的损失,利用输入变量节点边信息绝对值的最小值、次小值和双曲正切函数的关系,引入自适应乘性因子提升算法性能。仿真结果显示,自适应最小和算法的性能比传统的对数似然比置信传播算法(Log-Likelihood Ratio Belief Propagation,LLR BP)提升了 0.2dB。此外,通过在 Xilinx 公司的 FPGA 平台上进行硬件测试,实现了 155 码长的 LDPC 码最小和算法仿真。

关键词:LDPC 码;自适应最小和算法;自适应乘性因子;算法性能;FPGA 实现
中图法分类号　TN919.3

LDPC Adaptive Minimum Sum Decoding Algorithm and Its FPGA Implementation

WANG Deng-tian[1],ZHOU Hua[1,2,3] and QIAN He-yue[1]
1 Nanjing University of Information Technology,Nanjing 210044,China
2 Jiangsu Collaborative Innovation Center for Atmospheric Environment and Equipment Technology,Nanjing 210044,China
3 Jiangsu Key Laboratory of Meteorological Observation and Information Processing,Nanjing 210044,China

Abstract　The belief propagation(BP) decoding algorithm for low-density parity-check (LDPC) codes has been shown to approach the Shannon limit,however it requires extremely complex logarithmic and trigonometric functions,which is not of practical interest. The minimum sum (MS) algorithm improves the convenience speed and simplifies the calculation at the expense of loss in decoding performance. In order to reduce the loss in bit error rate (BER),this paper introduce an adaptive multiplicative factor which considers the relationship between the absolute value of the input variable node side information,the second smallest value and the hyperbolic tangent function. As a result,the performance of the proposed adaptive MS algorithm is 0.2dB superior to the traditional LLR (Log-Likelihood Ratio) BP algorithm. Also,LDPC codes of 155 lengths are implemented based on the Xilinx FPGA platform.

Keywords　LDPC code,Adaptive minimum sum algorithm,Adaptive multiplicative factor,Algorithm performance,FPGA implementation

1　引言

1962 年,RoberGallager 率先提出了低密度奇偶校验码的概念,其译码性能趋近香农(Shannon)限[1]。LDPC 码凭借其可并行的译码结构、非常好的纠错性能和适中的译码难度,被国际移动通信标准化组织确定为 5G 数据通信编码方案[2]。当 LDPC 码的码长接近无限长,并且在最优设计情况下,其校验矩阵相匹配的 Tanner 图中不存在环路时,置信传播(Belief Propagation,BP)译码算法是首选的方案[3]。然而,对于有限长 LDPC 码环路不可避免,这时尽管 BP 算法有着优秀的译码能力,但其计算复杂度对硬件实现同样提出了较高的要求。针对这个问题,后人用加法运算代替原算法中的乘法运算,并添加双曲正切函数,这也就是对数域 BP 译码算法。虽该算法减少了计算量,但工程中难以实现的问题仍未完全解决。随后通过对数似然比置信传播算法(Log-Likelihood Ratio Belief Propagation,LLR BP)的校验节点更新运算进行近似处理,得到了最小和译码算法(Minimum Sum,MS),极大地降低了运算复杂度使其成为 LDPC 码在硬件上常规使用的译码方法[4]。

尽管 MS 算法拥有低复杂度,和便于在硬件上实施的优势,但削弱了其译码能力,也因此各种修正最小和算法被提出,其中补偿最小和算法(Offset Minimum Sum,Offset MS)和归一化最小和算法(Normalized Minimum Sum,Normalized MS)较为典型[5-6]。但其中的修正因子是通过仿真所确定的定值,不能在任意情况下有效地解决最小和算法中校验节点信息过高估计的问题,而且接收到的变量节点信息绝对值的最小值和次最小值对 BP 算法中校验节点信息值的大小起着决定因素[4]。基于此,本文针对归一化最小和算法提出了一种自适应最小和算法。基于码长为 155 的 LDPC 码仿真结果表明,其拥有更低的误码率,译码性能优于传统 MS 算法,当

基金项目:国家自然科学基金面上项目(61771248)
This work was supported by the General Project of the National Natural Science Foundation of China(61771248).
通信作者:王登天(1014765180@qq.com)

