# 基于FPGA的一类低密度奇偶校验码的实现

刘晓明 刘 强 鲁俊成

(重庆大学通信工程学院 重庆400045)

**摘 要** 本设计用可编程逻辑器件(FPGA)实现了一种低密度奇偶校验LDPC(Low Density Parity Check)码。本文所提到的LDPC码是采用并行编码和部分并行译码结构。同时本文采用的是一种系统码结构,这种码的最主要的优点就是它的生成矩阵能够很容易地从奇偶校验矩阵的一定变换而得到,这样,应用FPGA实现译码器的同时,能够简单有效地实现对应的编码器。该设计是针对分组块长为345比特,码率为4/5,采用了6位量化方案。本文用现场可编程门阵列(FPGA)实现了LDPC码的编码,译码电路,并且通过QUARTUS仿真测试以及下载到实验板ATERA芯片的调试,表现出好的纠错性能。

**关键词** 低密度奇偶校验码,置信传播算法,系统码,部分并行译码,变量节点单元,校验节点单元

## FPGA Implementation of a Class of LDPC Encoder and Decoder

LIU Xiao-Ming LIU Qiang LU Jun-Cheng

(College of Communication Engineering, Chongqing University, Chongqing 400045)

**Abstract** We have implemented a class of Low Density Parity Check(LDPC)codes with FPGA technique. The codes considered here are based on parallelly concatenated parity check encoding and partly parallelly decoding structure. We have used the system code, and a major advantage of these codes is that the generator matrix can be got from the parity check matrix easily, which leads to efficient FPGA implementations for the encoder and the decoder. Our designs use 6-bit quantizaton with a code rate of 4/5 and a block size of 345 bits. We have applied FPGA technique to implement LDPC encoding and decoding circuit and performed QUARTUS simulation test, then downloaded it to breakboard construction with ATERA chip for hardware debugging, which has shown a good error-correct performance.

**Keywords** LDPC (Low Density Parity Check), BP algorithm, Sytem code, Partly parellel decoder, VNU(Variable Node Unit), CNU(Check Node Unit)

## 0 引言

低密度奇偶校验码LDPC(Low Density Parity Check)码由于其接近香农限的性能最近引起了人们极大的兴趣。低密度校验码又称为哥拉格码[1],它是哥拉格于1962年提出的一种性能接近香农限的好码。在很长的一段时间里,LDPC码并未受到人们的重视。直到1993年,Berrou等提出了Turbo码后,人们研究发现Turbo码其实就是一种LDPC码,LDPC码又重新引起了人们的研究兴趣。1996年,MacKay的研究,使LDPC码的研究跨入了一个新的阶段。最近几年的研究表明,在非规则图上构造的基于GF(q)域上的LDPC码性能要好于Turbo码,它的性能非常接近香农限。LDPC码是根据稀疏随机图来构造的,其码子之间具有很好的码距离。LDPC码属于线性纠错码,它的校验矩阵是一个稀疏校验阵,每个码子满足一定数目的线性约束,而约束的数目通常是非常小的。目前,LDPC码已经成为编码领域的一个新的研究热点[2]。

然而,LDPC编码器可能会变得很复杂。事实上当奇偶校验矩阵很稀疏时,对应的生成矩阵通常不会再是稀疏的,而编码器的设计由生成矩阵决定。最近,一种新的LDPC码采用了并行奇偶校验编码结构[3],这意味着编码器和译码器都有简化结构,因此,这些码使得VLSI实现已经成为可能。

国际电信联盟(ITU)已经提出了一种用于ADSL的高速率二进制LDPC码[4],这种LDPC码表现出非常接近香农限的性能。通过结合系统码的形式[5],能够使对应的生成矩阵非常简单,而编码是由生成矩阵决定,从而大大简化了编码的难度,使得这种LDPC码能够较容易用VLSI实现。

