一种(41, 21, 9)平方剩余码的快速代数译码算法

吴 怡* 罗春兰 张新球 林 潇 徐哲鑫
(福建师范大学福建省光电传感应用工程技术研究中心 福州 350007)

摘 要：为了降低译码时的计算复杂度以及减少译码时间，该文通过对牛顿恒等式进行推导得到了(41, 21, 9) QR码不需要计算未知校验子就可求得错误位置多项式系数的代数译码算法，同时也针对改善部分客观地给出了计算复杂度的理论分析。此外，为了进一步降低译码时间，提出判定接收码字中出现不同错误个数的更简化的判断条件。仿真结果表明该文提出算法在不降低Lin算法所达到的译码性能的前提下，降低了译码时间。

关键词：平方剩余码；代数译码；牛顿恒等式；未知校验子；错误位置多项式
中图分类号：TN911.22 **文献标识码**：A **文章编号**：1009-5896(2018)08-1949-07
DOI: 10.11999/JEIT170983

A Fast Algebraic Decoding of the (41, 21, 9) Quadratic Residue Code

WU Yi LUO Chunlan ZHANG Xinqiu LIN Xiao XU Zhexin
(*Fujian Provincial Engineering Technology Research Center of Photoelectric Sensing Application, Fujian Normal University, Fuzhou* 350007, *China*)

Abstract: In order to reduce the computational complexity of computing unknown syndromes for the coefficients of the error-locator polynomial and reduce the decoding time when one is decoding, this paper proposed an algebraic decoding algorithm of (41, 21, 9) QR code without calculating the unknown syndromes by solving the Newtonian identity. Simultaneously, an objective theoretical analysis of the computational complexity is given for the part of improvement. Besides, this paper also puts forward the simplifying conditions to determine the number of errors in the received word, which in order to further reducing the decoding time. Simulation results show that the proposed algorithm reduces the decoding time with maintaining the same decoding performance of Lin's algorithm.

Key words: Quadratic residue code; Algebraic decoding; Newtonian identity; Unknown syndrome; Error-locator polynomial

1 引言

平方剩余(Quadratic Residue, QR)码的概念是Prang[1]在1957年发表的一篇报告中首次提出的。QR码是循环(Bose, ray-Chaudhuri and Hocquenghem, BCH)码的一个重要子类，它不仅具有丰富并且严谨的代数结构，还拥有较大的最小汉明距离以及接近于1/2的码率。比如(23, 12, 7) QR码[2](也称为Golay码)是一个完备码，能够纠正23位分位码组中的最多3个错误。在过去，最为广泛使用的代数译码算法是用Sylvester结子[3]，Gröbner基[4]的方法来计算牛顿恒等式确定错误位置多项式。最后再使用Chien[5]搜索法找出错误位置多项式的根，从而完成译码。此外，也有利用查表法[6]快速求解QR码，但需要使用额外的内存，最近也有利用QR码的循环特性不用计算错误位置多项式来直接译码[7]，这两种方法都没有涉及错误位置多项式。由于不同码长QR码的相关未知校验子都不相同，所以到目前为止还没有一个通用的代数译码方法对所有的QR码进行译码。比如(47, 24, 11)QR码[8]以及(71, 36, 11)QR码[9–11]，均能纠正5个错，但其未知校验子分别为S_5和S_7。在译码时为了求得错误位置多项式的系数，文献[8]及文献[9]分别提出了当判定接收码字中出现4个错的时候求解错误位置多项式的系数避免了未知校验子的代数译码算法。

在1992年，Reed等人[3]首次提出了对(41, 21, 9)QR码的一种代数译码方法，即利用Sylvester结子的方法去计算牛顿恒等式从而找到错误位置多项

收稿日期：2017-10-23；改回日期：2018-04-23；网络出版：2018-05-30
*通信作者： 吴怡 wuyi@fjnu.edu.cn
基金项目：国家自然科学基金(61571128, 61701118)
Foundation Items: The National Natural Science Foundation of China (61571128, 61701118)