信噪比大于 3.5dB 时，译码性能优于 LLR BP 算法。

2 LDPC 码的基本原理

2.1 对数域 BP 译码算法

BP 算法的主要原理是信息顺着 Tanner 图的边持续发送和交换，校验节点信息和变量节点信息轮流更新，每次轮流更新表示一次迭代。迭代过程中，当更新校验节点信息时，与其连接的变量节点向其发送信息，信息被运算更新后发送回原来的变量节点[7]。类似地，更新变量节点信息时，执行同样的操作，并汇总收集的信息执行译码判决操作和输出译码结果。译码过程中校验节点更新和变量节点更新产生的信息称为外信息，而译码器开始工作时从信道接收到的初始信息被称为内信息。

BP 译码算法尽管在译码能力方面取得了很大的提升，但随之也带来了计算难度加大的问题，为优化此问题，对数似然比 LLR BP 算法被提出[8]。

对数域 BP 译码算法是将概率域 BP 译码算法传递的概率信息转换为对数似然比信息，从而由简单的加法运算取代原来复杂的乘法计算过程，在降低译码难度的同时译码时间也被大幅缩减，具体的 LLR BP 译码算法如下[4]。

1)初始化：当前迭代次数 $l=0$，最大迭代次数 $I_{\max}$，根据来自信道的接收信息 $y=(y_1,y_2,y_3,\cdots,y_n)$，初始化所有变量节点信息。

$$L^{(l)}(v_{ij})=L_{(y_i),i=1,2,\cdots,n} \tag{1}$$

2)校验节点更新：根据式(2)，对校验节点进行更新。

$$L^{(l)}(c_{ji})=2\tanh^{-1}\Big(\prod_{i'\in N(j)/i}\tanh\Big(\frac{1}{2}L^{(l)}(v_{ji'})\Big)\Big) \tag{2}$$

3)变量节点更新：根据式(3)对变量节点进行更新。

$$L^{(l)}(v_{ij})=L(y_i)+\sum_{j'\in N(i)/j}L^{(l)}(c_{j'i}) \tag{3}$$

4) 计算变量节点后验概率：

$$L^{(l)}(q_i)=L(y_i)+\sum_{j\in N(i)}L^{(l)}(c_{ji}) \tag{4}$$

5)译码判决

若 $L^{(l)}(q_i)>0$，则 $\hat{x}_i=0$；否则 $\hat{x}_i=1$。结束译码的标志为达到最大迭代次数 $I_{\max}$ 或 $[\hat{x}_1,\hat{x}_2,\hat{x}_3,\cdots,\hat{x}_n]*\boldsymbol{H}^{\mathrm{T}}=0$，否则 $l=l+1$ 并返回步骤 1)。公式各符号的含义如表 1 所列。

表 1 上述公式各符号的表示含义

Table 1 Meaning of each symbol in the above formula

符号	含义
c_j	校验节点
v_i	变量节点
$N(i)$	所有与 v_i 相邻的校验节点集合
$N(j)$	所有与 c_j 相邻的变量节点集合
$L(v_{ij})$	变量节点 v_i 传递给校验节点 c_j 的边信息
$L(c_{ji})$	校验节点 c_j 传递给变量节点 v_i 的边信息
$N(i)/j$	除了校验节点 c_j 外，所有与变量节点 v_i 相邻的校验节点集合
$N(j)/i$	除了变量节点 v_i 外，所有与校验节点 c_j 相邻的变量节点集合

2.2 最小和译码

根据上文对 BP 算法的分析可知，LLR BP 算法具有低复杂度、高译码效率的优点，但在计算校验节点信息过程中包含 tanh 函数，根据双曲正切函数的性质，可通过查找表来实现，在获得相对高的译码准确度的同时也带来了占用大量逻辑资源的问题。为了解决此问题，简化 BP 算法应运而生，被称为最小和译码算法[9]。

最小和译码算法是 LLR BP 的近似算法，利用辅助函数进一步优化 LLR BP 算法中的校验节点更新公式，得到新的校验节点更新公式[10]，即式(2)可演变为式(6)。

$$f(x)=-\ln(\tanh(\frac{1}{2}x))=\ln\frac{e^x+1}{e^x-1},x\geqslant 1 \tag{5}$$

$$L^{(l)}(c_{ji})=\prod_{i'\in N(j)/i}sign(L^{(l)}(v_{i'j}))*\min_{i'\in N(j)/i}|L^{(l)}(v_{ji'})| \tag{6}$$

