

# 一种LOG-MAP算法的改进迭代实现及其结构

周 亮\*

(电子科技大学通信抗干扰技术国家级重点实验室 成都 610054)

**【摘要】**对LOG-MAP算法进行了可并发性分析,利用可以完整接收一个 $N$ 长符号传输帧的条件以及正向迭代和反向迭代中的固有对称性质,提出了LOG-MAP算法的一种修正迭代实现算法,其正向和反向迭代计算次数只需码长 $N$ 的一半,故比标准LOG-MAP算法提高了一倍的处理速度,且没有空间开销的增加。同时,根据修正的迭代实现算法给出了相应的适于FPGA实现的双总线硬件结构的实现方案。

**关键词** 最大后验概率; 对数最大后验概率; 迭代算法; 双总线结构

中图分类号 TN91 文献标识码 A

## A New Iterative Implementation and Architecture for LOG-MAP Algorithm

Zhou Liang

(National Key Laboratory of Communication, UEST of China Chengdu 610054)

**Abstract** This paper analyzes the parallel mechanism of logarithmic maximum a posteriori algorithm and presents a modified iterative procedure by the use of the possibility of receiving the full  $N$ -symbols frame and the symmetry being inherent in the forward-backward iteration. The procedure achieves a double decoding speed faster than that of the conventional one without the increase of RAM cost because the number of iterations is limited to a half of the code-length  $N$ . Also according to the modified algorithm this paper proposes a hardware scheme characterized by a two-busses architecture suitable for implementing with FPGA.

**Key words** maximum a posteriori; logarithmic maximum a posteriori; iterative algorithm; two-busses architecture

最大后验概率(Maximum a Posteriori, MAP)算法或对数最大后验概率(Logarithmic MAP, LOG-MAP)算法是Turbo码译码的核心算法<sup>[1]</sup>,但迄今为止,在MAP和LOG-MAP算法的实现中仍然沿用标准的实现方案或其非本质性的修正<sup>[2,3]</sup>,其正向和反向迭代计算次数分别等于码长 $N$ ,因而具有 $2N$ 长的时间开销。尽管开窗算法可以减小一定的计算量<sup>[4]</sup>,但并没有获得足够满意的性能改善。本文利用可以完整接收一个 $N$ 长传输帧的条件以及正向迭代和反向迭代中的固有对称性质,提出一种改进的迭代算法。

### 1 LOG-MAP算法

设 $1/2$ 码率递归系统卷积码的状态数为 $M$ ,接收实数符号序列为 $R(k) = (x_k, y_k)$ ,  $k = 1, 2, \dots, N$ ,码的前向和反向递归向量分别为 $A(k) = (A_k^0, A_k^1, \dots, A_k^m, \dots, A_k^{M-1})$ 和 $B(k) = (B_k^0, B_k^1, \dots, B_k^m, \dots, B_k^{M-1})$ ,数据 $i$ 的分支度量量为 $D^{(i)}(k) = (D_k^{(i,0)}, D_k^{(i,1)}, \dots, D_k^{(i,m)}, \dots, D_k^{(i,M-1)})$ ,  $i=0,1$ 。又记多元 $E$ 操作为

$$E_{i=1}^n a_i = -\log_b(b^{-a_1} + b^{-a_2} + \dots + b^{-a_n}) \quad (1)$$

2003年9月1日收稿

\*男 42岁 硕士 副教授 主要从事信道编码及其应用方面的研究

则对有零状态结尾码LOG-MAP算法是计算第 $k$ 时刻数据的似然值 $L(k)$ <sup>[2,3,5]</sup>,并按其符号作出0,1消息判决 $\hat{d}_k$ ,其迭代算式为

$$A_k^m = E_{j=0,1}(A_{k-1}^{b(j,m)} + D_k^{(j,b(j,m))}) \quad m=0,1,\dots,M-1, \quad k=1,2,\dots,N \quad (2)$$

$$B_k^m = E_{j=0,1}(B_{k+1}^{f(j,m)} + D_k^{(j,m)}) \quad m=0,1,\dots,M-1, \quad k=N,\dots,2,1 \quad (3)$$

$$D_k^{(i,m)} = -W_k - (Z_k + Ax_k)i - Ay_k c^{i,m} \quad k=1,2,\dots,N \quad (4)$$

$$A_0^m = B_{N+1}^m = \begin{cases} 0, m=0 \\ -\infty, m \neq 0 \end{cases} \quad (5)$$

$$L(k) = \prod_{m=0}^{M-1} [A_k^m + D_k^{(0,m)} + B_{k+1}^{f(0,m)}] - \prod_{m=0}^{M-1} [A_k^m + D_k^{(1,m)} + B_{k+1}^{f(1,m)}] \quad k=1,2,\dots,N \quad (6)$$