式，再求解错误位置多项式的解，找到错误位置，修正错误达到译码的目的。但是，可以发现文献[3]中当判定接受码字中出现错误的个数为3个或者4个的时候，在译码时需要很高的计算复杂度和较大的存储空间。为了改善这个问题，Lin等人[12]提出了对(41, 21, 9)QR码的一个有效的代数译码方法，称为Lin算法。该算法是利用Feng等人[13]提供的方法，即根据(41, 21, 9)QR码的已知校验子去计算未知校验子S_3，从而求得错误位置多项式的系数，然后求解出错误的位置进而完成译码。实验仿真表明Lin算法在计算未知校验子S_3时需要很长的时间。鉴于此，本文提出了(41, 21, 9)QR码一个新的快速代数译码算法。当判定接收码字中出现错误个数分别为2, 3, 4个时，通过对牛顿恒等式进行数学推导，得出了不需要计算未知校验子就能求得相关的错误位置多项式系数的代数译码算法，该算法可以降低译码时的计算复杂度。此外，本文还提出了判定接收码字中出现不同错误个数的更加简单的判断条件，从而能够进一步降低在译码时的仿真时间。将所有的码字经过BPSK调制后传输到加性高斯白噪声(AWGN)信道中，仿真结果表明，本文提出的算法的性能与原有算法的性能一致，然而当判定接收码字中出现的错误个数为1, 2, 3, 4个的时候，译码时间分别减少了83.72%, 62.99%, 15.92%, 14.98%。

2 (41, 21, 9)QR码

在有限域GF(2^m)上，(n, k, d)QR码的码长质数n是由$n=8l\pm1, l\in N$得到的，其中m是满足$n|(2^m-1)$的最小正整数；$k=(n+1)/2$表示的是该码的信息长度；最大纠错能力$t=\lfloor(d-1)/2\rfloor$是由最小汉明距离d决定的。令Q_n为QR码的平方剩余集，即$Q_n=\{j|j\equiv x^2 \bmod n, 1\le x\le(n-1)\}$。因此，(41, 21, 9)QR码是在有限域GF(2^{20})上构建的，纠错能力为4个错的一种QR码，其平方剩余集为

$$Q_{41}=\{1,2,4,5,8,9,10,16,18,20,21,23,25,31,32,33,36,37,39,40\} \tag{1}$$

假设α为有限域GF(2^{20})的本原元，则$\beta=\alpha^u$，其中$u=(2^m-1)/n=(2^{20}-1)/41=25575$，是GF($2^{20}$)上的一个本原41次的单位根。于是(41, 21, 9) QR码的生成多项式可表示为

$$g(x)=\prod_{i\in Q_{41}}(x-n^i)=x^{20}+x^{19}+x^{17}+x^{16}+x^{14}+x^{11}+x^{10}+x^9+x^6+x^4+x^3+x+1 \tag{2}$$

假设输入的信息多项式为$m(x)=\sum_{i=0}^{20}m_ix^i$，其中$m_i\in$ GF(2)，接收码字$c(x)$是生成多项式$g(x)$与信息多项式$m(x)$的乘积，即$c(x)=m(x)g(x)=\sum_{i=0}^{40}c_ix^i$，其中$c_i\in$ GF(2)。码字$c(x)$经过AWGN信道，受到噪声$e(x)$干扰后的接收码字为

$$r(x)=c(x)+e(x)=\sum_{i=0}^{40}c_ix^i+\sum_{i=0}^{40}e_ix^i \tag{3}$$

定义已知校验子：

$$S_i=r(\beta^i)=e(\beta^i)=e_{40}(\beta^i)^{40}+e_{39}(\beta^i)^{39}+\cdots+e_1(\beta^i)+e_0 \tag{4}$$

其中，$i\in Q_{41}$。如果$i\notin Q_{41}$，S_i则称为未知校验子，例如$3\notin Q_{41}$，所以S_3是(41, 21, 9)QR码的未知校验子。除此之外，如果接收码字中发生的错误个数为奇数，$S_0=1$；否则$S_0=0$。

若(41, 21, 9)QR码的码字在传输的过程中受到干扰时发生错误的个数$v\le4$，则错误多项式$e(x)$为

$$e(x)=x^{l_1}+x^{l_2}+\cdots+x^{l_v},\quad 0\le l_1<l_2<\cdots<l_v\le40 \tag{5}$$