## 1 LDPC码的校验矩阵 $H$ 与生成矩阵 $G$

LDPC码实际上是一种线性分组码,其编码器由其生成矩阵决定,译码器由奇偶校验矩阵决定。本次设计选择了一个相对较小的奇偶校验矩阵 $H$,$H=[E_1|A]_{69\cdot 345}$。其中 $E_1$ 为单位矩阵,这里选的是69阶单位矩阵;$A$ 是一个69 * 276的稀疏矩阵,

$$A=\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{k-1} \\ 1 & \alpha^2 & \alpha^4 & \cdots & \alpha^{2(k-1)} \end{bmatrix}$$

$$\alpha=\begin{bmatrix} 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ \vdots & \vdots & \cdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & 0 & \cdots & 0 & 0 \end{bmatrix}$$

这里整数值 $\kappa=12$,$I$ 为23阶单位矩阵,$\alpha$ 为 $A$ 的一个23 * 23阶子矩阵。可见矩阵 $H$ 是一个69 * 345的稀疏的奇偶校验矩阵。因此可以很容易地得到对应的生成矩阵 $G=[A^T|E_2]_{276\cdot 345}$。$A^T$ 为矩阵 $A$ 的转置矩阵,$E_2$ 为276阶单位矩阵。

LDPC码除了作为线性分组码而用其生成矩阵和校验矩阵表示外,还可以用二分图(Bi-Partite graph)进行表示和分析。二分图中,有两类节点:校验节点(Check node),其每个节

刘晓明 副教授、博士后,主要从事测控系统数字化、软化、智能化和软件无线电的研究。刘 强 硕士研究生,主要从事测控系统数字化、软化、智能化。鲁俊成 硕士研究生,主要研究方向为软件无线电以及信道中的编译码技术。

点表示码字的一个校验集；变量节点(Variable node)，其每个节点表示码字的一个比特位。

## 2 并行编码器

本次对 LDPC 码的设计，除了设计其译码器外，还要设计其编码器。在 LDPC 码编码器的设计中，对于随机构造的 LDPC 码，其编码器的设计非常复杂。对于一般的 LDPC 码，设 $x$ 为编码器输出序列(列向量)，$H_1$ 为 $H$ 矩阵经过行变换得到的满矩阵，$P$，$Q$，$Q^{-1}$ 为满矩阵，则 $x$ 必须满足 $Hx=0$，得到 $H_1x=0$，$PH_1(QQ^{-1})x=0$，即 $PH_1Q(Q^{-1}x)=0$，设 $y=Q^{-1}x$，$A=PH_1Q$，而 $A=[E|D]$，$E$ 为单位矩阵，则有 $Ay=0$，$A$ 实际上是一种系统码的形式，很容易对 $A$ 进行编码从而得到 $y$，而 $x=Qy$，从而得到 $x$，作为编码器最后的输出。

具体分析该 LDPC 码的校验矩阵 $H$ 很容易得到其对应的生成矩阵 $G$，编码器是根据生成矩阵来实现的。本设计实际上利用系统码的形式简化了编码器的设计，避开了对校验矩阵的复杂变换。

将前面的生成矩阵 $G$ 影射在硬件 FPGA 中实现编码，这里采用了并行的编码结构[5](见图1)。该编码器包含5个模块，输入缓冲存储器用于缓存输入的信息位。每一个标为 $\Pi_i$ 的块根据生成矩阵指定的规则重新排列输入的信息位，由于生成矩阵的规律性，直接用 VHDL 和原理图连接生成该部分。每个标为奇偶校验的块将输入的信息位进行异或运算。输出缓冲存储器用于缓存生成的校验位。并串转换逻辑将并行的校验位连同信息位按一定的顺序转换成为串行的码字作为信道的输入进行传送。

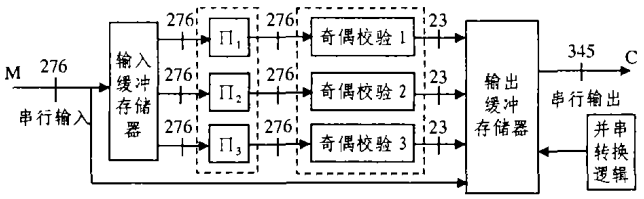


图1 编码器框图

## 3 部分并行译码器

### 3.1 Log-BP 算法

LDPC 码能够有效地应用置信传播算法(BP 算法)译码。由于大量的乘法运算，直接采用 BP 算法会导致很高的硬件复杂性，因此采用了对数(Log)运算将乘法变换为加法，这种算法叫做 Log-BP 算法。实际上，BP 和 Log-BP 算法实现的是同样的译码准则。在提出 Log-BP 算法之前，先介绍一些必要的规定：