MS 算法虽然降低了校验节点公式的运算复杂度，但不可避免地出现了译码性能下降的问题。由于 tanh 函数范围为(−1,1)，arctanh 函数范围为(−∞,∞)，在 LLR BP 算法中，变量节点信息首先经过 tanh 函数运算，其结果范围为(−1,1)，在经过多次乘法运算，结果的范围变为(0,1)，最后由 arctanh 函数将其恢复到(−∞,∞)范围。经过此过程后，其结果小于 $\min_{i'\in N(j)/i}(\tanh(\frac{1}{2}|L(v_{ji'})|))$，最后得到的校验节点信息绝对值小于 $\min_{i'\in N(j)/i}|L(v_{ji'})|$，这就是导致最小和算法中校验节点信息值偏高的原因[1]。

为了解决过高估计的问题，提升 MS 算法的性能，在式(6)中引入乘性因子 $\alpha(0<\alpha<1)$ 和修正因子 $\beta(\beta>0)$，即可得到改进后的校验节点更新公式，如式(7)所示[11]：

$$L^{(l)}(c_{ji})=\alpha*\prod_{i'\in N(j)/i}sign(L^{(l)}(v_{ji'}))*\max(\min_{i'\in N(j)/i}|L^{(l)}(v_{ji'})|-\beta,0) \tag{7}$$

2.3 自适应补偿最小和算法

文献[1]提出了一种自适应补偿最小和算法[1]。分析最小和算法存在过高估计的问题[12]，首先提出修正因子 β 的自适应修正因子 $\hat{\beta}$：

$$\hat{\beta}=1-\tanh(x_2-x_1) \tag{8}$$

但通过仿真分析可知，在处于较低码率的情况下，修正因子为自适应的算法译码性能并没有优于修正因子为固定值的算法，BP 算法与此算法的校验节点信息值的差值并没有得到有校的减少，即最小和算法中引入此自适应因子 $\hat{\beta}$ 相比采用固定的修正因子，其校验节点信息值偏高的问题并没有得到有效的解决[1]。

文献[1]进一步对比了不同码率、不同信噪比条件下，自适应修正因子和固定修正因子的关系，最后提出新的自适应修正因子 $\tilde{\beta}$：

$$\tilde{\beta}=\begin{cases}1-\tanh(diff), & diff\geqslant\operatorname{arctanh}(1-\beta)\\ \beta, & \text{other}\end{cases} \tag{9}$$

其中，β 为固定修正因子；$diff=x_2-x_1$，x_1，x_2 分别为输入变量节点边信息绝对值的最小值和次小值。

即文献[1]中更新校验节点公式为：

$$L^{(l)}(c_{ji})=\prod_{i'\in N(j)/i}sign(L^{(l)}(v_{ji'}))*\max(\min_{i'\in N(j)/i}|L^{(l)}(v_{ji'})|-\tilde{\beta},0) \tag{10}$$

为了分析和对比文献[1]提出的改进算法，现分别选用码长为 155、码率为 0.4 的 LDPC 码，在信噪比分别为 1 dB 和 2 dB，码长为 576 和 1 152、码率为 0.5 的 LDPC 码时，在信噪比分别为 0.5 dB 和 1 dB 下进行仿真，得到不同固定修正因子 β。当选用码长为 155 时，固定修正因子 β 选取 0.3。当选用码长为 576 和 1 152 时，固定修正因子 β 为 0.4。

3 自适应最小和算法

由第 2 节的分析可知，为解决最小和算法中校验节点信息值偏高的问题，可以通过引入乘性因子 α 和修正因子 β 来

减小 $\min_{i'\in N(j)/i}|L(v_{i'j})|$ 的值，补偿性能损失。但对于修正最小和算法，如果选择错误的乘性因子 α 或者修正因子 β，对解决信息值偏高问题无疑是雪上加霜。

通过第 2 节可知导致最小和算法过高估计的原因，文献[1]针对 2.2 节改进最小和算法中的固定修正因子 β 提出改进方法，引入自适应修正因子 $\tilde{\beta}$ 替代固定修正因子 β。本文根据文献[1]中提出的方法，针对 2.2 节改进最小和算法中的固定乘法性因子 α，引入自适应乘性因子 $\tilde{\alpha}$ 代替固定乘法性因子 α。