根据式(4)和式(5)，可以得到校验子：$S_i=X_1^i+X_2^i+\cdots+X_v^i$，其中$X_j=\beta^{l_j}, 1\le j\le v$，记做接收码字中发生错误的错误位置。定义错误位置多项式，记为$L(z)$，表示如式(6)：

$$L(z)=\prod_{j=1}^{v}(z-X_j)=(z-X_1)(z-X_2)\cdots(z-X_v)=\sigma_0z^v+\cdots+\sigma_{v-1}z+\sigma_v \tag{6}$$

$L(z)$的系数可以表示为

$$\left.\begin{aligned}&\sigma_0=1\\&\sigma_1=X_1+X_2+\cdots+X_v\\&\sigma_2=X_1X_2+X_1X_3+\cdots+X_{v-1}X_v\\&\vdots\\&\sigma_v=X_1X_2\cdots X_v\end{aligned}\right\} \tag{7}$$

确定了错误位置多项式的系数后，求解出错误位置多项式$L(z)$，然后利用Chien搜索法找到$L(z)$的根，即是错误的位置，将此位置的错误改正过来，就完成了译码的目的。

3 (41, 21, 9)QR码的代数译码算法

在文献[12]的Lin算法中，当判定接收码字中的错误个数分别为2, 3, 4的时候，在计算相关的错误位置多项式的系数时需要计算未知校验子S_3，但是计算未知校验子非常复杂，占用内存多。所以，本

文提出了一种仅用已知校验子直接求得相关错误位置多项式系数的代数译码算法。

3.1 新的错误位置多项式的推导

为了能够得到(41, 21, 9)QR码的一种不需要计算未知校验子就可以求解出错误位置多项式的系数的算法，需要引进可以表明S_i和σ_j关系的牛顿恒等式即引理1和引理2，详细的证明可以分别参见文献[14]和文献[3]。

引理 1 **(牛顿恒等式)** 对于二进制QR码，校验子S_i和σ_j存在对称关系，即

$$\left.\begin{aligned} &S_i+\sum_{j=1}^{i-1}\sigma_j S_{i-j}+\sigma_j=0, 1\le i\le v, \ i\text{为奇数}\\ &S_i+\sum_{j=1}^{i-1}\sigma_j S_{i-j}=0, 1\le i\le v, \qquad i\text{为偶数}\\ &S_i+\sum_{j=1}^{v}\sigma_j S_{i-j}=0, \qquad i\ge v \end{aligned}\right\} \tag{8}$$

引理 2 当(n, k, d)QR码的接收码字中出现错误个数$v\ge 2$时，就有$S_{n-1}\sigma_v=\sigma_{v-1}$。

当(41, 21, 9)QR码的接收码字中出现的错误个数$2\le v\le 4$时，其不需要计算未知校验子的错误位置，多项式的系数可以从式(9)–式(23)的牛顿恒等式中获得。

$$S_1+\sigma_1=0 \tag{9}$$

$$S_3+\sigma_1S_2+\sigma_2S_1+\sigma_3=0 \tag{10}$$

$$S_5+\sigma_1S_4+\sigma_2S_3+\sigma_3S_2+\sigma_4S_1=0 \tag{11}$$

$$S_{20}+\sigma_1S_{19}+\sigma_2S_{18}+\sigma_3S_{17}+\sigma_4S_{16}=0 \tag{12}$$

$$S_{21}+\sigma_1S_{20}+\sigma_2S_{19}+\sigma_3S_{18}+\sigma_4S_{17}=0 \tag{13}$$

$$S_{22}+\sigma_1S_{21}+\sigma_2S_{20}+\sigma_3S_{19}+\sigma_4S_{18}=0 \tag{14}$$

$$S_{23}+\sigma_1S_{22}+\sigma_2S_{21}+\sigma_3S_{20}+\sigma_4S_{19}=0 \tag{15}$$

$$S_{24}+\sigma_1S_{23}+\sigma_2S_{22}+\sigma_3S_{21}+\sigma_4S_{20}=0 \tag{16}$$

$$S_{25}+\sigma_1S_{24}+\sigma_2S_{23}+\sigma_3S_{22}+\sigma_4S_{21}=0 \tag{17}$$

$$S_{34}+\sigma_1S_{33}+\sigma_2S_{32}+\sigma_3S_{31}+\sigma_4S_{30}=0 \tag{18}$$