$H$ 表示 $M*N$ 奇偶校验矩阵，$H_{i,j}$ 表示位置 $(i,j)$ 的矩阵元素值(0或1)。与第 $m$ 个奇偶校验节点有直接联系的变量节点 $n$ 表示为 $N(m)=\{n:H_{m,n}=1\}$，与变量节点 $n$ 有直接联系的校验节点 $m$ 表示为 $M(n)=\{m:H_{m,n}=1\}$。而 $N(m)\backslash n$ 表示为 $N(m)$ 中去掉第 $n$ 个变量节点后剩下的变量节点，同样，$M(n)\backslash m$ 表示为 $M(n)$ 中去掉第 $m$ 个校验节点后剩下的校验节点。

Log-BP 算法如下[6~8]：

**输入**：先验概率信息
$p_n^0=P(y_i|x_n=0)$ 以及 $p_n^1=P(y_i|x_n=1)$，$n=1,\cdots,N$；

**输出**：译码输出 $\hat{x}=\{\hat{x}_1,\hat{x}_2,\cdots,\hat{x}_N\}$；(与编码前的信息有可能不一样)；

①初始化：对每一个接收到的信息位 $n$，计算信道信息 $\gamma_n=\log(p_n(0)/p_n(1))$，对每一对在 $\{(i,j)|H_{i,j}=1\}$ 的 $(m,n)$，计算

$\alpha_{m,n}=sign(\gamma_n)\log(\frac{1+e^{-|\gamma_n|}}{1-e^{-|\gamma_n|}})$，这里 $sign(\gamma_n)=\begin{cases}+1,\gamma_n\geqslant 0\\-1,\gamma_n<0\end{cases}$

②迭代译码：

a、校验节点单元(CNU)处理：对每一校验节点 $m$，计算

$\beta_{m,n}=\log(\frac{1+e^{-\alpha}}{1-e^{-\alpha}})\prod_{n'\in N(m)\backslash n}sign(\alpha_{m,n'})$ 这里 $\alpha=\sum_{n'\in N(m)\backslash n}|\alpha_{m,n'}|$

b、变量节点单元(VNU)处理：对每一变量节点 $n$，计算

$\alpha_{m,n}=sign(\gamma_{m,n})\log(\frac{1+e^{-|\gamma_{m,n}|}}{1-e^{-|\gamma_{m,n}|}})$ 这里 $\gamma_{m,n}=\gamma_n+\sum_{m'\in M(n)\backslash m}\beta_{m',n}$

同时，更新后验对数似然比率：$\lambda_n=\gamma_n+\sum_{m\in M(n)}\beta_{m,n}$

③译码判决

a、由得到的 $\lambda=\{\lambda_1,\lambda_2,\cdots,\lambda_N\}$，执行硬判决，得到 $\hat{x}=\{\hat{x}_1,\hat{x}_2,\cdots,\hat{x}_N\}$，如果 $\lambda_n>\cdots 0$，则判 $\hat{x}_n=0$，否则判 $\hat{x}_n=1$。

b、如果 $H*\hat{x}=0$，则迭代运算停止，否则，继续到(②、迭代译码)进行迭代。如果预定的最大迭代次数出现(在本次设计中设为39)，则说明译码失败。

上面的 $\alpha_{m,n}$ 和 $\beta_{m,n}$ 分别叫作变量节点到校验节点的进化信息和校验节点到变量节点的进化信息。算法中迭代步骤中 CNU 部分的函数 $f(x)=\log(\frac{1+e^{-|x|}}{1-e^{-|x|}})$，分别利用查表公式1：$f_1(x)=6*(-\log(|\tanh(x/4)|))$(通过 VNU 中的查找表 LUT1实现)和查表公式2：$f_2(x)=-\log(|\tanh(x/16)|)$(通过 CNU 中的查找表 LUT2实现)来实现，借以扩大函数 $f(x)$ 输出的动态范围[9]。