$$\hat{d}_k = 1 - \text{sgn}(L(k)) \quad k=1,2,\dots,N \quad (7)$$

式中  $W_k = \log_b T_k + \log_b p_k^0$ ,  $T_k = \frac{1}{M} \frac{1}{2\pi\sigma^2} \exp\left[\frac{1}{2\sigma^2}((x_k+1)^2 + (y_k+1)^2)\right]$ ,  $Z_k = -\log p_k^0 / p_k^1$ ,  $A = \frac{2}{\sigma^2} \log_b e$ ,  $p_k^i$  为第 $k$ 时刻数据为 $i$ 的先验概率,  $\sigma^2$  为AWGN信道的方差,  $c^{i,m}$  为信息 $i$ 、状态 $m$ 时的编码数据。 $b(j,m)$  是当前数据为 $j$ 且状态为 $m$ 时编码器的前一状态,  $f(j,m)$  是当前数据为 $j$ 且状态为 $m$ 时编码器的下一状态。

## 2 LOG-MAP的改进迭代算法

在标准LOG-MAP的基本迭代过程中因正向迭代与反向迭代分别进行,共有 $2N$ 次迭代(延时),而在几乎所有Turbo码的应用场合<sup>[6,7]</sup>,都可在译码前完整接收一个 $N$ 长的传输帧( $R(k)$ )或码字,因而注意到迭代过程的对称性,可以并行的成对计算 $D$ 参数,理论上至少减少一半的迭代次数,从而可以大大减小迭代延时。

对 $k=1,2,\dots,N$ ,仍然记 $D,A,B,L$ 诸参数的迭代计算为 $D^i(k) = D^i(R(k))$ ,  $A(k)=A(A(k-1), D^i(k))$ ,  $B(k)=B(B(k+1), D^i(k))$ ,  $L(k)=L(A(k), B(k+1), D^0(k), D^1(k))$ ,则新的LOG-MAP迭代过程描述为“修正LOG-MAP迭代算法”如下:

1) 对 $k=1,2,\dots,N/2$

(1.1) 读取 $R(k)$ , 读取 $A(k-1)$ , 计算 $D^i(k)$ 并存储, 计算 $A(k)$ 并存储;

(1.2) 读取 $R(N-k+1)$ , 读取 $B(N-k+2)$ , 计算 $D^i(N-k+1)$ 并存储, 计算 $B(N-k+1)$ 并存储。

2) 对 $k=N/2, (N/2)-1, \dots, 1$

(2.1) 读取 $D^i(N-k+1)$ , 读取 $B(N-k+2)$ , 读取 $A(N-k)$ , 计算 $A(N-k+1)$ 并存储, 计算 $L(N-k+1)$ 并判决输出 $\hat{d}(N-k+1)$ ;

(2.2) 读取 $D^i(k)$ 、 $A(k)$ 和 $B(k+1)$ , 计算 $B(k)$ 并存储, 计算 $L(k)$ 并判决输出 $\hat{d}(k)$ 。

迭代前半程 ( $k=1,2,\dots,N/2$ )	迭代后半程 ( $k=N/2,(N/2)-1, \dots,1$ )
$R(1) \dots R(k) \dots R(N/2)$	
$D^i(1) \dots D^i(k) \dots D^i(N/2)$	$D^i((N/2)+1) \dots D^i(N-k+1) \dots D^i(N)$
$A(0) A(1) \dots A(k) \dots A(N/2)$	$A((N/2)+1) \dots A(N-k+1) \dots A(N)$
	$B((N/2)+2) \dots B(N-k+2) \dots B(N+1)$
	$L((N/2)+1) \dots L(N-k+1) \dots L(N)$
$R(N) \dots R(N-k+1) \dots R((N/2)+1)$	
$D^i(N) \dots D^i(N-k+1) \dots D^i((N/2)+1)$	$D^i(N/2) \dots D^i(k) \dots D^i(1)$
$B(N+1) B(N) \dots B(N-k+1) \dots B((N/2)+1)$	$B(N/2) \dots B(k) \dots B(1)$
	$A(N/2) \dots A(k) \dots A(1)$
	$B((N/2)+1) \dots B(k+1) \dots B(2)$
	$L(N/2) \dots L(k) \dots L(1)$