$$S_{35}+\sigma_1S_{34}+\sigma_2S_{33}+\sigma_3S_{32}+\sigma_4S_{31}=0 \tag{19}$$

$$S_{36}+\sigma_1S_{35}+\sigma_2S_{34}+\sigma_3S_{33}+\sigma_4S_{32}=0 \tag{20}$$

$$S_{38}+\sigma_1S_{37}+\sigma_2S_{36}+\sigma_3S_{35}+\sigma_4S_{34}=0 \tag{21}$$

$$S_{39}+\sigma_1S_{38}+\sigma_2S_{37}+\sigma_3S_{36}+\sigma_4S_{35}=0 \tag{22}$$

$$S_{40}+\sigma_1S_{39}+\sigma_2S_{38}+\sigma_3S_{37}+\sigma_4S_{36}=0 \tag{23}$$

本文提出的避免未知校验子的快速译码算法中求解错误位置多项式的步骤具体如下：

(1) 当接收码字中出现错误个数v=2时，由式(9)–式(11)结合引理2，可以得到σ_1和σ_2的表达式为：$\sigma_1=S_1$, $\sigma_2=S_1/S_{40}$，所以当接到码字中出现错误个数v=2时，其错误位置多项式为

$$L_2(z)=z^2+\sigma_1z+\sigma_2=z^2+S_1z+S_1/S_{40} \tag{24}$$

(2) 当接收码字中出现错误个数v=3时，由式(9)–式(11)，式(18)–式(20)，式(22)和式(23)结合引理2，可以得到

$$\begin{aligned} &\sigma_1=S_1, \sigma_2=\sigma_1S_{40},\\ &\sigma_3=(B_{36}B_0+A_5B_{31})/A_{33}B_0 \end{aligned} \tag{25}$$

其中，$A_5=S_1^5+S_5$, $A_{33}=S_1^{33}+S_{33}$, $B_0=S_1S_{40}+1$, $B_{31}=S_1^{32}S_{40}+S_{31}$, $B_{36}=S_1^3S_{33}+S_{31}$，所以当接到码字中出现错误个数$v$=3时，其错误位置多项式为

$$\begin{aligned} L_3(z)=&z^3+S_1z^2+((B_{36}B_0+A_5B_{31})/A_{33}B_0)\\ &\cdot(S_{40}z+1) \end{aligned} \tag{26}$$

(3) 当接收码字中出现错误个数v=4时，由式(9)–式(11)可以得到σ_2与σ_4的关系式如式(27)：

$$\begin{aligned} &\sigma_2^2S_1+\sigma_2\left(\sigma_4S_{40}+S_1^3\right)+\sigma_4\left(S_1^2S_{40}\right.\\ &\quad\left.+S_1\right)+S_1^5+S_5=0 \end{aligned} \tag{27}$$

由式(12)–式(15)可以得到σ_2与σ_4的关系式如式(28)：

$$\begin{aligned} &\sigma_2^2(S_1S_{20}S_{40}+S_{21}S_{40})+\sigma_2\left(\sigma_4\left(S_{18}+S_{20}S_{40}^2\right)\right.\\ &\left.+S_1^2S_{20}+S_1S_{21}+S_1^2S_{21}S_{40}+S_{23}S_{40}\right)\\ &+\sigma_4^2\left(S_1^{16}+S_1^{17}S_{40}+S_{18}S_{40}^2+S_1S_{18}S_{40}^3\right)\\ &+\sigma_4\left(S_1^2S_{18}+S_{20}+S_1S_{20}S_{40}+S_1^2S_{20}S_{40}^2\right.\\ &\left.+S_{21}S_{40}+S_1S_{21}S_{40}^2\right)+S_1^3S_{21}+S_1S_{23}=0 \end{aligned} \tag{28}$$