分析 BP 算法中校验节点信息更新公式可知，其是多个(0,1)的值进行乘积运算，累积结果主要由边信息绝对值的最小值 x_1 和次小值 x_2 决定，当两者之间的差值 $diff=x_2-x_1$ 越大，乘积的结果就越趋近于最小值 x_1，则相应的乘性因子 α 越接近 1 和固定修正因子 β 越趋近于 0[1]。此外，MS 算法是在双曲正切函数的基础上将式(2)变为式(6)，因此 α 值的选取应从双曲正切函数的关系上考虑。基于此，在适当增加 MS 译码算法复杂度的前提下，提出乘性因子 α 的自适应公式：

$$\tilde{\alpha}=\tanh(x_2-x_1) \tag{11}$$

其中，x_1 和 x_2 分别为输入变量节点边信息绝对值的最小值和次小值。

即最后的自适应最小和译码算法的流程如下。

1)初始化：当前迭代次数 $l=0$，最大迭代次数 $I_{\max}$，根据来自信道的接收信息 $y=(y_1,y_2,y_3,\cdots,y_n)$，初始化所有变量节点信息。

$$L^{(l)}(v_{ij})=L_{(y_i),i=1,2,\cdots,n} \tag{12}$$

2)校验节点更新：

$$L^{(l)}(c_{ji})=\tilde{\alpha}\prod_{i'\in N(j)/i}\operatorname{sign}(L^{(l)}(v_{ji'}))*\min_{i'\in N(j)/i}|L^{(l)}(v_{ji'})| \tag{13}$$

其中，$\tilde{\alpha}$ 为自适应乘性因子。

3)变量节点更新：

$$L^{(l)}(v_{ij})=L(y_i)+\sum_{j'\in N(i)/j}L^{(l)}(c_{j'i}) \tag{14}$$

4) 计算变量节点的后验概率：

$$L^{(l)}(q_i)=L(y_i)+\sum_{j\in N(i)}L^{(l)}(c_{ji}) \tag{15}$$

5)译码判决：若 $L^{(l)}(q_i)>0$，则 $\hat{x}_i=0$；否则 $\hat{x}_i=1$。结束译码的标志为达到最大迭代次数 $I_{\max}$ 或 $[\hat{x}_1,\hat{x}_2,\hat{x}_3,\cdots,\hat{x}_n]*\boldsymbol{H}^{\mathrm{T}}=\mathbf{0}$，否则 $l=l+1$ 并返回步骤 2)。

4 性能仿真与分析

本文采用码长为 155 和码率为 0.4、码长为 576 和码率为 0.5 以及码长为 1152 和码率为 0.5 的 QC-LDPC 码进行仿真，以验证自适应最小和算法(Adaptive Minimal Sum Algorithm，AMS)。采用 BPSK 调制和高斯信道传输，设置最大迭代次数，并统计误帧数，当误帧数达到设置数值时跳出迭代循环并统计误码率和平均迭代次数。其中，码长为 155 和码长为 576 的 LDPC 码的最大迭代次数为 50，误帧数为 10000 次，码长为 1152 的 LPDC 码设置迭代次数为 30，误帧数为 5000。

如图 1 所示，最小和算法和 LLR BP 算法相差 0.3 dB 左右，本文提出的自适应最小和算法从信噪比 2 dB 到 4 dB 一直优于原最小和算法，当信噪比处于 3.5 dB 时，本文提出的自适应最小和算法开始达到 LLR BP 算法的性能，并从 3.5 dB 之后，开始优越于 LLR BP 算法的译码效果。虽然增加了较小的算法复杂度，但让算法性能得到很大的提高，得到了更突出的译码效果。

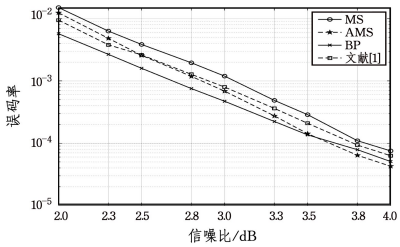

图 1 当码长为 155 时不同算法的误码率

Fig. 1 Bit error rate of different algorithms with code length 155

虽然文献[1]提出的自适应补偿最小和算法，优于原最小和算法，其平均迭代次数比 MS 算法减少了 1 到 2 次，但其译码性能一直无法优于 LLR BP 算法。本文提出的自适应改进方法，虽然增加了一次 tanh 函数的运算，平均迭代次数相比 MS 算法增加了 1 到 2 次，但从信噪比 2 dB 开始，本文提出的自适应译码算法的译码性能不断逼近文献[1]提出的算法的译码性能，并当信噪比为 2.8 dB 开始，本文算法优于文献[1]提出的译码算法。