### 3.2 译码器实现框图

称该译码器结构是部分并行的，是因为该译码器中的这些校验节点单元(CNU)和变量节点单元(VNU)的操作是时分复用的[10]。同时有多个 CNU 或 VNU 在并行处理，而每个 CNU 或 VNU 又是时分复用的(23个流水线操作)，称之为部分并行译码器。其结构框图见图2。

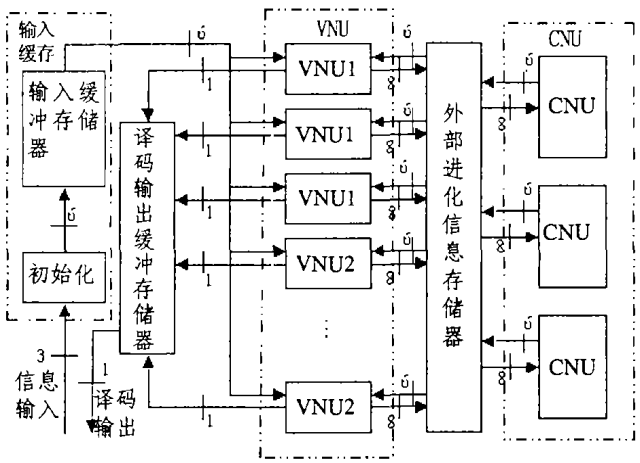


图2 部分并行译码器结构框图

3.2.1 *初始化信息* 信息经过编码送给调制解调器，发射的数据经由信道传输过来，经过同步和信道估计，针对二进制输入，八进制输出的 DMC 信道，根据 CSI 信息对数据进行 $Q=8=2^3$(3bits)电平量化，将量化后的信息送入该 LDPC 译码器。CSI 的信息由信道的高斯分布特性决定。一般情况下的转移概率[11]如表1所示。

表1 信道的转移概率表

| ri | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
|---|---|---|---|---|---|---|---|---|
| 0 | 0.434 | 0.197 | 0.167 | 0.111 | 0.058 | 0.023 | 0.008 | 0.002 |
| 1 | 0.002 | 0.008 | 0.023 | 0.058 | 0.111 | 0.167 | 0.197 | 0.434 |

表中 ri 是信道输入信息经过8电平量化器后的3bit 数据。根据 $\gamma_n=\log(p_n(0)/p_n(1))$，得到6位初始化信息 $\gamma_n$(这里认为 $\gamma_n$ 为本文中所提到的初始化信息)，其中包含符号位。初始化信息被存储在输入缓冲存储器中。

3.2.2 *输入缓冲存储器* 在译码过程中，会有连续的3

帧数据共存，上一帧数据的译码判决结果正在串行地作为译码输出，当前帧的数据正在内部做不断的迭代运算，正在输入的一帧数据串行地流入输入缓冲存储器中。因此在输入缓冲存储器中，必须能够同时存储2帧的数据，也就是需要345 * 2个6比特的数据。本次设计选用了15个RAM，每个RAM存储23 * 2个6比特的数据，以23个存储单元为一组分为两组交替地存储正在输入的来自信道的初始化信息，同时另一组用来存储正在进行迭代运算的初始化信息。

3.2.3 变量节点单元(VNU) 在该译码器设计中，本次设计用到了15个VNU单元(包括3个VNU1和12个VNU2)，每个VNU生成了对应变量节点的硬判决和该变量节点到校验节点的进化信息。VNU是进行时分复用操作的[10]，在一次迭代运算中一个VNU需要对23个变量节点进行运算。度数为3的变量节点所对应的VNU2结构见图3。在VNU2中，用到了查表运算LUT1，用以计算查表公式1，查表运算是通过VHDL生成的。度数为1的VNU1结构框图与此类似。