图1 LOG-MAP算法的改进迭代时序

修正LOG-MAP迭代过程时序如图1所示,由此可见其迭代过程共只有 $N$ 次迭代,比基本LOG-MAP迭代次数减小一半,因而提高约一倍的译码速度。此外,在新迭代过程的前半程( $k=1,2,\dots,N/2$ )和后半程( $k=N/2,(N/2)-1,2,\dots,1$ )中,步骤1)的存取地址不发生冲突,故可并行处理实现,同理步骤2)也可并行处理实现。由于前半过程完成了全部( $k=1,\dots,N$ )分支度量 $D^i(k)$ 的计算,可在后半过程处理时处理下一传输帧,从而实现比基本LOG-MAP迭代小一半窗口宽度的流水线方式处理,因此还可以提高一倍的译码处理吞吐量。

### 3 LOG-MAP改进迭代算法的实现结构

记新的LOG-MAP迭代算法中步骤(1.1)和(1.2)、(2.1)和(2.2)的实现模块分别为M(1.1)、M(1.2)、M(2.1)和M(2.2)。

由迭代算法的时序知, M(1.1)和M(1.2)可以并行操作, M(2.1)和M(2.2)也可以并行操作。在M(1.1)中 $A(0)$ 可以由时序置位控制实现,同样M(1.2)中 $B(N+1)$ 也可以由时序置位控制实现,其后的 $A(k-1)$ 和 $B(N-k+2)$ 则由时序移位锁存控制实现。类似地, M(2.1)中的 $A(N/2)$ , M(2.2)中的 $B((N/2)+1)$ 可以由时序写入锁存控制实现,其后的 $A(N-k)$ 和 $B(k+1)$ 由时序移位锁存实现。

如果在M(2.1)中设置反馈缓冲机制,则前向度量 $A((N/2)+1), A(N/2)+2, \dots, A(N)$ 无需存储,同样M(2.2)的后向度量 $B(N/2), B(N/2)-1, \dots, B(1)$ 也无需存储,因此只有M(1.1)和M(1.2)的输出需要缓存,对RAM的需求为

$$2NMQ(D\text{存储}) + (N/2)MQ(A\text{存储}) + (1+(N/2))MQ(B\text{存储}) = (3N+1)MQ \quad (8)$$

式中  $Q$ 为每个度量分量的量化比特数。

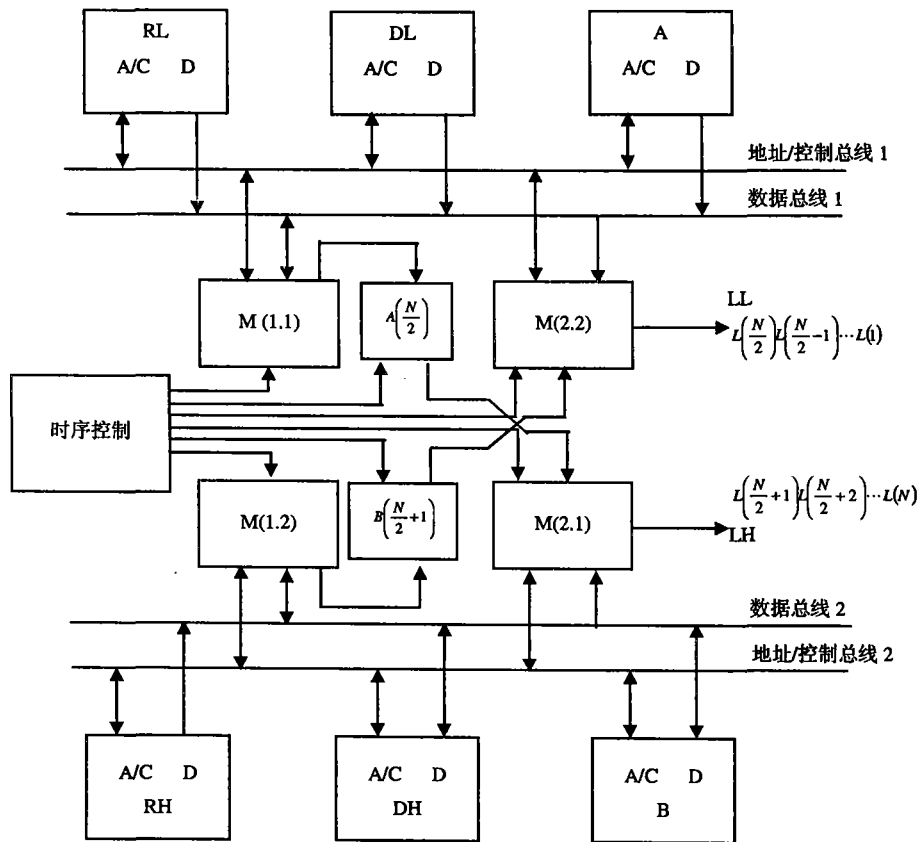


图2 修正LOG-MAP迭代算法的双总线实现结构

由于并行操作的需要, A、B两参数存储需设置为独立的两存储器, D参数分为DL和DH两个独立的前两部分数据存储器,分别存放M(1.1)和M(1.2)的输出,另外考虑存储器的字节结构,所以DL、DH同为 $NM \lceil Q/8 \rceil$ 字节, A存储器为 $(N/2)M \lceil Q/8 \rceil$ 字节, B存储器为 $(1+N/2)M \lceil Q/8 \rceil$ 字节。DL、DH、A、B数