由式(14)–式(17)可以得到σ_2与σ_4的关系式如式(29)：

$$\begin{aligned} &\sigma_2^2\left(S_1S_{20}+S_1S_{21}S_{40}\right)+\sigma_2\left(\sigma_4\left(S_1S_{18}+S_{20}S_{40}\right.\right.\\ &\left.+S_1S_{20}S_{40}^2+S_{21}S_{40}^2\right)+S_1^2S_{21}+S_{23}\left.\right)\\ &+\sigma_4^2\left(S_{18}S_{40}+S_{20}S_{40}^3\right)+\sigma_4\left(S_1S_{20}+S_1^2S_{20}S_{40}\right.\\ &+S_{21}+S_1S_{21}S_{40}+S_1^2S_{21}S_{40}^2+S_{23}S_{40}^2\left.\right)\\ &+S_1^2S_{23}+S_1^3S_{23}S_{40}+S_{25}+S_1S_{25}S_{40}=0 \end{aligned} \tag{29}$$

由式(19)–式(23)可以得到σ_2与σ_4的关系式如式(30)：

$$\begin{aligned} &\sigma_2^2\left(\sigma_4S_{33}+S_1S_{36}+S_{37}+S_1S_{37}S_{40}\right)\\ &+\sigma_2\left(\sigma_4^2\left(S_{31}+S_{32}S_{40}\right)+\sigma_4\left(S_{36}S_{40}\right.\right.\\ &\left.+S_1S_{36}S_{40}^2\right)+S_1^2S_{37}+S_{39}+S_1S_{39}S_{40}\left.\right)\\ &+\sigma_4\left(S_1^2S_{36}S_{40}+S_1^2S_{37}S_{40}^2\right)+S_1^2S_{40}^2\\ &+S_1^3S_{39}S_{40}=0 \end{aligned} \tag{30}$$

为了化简式(27)–式(30)中的σ_2和σ_4，定义：

$$a_{112}=S_1S_{31}+S_1S_{32}S_{40}+S_{33}S_{40} \tag{31}$$

$$a_{111}=S_1^3S_{33}+S_1^2S_{36}S_{40}^2+S_{37}S_{40}+S_1S_{37}S_{40}^2 \tag{32}$$

$$a_{110}=S_1^4S_{36}+S_1^4S_{37}S_{40}+S_1S_{39}+S_1^2S_{39}S_{40} \tag{33}$$

$$a_{102}=S_1S_{33}+S_1^2S_{33}S_{40} \tag{34}$$

$$a_{101}=S_1^5S_{33}+S_5S_{33}+S_1^2S_{36}+S_1S_{37} \tag{35}$$

$$\begin{aligned}a_{100}=&S_1^3S_{40}^2+S_1^6S_{36}+S_1S_5S_{36}+S_1^6S_{37}S_{40}\\&+S_1^5S_{37}+S_5S_{37}+S_1S_5S_{37}S_{40}+S_1^4S_{39}S_{40}\end{aligned} \tag{36}$$

$$a_{211}=S_1S_{18}+S_1S_{40}^2 \tag{37}$$

$$a_{210}=S_1^3S_{20}+S_1^4S_{20}S_{40}+S_1^2S_{21}+S_1S_{23}S_{40} \tag{38}$$

$$a_{202}=S_1^{17}+S_1^{18}S_{40}+S_1S_{18}S_{40}^2+S_1^2S_{18}S_{40}^3 \tag{39}$$

$$a_{201}=S_1^3S_{18}+S_1S_{20} \tag{40}$$

$$\begin{aligned}a_{200}=&S_1^6S_{20}S_{40}+S_1S_5S_{20}S_{40}+S_1^4S_{21}+S_1^5S_{21}S_{40}\\&+S_5S_{21}S_{40}+S_1^2S_{23}\end{aligned} \tag{41}$$

$$a_{311}=S_1S_{18}+S_1S_{20}S_{40}^2 \tag{42}$$

$$a_{310}=S_1^3S_{20}+S_1^2S_{21}+S_1^3S_{21}S_{40}+S_{23} \tag{43}$$

$$a_{302}=S_{20}S_{40}^3+S_1^{18}S_{40} \tag{44}$$

$$a_{301}=S_{21}+S_{23}S_{40}^2 \tag{45}$$

$$\begin{aligned}a_{300}=&S_1^5S_{20}+S_5S_{20}+S_1^5S_{21}S_{40}+S_5S_{21}S_{40}\\&+S_1^2S_{23}+S_1^3S_{23}S_{40}+S_{25}+S_1S_{25}S_{40}\end{aligned} \tag{46}$$