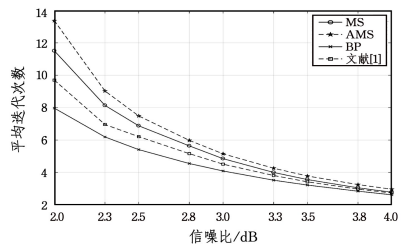

图 2 当码长为 155 时不同算法的平均迭代次数

Fig. 2 Average number of iterations of different algorithms with code length 155

由图 3 可知，码率为 0.5、码长为 576 时，本文提出的译码算法的误码率在 1.5 dB 到 2 dB 期间不断逼近文献[1]中提出的算法，当进入信噪比 2.3 dB 后，本文提出的译码算法是 4 种译码算法中性能最好的一种，相比 LLR BP 算法性能提升了 0.2 dB。通过引入自适应乘性因子，本文提出的译码算法平均迭代次数相比文献[1]中的算法增加了 4 次，相比 LLR BP 算法增加了 1 次，增加了译码算法的复杂度。

通过对码长为 1152 的 LDPC 码进行仿真可以发现，引入自适应乘性因子，增加了译码算法的复杂度，相比文献[1]中的算法，增加了 5 次左右的平均迭代次数，但当信噪比进入 2 dB 时，本文提出的译码算法相比文献[1]中的算法获得了 0.1 dB 的性能增益，相比 LLR BP 算法获得了 0.2 dB 的性能增益。

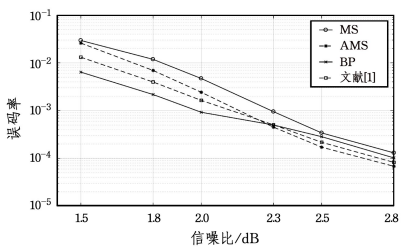

图 3 当码长为 576 时不同算法的误码率

Fig. 3 Bit error rate of different algorithms with code length 576

综上所述，文献[1]中的方法虽然优化了算法复杂度和译码性能，但本文通过增加 4 次左右的平均迭代次数，获得了更好的译码性能，弥补了文献[1]中的算法在提升译码性能方面的不足，并且算法增加的复杂度在可接受范围内。

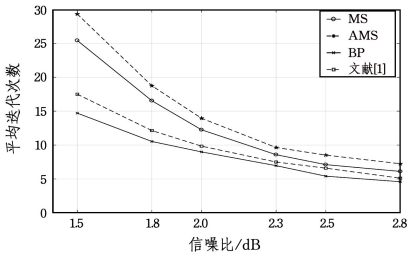

图 4　当码长为 576 时不同算法的平均迭代次数

Fig. 4　Average number of iterations of different algorithms with code length 576

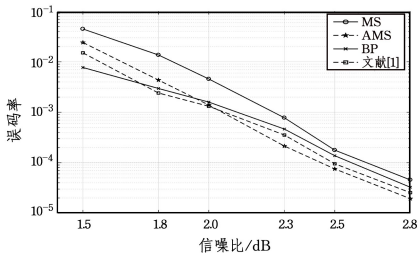

图 5　当码长为 1 152 时不同算法的误码率

Fig. 5　Bit error rate of different algorithms with code length 1 152

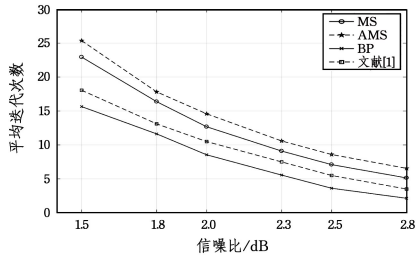

图 6　当码长为 1 152 时不同算法的平均迭代次数

Fig. 6　Average number of iterations of different algorithms with code length 1 152

5　LDPC 译码器设计

在设计 LDPC 译码器的过程中，须兼顾译码速率和硬件资源消耗这两个需求，但这两个要求是相互矛盾的，追求一方性能的同时，必然会导致另一方情况的恶化。随着使用场景的变化，对这两个要求的标准也不一样，为了能更好地应对不同场景下的使用需求，一共出现了 3 种硬件结构：全并行结构、串行结构和部分并行结构[13]。