据的存储结构由算法流程限定为堆栈结构。

由于RL、DL、A与RH、DH、B之间除了 $A(N/2)$ 和 $B((N/2)+1)$ 两特殊数据外,两组RAM之间没有直接的数据交换,因此为提高算法实现速度,可采用双总线结构来最大速率的实现新的LOG-MAP算法,其双总线实现结构如图2所示。图中,A/C表示地址/控制端口,D表示数据端口。 $M(1.1)$ 模块为加入时序控制,包括地址和读写控制后的修改 $M(1.1)$ 模块, $M(1.2)$ 、 $M(2.1)$ 和 $M(2.2)$ 类似。

## 4 结束语

在Turbo码的应用场合,可以在译码前完整接收一个 $N$ 长符号的传输帧( $R(k)$ )或码字,因图1所示的LOG-MAP算法本身迭代过程的对称性,其迭代过程可以分解为 $k=1, \dots, N/2$ 的前半正向迭代过程和 $k= N/2, (N/2)-1, \dots, 1$ 的后半反向迭代过程两部分,并且可以并行的成对计算 $D$ 参数,在理论上减少了一半的迭代次数,从而可以大大减小迭代延时,即LOG-MAP算法本身具有可并发操作的特性。此外,由于前半迭代过程中完成了全部( $k=1, \dots, N$ )分支度量 $D'(k)$ 的计算,可以在后半过程处理时处理下一传输帧,从而实现比基本LOG-MAP迭代小一半窗口宽度的流水线方式处理,因此提高了一倍的译码处理吞吐量。

本文通过对LOG-MAP算法本身可并发操作的特性分析,同时注意到前向度量 $A((N/2)+1)$ ,  $A((N/2)+2)$ ,  $\dots$ ,  $A(N)$ 和后向度量 $B(N/2)$ ,  $B((N/2)-1)$ ,  $\dots$ ,  $B(1)$ 之间没有数据交换且均无需存储,只有 $A(N/2)$ 和 $B((N/2)+1)$ 两个特殊数据需要进行数据的交换,因此这种数据结构特性提供了LOG-MAP算法新的并行迭代实现方式的可能,并且不增加数据存贮量,由此获得了采用双总线方式的LOG-MAP算法新的硬件实现结构,其实现结构特别适合于FPGA实现。

## 参 考 文 献

- [1] Berrou C, Glavieux A, Thitimajshima P. Near Shannon limit error-correcting coding and decoding[C]. ICC'93, 1993. 1064-1070
- [2] Pietrobon S S, Gross W J, Gulak P G. Implementation and performance of a Turbo/MAP decoder[J]. Int. J. Satell. Commun., 1998, 16: 23-46
- [3] Boutillon E, Gross W J, Gulak P G. VLSI architectures for the MAP algorithm[J]. IEEE Trans-COM, 2003, 51: 175-185
- [4] Yeo E, Nikolic B, Anantharam V. Iterative decoder architectures[EB/OL]. URL <http://www.eecs.berkeley.edu/~ananth/2002+/Engling/ieeecommm2003.pdf> 2002-12-17/2003-08-02
- [5] Zhang Zhongpei, Zhou Liang. A low complex parallel decoding structure for Turbo-codes[EB/OL]. ICCT 2000, <http://www.ifip.org.at/2001-12-14/2003-08-04>
- [6] Divsalar D, Pollara F. On the design of Turbo codes[EB/OL]. TDA Progress Report 42-123, 1995 URL [http://tmo.jpl.nasa.gov/tmo/progress\\_report/42-123/123D.pdf](http://tmo.jpl.nasa.gov/tmo/progress_report/42-123/123D.pdf) 2001-12-05/2003-08-02
- [7] VOCAL Technologies Ltd Turbo Codes 2000[EB/OL] URL [http://www.vocal.com/white\\_paper/CF-036](http://www.vocal.com/white_paper/CF-036) 2001.pdf 2003-04-15/2003-08-07

编辑 徐培红