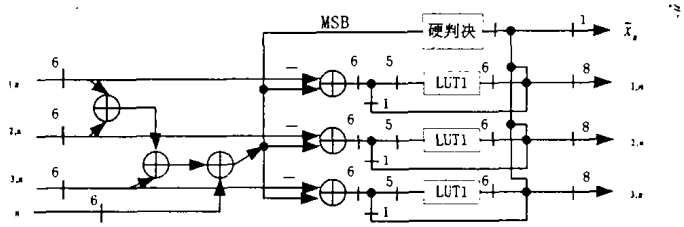


图3 度数为3的VNU结构框图

3.2.4 校验节点单元(CNU) 每一个CNU单元所执行的运算操作，包括进行奇偶校验，以及对校验节点到变量节点进化信息的计算。在设计中，用到了3个CNU块，每个CNU与前面讲到的VNU相似，都是时分复用的[10]，每个CNU在一次迭代中要对23个校验节点进行运算。度数为13的校验节点所对应的CNU结构框图见图4。在CNU中，用到了查表运算LUT2用以实现查表公式2，查表运算是通过VHDL生成的。

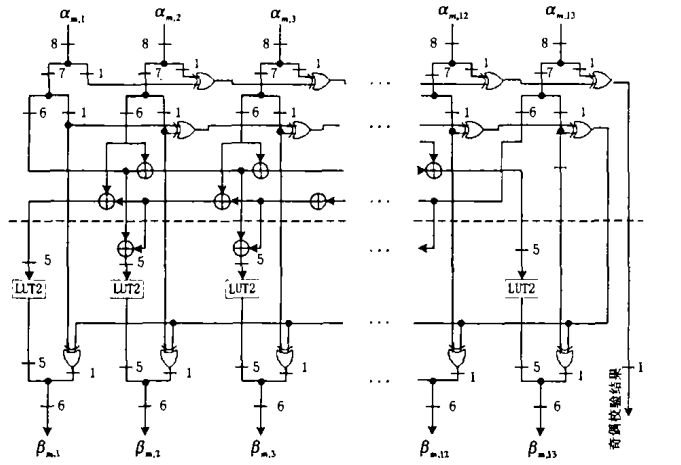


图4 度数为13的CNU结构框图

3.2.5 外部进化信息存储器 变量节点单元VNU和校验节点单元CNU都要对进化信息存储器进行读写操作。进化信息存储器采用的是双端口存储器(LPM_RAM_DP)，可以同时进行读写操作，在一次迭代运算中，VNU从进化信息存储器中读出数据，经过3个时钟周期后，VNU完成对变量位到校验位进化信息的计算，输出结果写入对应的进化信息存储器单元。VNU完成处理后，CNU从进化信息存储器中并行读出所需的13个数据，3个时钟周期后，CNU完成了奇偶校验和对校验位到变量位进化信息的计算，再将该信息写入对应的进化信息存储器单元，奇偶校验结果用以判断迭代运算是否终止。写地址是将读地址延时3个周期而得到的。分别控制VNU和CNU的读初始地址(用LPM-counter)，实现校验矩阵 $H$ 在FPGA中的影射。

3.2.6 译码输出缓冲存储器 译码器的输出是串行连续的，一帧接一帧地输出硬判决，同时，内部不断地迭代运算使下一帧的硬判决在不断地更新，迭代结束后，最后更新的硬判决将作为译码输出。可见，和输入缓冲存储器类似，需要两部分存储块，一部分用于当前帧正在向外输出，一部分用于下一帧硬判决的更新；每一存储块的更新或输出功能不是固定的，而是轮流用于输出和更新的。在向外输出之前，还需要一个并串转化电路将最后更新的硬判决转化为串行的比特位输出。本次设计用逻辑单元实现了该部分存储器。