为了验证 LDPC 自适应最小和译码算法的硬件可实行性，选用 155 码长的 QC_LDPC。根据码长特点，在硬件结构上选用全并行结构，可以达到很高的译码速率，且不需要过多的控制逻辑，简化译码器的设计过程，只需对两个节点的交替工作进行控制，但也有不可忽视的缺点，即消耗大量的硬件资源、缺乏灵活性、一种译码器只能针对一种码型的 LDPC 码[14]。

根据自适应最小和译码算法的译码流程，模块化的设计思路贯穿了整个译码器设计的全过程，整体结构设计如图 7 所示。

整个译码流程类似于流水线处理，当信息接收模块接收到初始信息后，即开始译码。根据对自适应最小和译码算法中数据大小的分析和对 Xilinx CORDIC IP 核数据位宽的设定，对译码器进行浮点数的定点化，设置整数 10 位、小数 5 位、符号位 1 位。控制模块监视和控制整个译码模块的运作，判断译码过程中是否达到最大迭代次数以及控制每个模块的启动与停止[15]。

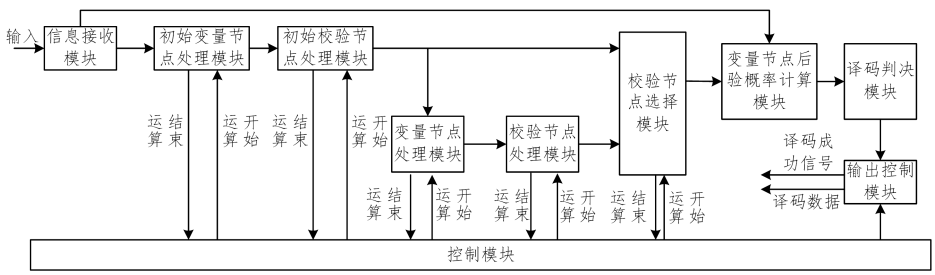

图 7　译码器整体结构设计

Fig. 7　Decoder overall structure design

将计算变量节点和校验节点分为 4 个模块，即初始变量节点处理模块、初始校验节点处理模块、变量节点处理模块和校验节点处理模块。此设计的目的是为了将更新两节点信息值的过程变为两个整体，即第一次迭代过程中与非第一次迭代过程中。此设计有助于简化第一次迭代过程中初始似然值生成变量节点的过程。因此，需要引入校验节点选择模块，在译码计算过程中，从初始校验节点处理模块和校验节点处理模块的输出值中选择出合适的值进行迭代运算。输出控制模块主要控制译码结果和译码完成信号，当译码成功时，译码成功信号 frame_error 输出低电平信号，如果译码失败，则输出高电平信号。LDPC 译码器的总体时序仿真图如图 8 所示。

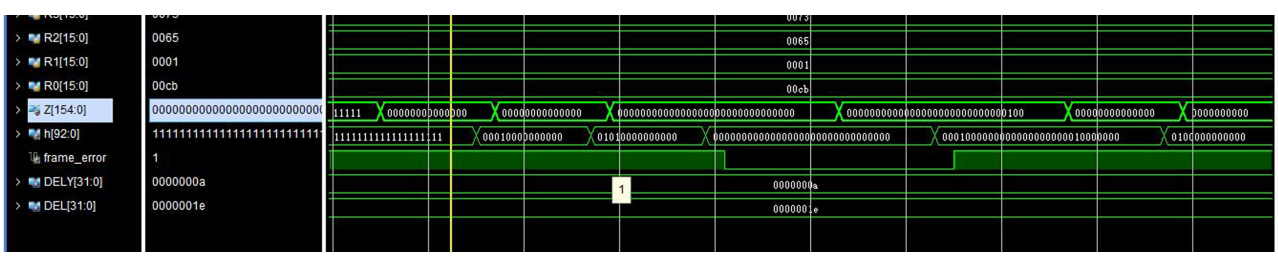

图 8　译码器的总体时序仿真图

Fig. 8　Decoder overall timing simulation diagram

结束语 本文分析了导致最小和算法存在过高估计问题的原因，并提出了 LDPC 码的自适应最小和算法。在本文提出的算法中，利用输入变量节点边信息绝对值的最小值和次小值差值的 tanh 函数的值来代替归一化最小和算法中固定的乘性因子 α，从而得到自适应最小和算法。改进后的译码算法，虽然增加了一次 tanh 函数的运算，使自适应最小和译码算法平均迭代次数相比文献[1]中的算法增加了 4 次左右，但获得了更加优异的译码性能。在高信噪比区域达到并超过了 LLR BP 译码算法的译码性能，并优于文献[1]提出的译码算法。最后在 FPGA 平台完成了 LDPC 译码器的设计，验证了算法的可行性，但硬件实现消耗了较大的计算资源，需进一步优化硬件实现方式。

参 考 文 献

[1] 王琼，李思舫，罗亚洁. LDPC 码的自适应补偿最小和译码算法[J]. 电讯术，2019，59(6)：635-640.