## 4 实验结果分析

整个编码器是在MAXPLUSII开发软件下设计完成的，从编码器的仿真波形可以看到，编码后的信息位与编码前的输入一致，显然为系统码。整个译码器是在QuartusII开发软件下设计完成的，整个设计可以下载到ATERA公司的APEX20KE系列的EP20K300EQC240-1集成块上实现。在观测到的示波器波形中(见图5)(系统内部时钟由外部40M晶振二分频得到)，从上到下，分别是cp6f(300ns为内部时钟6分频得到，作为串行译码输入时钟)、view [2]、view [1]、view [0](为了观察译码前后而从输入缓冲存储器中延迟读出并进行反查表，使6位初始化数据逆变为3位译码输入软信息)、dec1(译码输出)、cp8f(400ns为内部时钟8分频得到，作为串行译码输出时钟)、rst(复位端)。为了较快地验证该设计，以编码为基础，将1帧276位的信息经过编码后，得到编码后的数据，以这编码后的数据为参考，事先将一帧含有20个突发错误和一帧含有20个随机错误的信道初始化信息存储在输入缓冲存储器中，以便不断地读出进行迭代译码。为了便于观察译码前后的数据，将所存储的译码输入数据以时钟cp8f读出(所读出的值为view [2..0])并与对应的译码输出位对齐。图5所示的波形只是纠正突发错误帧的一小部分时序图，经过仔细观察分析了所有的波形，与理论上所要达到的(即译码输出正好是编码输入)完全一致，实现无差错译码。从QuartusII的仿真波形以及下载到芯片后示波器观测到的波形分析表明该译码器能够纠正20个突发错误或20个随机错误(采用了不同的编码信息进行了大量的实验)。

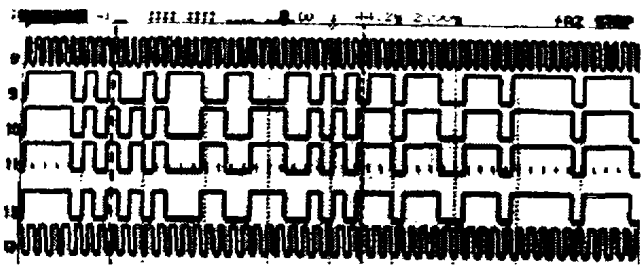

图5 示波器观察到的波形

在实验过程中，解调后的分层量化对译码器的性能有很大的影响，由于采用了3比特量化，量化级数应选为8，量化范围应作适当的选择。如果量化数据为7，表明是'1'的可能性最大，如果量化数据为0，表明'0'的可能性最大，错误的情况，多

由7错成3，或0错成4。同样在译码器初始化时也是非常关键的，可以根据实际情况作适当的修改。对初始化的修改只需对查表逻辑进行修改。

**结论** 采用部分并行译码结构以及系统码的并行编码方式，从软件仿真和硬件调试来看，设计出的LDPC的性能如下：工作时钟可达28M，译码输出比特率可达3.5Mbits，迭代次数为39次。减少迭代次数到19，译码输出比特率可达到7Mbits，还可以通过提高迭代处理时钟频率来提高最大迭代次数。可分别对20个突发错误和20个随机错误进行纠正。整个电路占用的逻辑单元7510LEs，占用15744ESB位。

## 参考文献


1 Gallager R G. Low-Density Parity-Check Codes. MLTPress, 1963. Available at:http://justice.mit.edu/people/gallager.html

2 孙韶辉，慕建君，王新梅. 低密度校验码研究及其新进展. 西安电子科技大学学报，2001，28(3)：393～397

3 Oenning T，Moon J. A Low-Density Generator Matrix Interpretation of Parallel Concat-enated Single Bit Parity Codes. IEEE Trans. on Magnetics，2001，37(2)：737～741

4 G. gen：LDPC codes for G. dmt. bis and G. lite. bis. Temporary Document CF-060，STUDY GROUP 4/15，Clearwater，Florida，Jan. 2001

5 Kim S，et al. Parallel VLSI architectures for a class of LDPC codes. Circuits and Systems，2002. In：ISCAS 2002. IEEE Intl. Symposium on，Volume 2，2002. 93～96

6 Zhang T，Parhi K K. VLSI implementation-oriented (3，k)-regular low-density parity-check codes. Signal Processing Systems. In：2001 IEEE Workshop on，Sept. 2001. 25～36

7 Zhang T，Wang Z，Parhi K K. on finite precision implementation of low density parity check cides decier. Circuits and Systems，2001. ISCAS 2001. In：The 2001 IEEE Intl. Symposium on，Volume：4，May 2001. 202～205

8 MacKay D J C. Good error-correcting codes based on very sparse matrices. IEEE Transactions on Information Theory，1999，45：399～431

9 Yeo E. Low Density Parity Check Code (OUTER)Decoder. Available at：http://bwrc.eecs.berkeley.edu/People/Grad_Students/yeo/ee225c/ldpc.htm