[2] 杨卫国. 基于减少过估计的改进 LDPC 码最小和译码算法[J]. 指挥控制与仿真，2017，39(6)：53-57.

[3] GALLAGER R. Low-density parity-check codes[J]. journal of circuits & systems，2008，8(1)：3-26.

[4] 张玉凯. 准循环 LDPC 码的编译码器设计及 FPGA 实现[D]. 西安：西安电子科技大学，2009.

[5] CHEN J H，DHOLAKIA A，ELEFTHERIOU E. Reduced-Complexity Decoding of LDPC Codes[J]. Communications IEEE Transactions on，2005，53(8)：1288-1299.

[6] ANGARITA F，VALLS J，ALMENAR V，et al. Reduced-Complexity Min-Sum Algorithm for Decoding LDPC Codes With Low Error-Floor[J]. Circuits and Systems I：Regular Papers，IEEE Transactions on，2014，61(7)：2150-2158.

[7] 姜莹. 基于 FPGA 的高效 LDPC 译码器的研究[D]. 北京：北京交通大学，2014.

[8] 韩家宇. 基于 FPGA 的 LDPC 码编译码研究及硬件实现[D]. 哈尔滨：哈尔滨工程大学，2017.

[9] VITYAZEVV V，LIKHOBABIN E A，USTINOVA E A. Min-sum algorithm-structure based decoding algorithms for LDPC codes[C]// 2014 3rd Mediterranean Conference on Embedded Computing(EMCO). 2014：256-259.

[10] 马志刚，郑鹏宇，王亚军. QC-LDPC 译码器的 FPGA 设计实现与分析[J]. 现代导航，2017，8(3)：204-209.

[11] 袁瑞佳. LDPC 码的高效编译码实现技术研究[D]. 西安：西安电子科技大学，2012.

[12] LI H，DING H，ZHENG L，et al. An efficient scheduling scheme for layered belief propagation decoding of regular LDPC codes[C]// International Conference on Ultra Modern Telecommunications. 2016：397-400.

[13] 黄福威. 5G-LDPC 码编译码器设计与 FPGA 实现技术研究[D]. 西安：西安电子科技大学，2019.

[14] 张辉. 基于 FPGA 的 LDPC 译码器硬件实现方法研究[D]. 哈尔滨：哈尔滨工业大学，2019.

[15] 李硕. 基于 FPGA 的 QC-LDPC 编译码器研究[D]. 哈尔滨：哈尔滨理工大学，2015.

WANG Deng-tian, born in 1995, master. His main research interests include information theory and coding.

(上接第 607 页)

[17] KUMAR N，ZEADALLY S，RODRIGUES J J P C. Vehicular delay-tolerant networks for smart grid data management using mobile edge computing[J]. IEEE Communications Magazine，2016，54(10)：60-66.

[18] NASTIC S，RAUSCH T，SCEKIC O，et al. A serverless real-time data analytics platform for edge computing[J]. IEEE Internet Computing，2017，21(4)：64-71.

[19] POISSANT A，CASTANO L，XU H. Mitigation of Ground Impact Hazard for Safe Unmanned Aerial Vehicle Operations[J]. Journal of Aerospace Information Systems，2020，17(12)：647-658.

[20] LI X，WAN J，DAI H N，et al. A hybrid computing solution and resource scheduling strategy for edge computing in smart manufacturing[J]. IEEE Transactions on Industrial Informatics，2019，15(7)：4225-4234.

[21] ZOU Y H，GUAN P J，YANG B，et al. Rapid Eye Movement Tracking System Based on FPGA and Projection Algorithm[J]. Advanced Engineering Sciences，2016，48(3)：100-106.

QIAN Ji-de, born in 1988, Ph. D, research assistant professor, is a member of China Computer Federation. His main research interests include flight operation safety, computer vision, deep learning and edge computing.

WANG Qian-lei, born in 1996, postgraduate. His main research interests include deep learning and computer vision.