10 Parhi K K. VLSI Digital Signal Processing Systems：Design and Implementation. John Wiley & Sons，1999

11 陈宗杰，左孝彪. 纠错编码技术. 人民邮电出版社，1987


（上接第96页）

用户对检索出的文档索引的点击情况（点击率：$C$），可以将其组织成这一网页的特征向量 $\boldsymbol{V}$：

$$\boldsymbol{V}=(R,P,U,C) \tag{9}$$

不同的网页文档种类对不同的网页特征有不同的权值 $W_{ij}$，这些权值组成矩阵 $\boldsymbol{W}$，然后计算网页的类型向量[8]：

$$\boldsymbol{T}=\boldsymbol{W} * \boldsymbol{V} \tag{10}$$

最后可以根据 $\boldsymbol{T}$ 来判断网页的类型，需要说明的是：网页的特征权值 $W_{ij}$ 可以经过统计或已有资料确定，一个网页可以同时属于多种类型。

## 4 基于RPUC的索引库的更新算法描述

根据以上的分析，可以利用以下的算法计算出各网页的基于RPUC的类型向量，然后根据类型向量对网页进行分类，最后结合实际统计的各类网页的变化情况确定各类网页的更新周期。

**Step 1** 初始化（计算网页的特征向量 $v$，注意在建立索引数据库时计算出的信息可直接使用，如 $P$）

**Step 2** for($i$=0；$i$<文档索引数；$i$++)
{计算第 $i$ 个文档索引的类型向量；
switch（第 $i$ 个文档索引的类型向量）
{将索引 $i$ 归入相应的文档索引库的更新队列；}}

**Step 3** for($j$=0；$j$<更新队列中的文档索引数；$j$++)
{更新第 $j$ 个文档索引；}

Step 1中，网页的Page Rank值和更新频度在进行文档索引时可以（或已经）由索引程序直接计算得出，用户的检索率和点击率是随着用户的检索行为动态更新的，所以不会对索引数据库进行有效性验证的时间产生太大的影响。

Step 2中，主要操作是对文档索引类型向量的计算，在此基础上将不同类型向量的索引归入不同的更新队列。

Step 3中，系统针对不同的更新队列，启动不同周期的更新进程。

算法的时间复杂度为：$O(N\log_2 N)$。该算法在充分利用了Web页的索引信息的基础上，结合对用户检索行为的分析，对需要更新的文档索引进行分类，实现索引数据库的智能更新。改进了统一更新策略周期长、单一更新策略可能产生改变频繁而非常重要的网站长期又得不到更新的问题。

**结束语** 本文提出的搜索引擎中索引数据库的更新算法，综合考虑了用户对文档的兴趣度、文档本身的重要程度以及文档的更新频率，利用文档索引时的相关信息和后天对用户检索时的一些相关信息的统计，确定了文档索引更新的优先队列，既保证了变化的索引信息更新的时效性，也考虑了文档本身的重要性和用户的兴趣度，对索引数据库的更新来说应该是一种有效的方法。

## 参考文献


1 Cho. Crawling the Web：Discover and Maintenance of Large-Scale Web Data：[PhD. dissertation]. Stanford university，2001

2 Brin S，Page L. The Anatomy of a Large-Scale Hyper textual Web Search Engine. Stanford，CA 94305，USA

3 Kleinberg J. Authoritative sources in a hyperlinked environment. In：Tarjan R E，et al.，eds. Proc. of the 9th ACM-SIAM Symposium on Discrete Algorithms，New Orleans：ACM Press，1997

4 王晓宇，周傲英. 万维网的链接结构分析及其应用综述. 软件学报，2003，14(10)

5 陈治平，林亚平. 基于最高响应比的WWW索引库更新方法. 计算机科学，2003，30(5)

6 李盛韬，余智华，程学旗，白硕. Web信息采集研究进展. 计算机科学，2003，30(2)

7 彭洪汇，林作铨. Internet上的搜索引擎和元搜索引擎. 计算机科学，2002，29(9)

8 李剑，金蓓弘. Web链接结构信息研究综述. 计算机科学，2003，30(4)