$$\begin{aligned}n_3=&a_{102}a_{211}^2a_{302}+a_{111}a_{202}^2a_{311}+a_{112}a_{202}^2a_{310}\\&+a_{102}a_{202}a_{211}a_{311}+a_{111}a_{202}a_{211}a_{302}\\&+a_{112}a_{201}a_{211}a_{302}+a_{112}a_{202}a_{210}a_{302}\\&+a_{112}a_{202}a_{211}a_{301}\end{aligned} \tag{47}$$

$$\begin{aligned}n_2=&a_{101}a_{211}^2a_{302}+a_{110}a_{202}^2a_{311}+a_{101}a_{202}a_{211}a_{311}\\&+a_{102}a_{202}a_{210}a_{311}+a_{102}a_{210}a_{211}a_{302}\\&+a_{110}a_{202}a_{211}a_{302}+a_{111}a_{201}a_{202}a_{311}\\&+a_{111}a_{201}a_{211}a_{302}+a_{112}a_{200}a_{211}a_{302}\\&+a_{112}a_{201}a_{202}a_{310}+a_{112}a_{202}a_{210}a_{301}\\&+a_{112}a_{200}a_{202}a_{300}\end{aligned} \tag{48}$$

$$\begin{aligned}n_1=&a_{100}a_{211}^2a_{302}+a_{100}a_{202}a_{211}a_{311}+a_{101}a_{202}a_{210}a_{311}\\&+a_{101}a_{210}a_{211}a_{302}+a_{110}a_{201}a_{202}a_{311}+\\&a_{110}a_{201}a_{211}a_{302}+a_{111}a_{200}a_{202}a_{311}\\&+a_{111}a_{200}a_{211}a_{302}+a_{112}a_{200}a_{202}a_{310}\\&+a_{112}a_{202}a_{210}a_{300}\end{aligned} \tag{49}$$

$$\begin{aligned}n_0=&a_{100}a_{202}a_{210}a_{311}+a_{100}a_{210}a_{211}a_{302}\\&+a_{110}a_{200}a_{202}a_{311}+a_{110}a_{200}a_{211}a_{302}\end{aligned} \tag{50}$$

$$m_3=a_{202}a_{311}+a_{211}a_{302} \tag{51}$$

$$m_2=a_{201}a_{311}+a_{202}a_{310}+a_{210}a_{302}+a_{211}a_{301} \tag{52}$$

$$m_1=a_{200}a_{311}+a_{201}a_{310}+a_{210}a_{301}+a_{211}a_{300} \tag{53}$$

$$m_0=a_{200}a_{310}+a_{210}a_{300} \tag{54}$$

联合式(27)–式(30)，根据定义式(31)–式(54)，可以得到一个关于σ_2和σ_4的表达式如式(55)和式(56)：

$$\sigma_2(\sigma_4a_{211}+a_{210})+\sigma_4^2a_{202}+\sigma_4a_{201}+a_{200}=0 \tag{55}$$

$$\sigma_4=A/B \tag{56}$$

其中，

$$\begin{aligned}A=&m_0n_1n_2m_3^2+n_0n_1m_3^3+m_0n_2^2n_3^2\\&+m_0n_2n_3m_2m_3+n_0n_2m_2m_3^2+m_0n_3^2m_1m_3\\&+n_0n_3m_1m_3^2+n_0n_3m_2^2m_3\end{aligned}$$

$$\begin{aligned}B=&n_1^2m_3^3+n_1n_2m_3^2+n_1n_3m_2^2m_3+n_2^2m_1m_3^2\\&+n_2n_3m_1m_2m_3+m_0n_2n_3m_3^2+n_0n_2m_3^3\\&+n_3^2m_1^2m_3+m_0n_3^2m_2m_3+n_0n_3m_2m_3^2\end{aligned}$$

根据引理2和式(56)可以得到σ_3为

$$\sigma_3=\sigma_4S_{40} \tag{57}$$

由式(55)，可以得到σ_2为

$$\sigma_2=\left(\sigma_4^2a_{202}+\sigma_4a_{201}+a_{200}\right)/(\sigma_4a_{211}+a_{210}) \tag{58}$$

当接到码字中出现错误个数v=4时，$\sigma_1=S_1$，σ_2, σ_3, σ_4分别由式(58)，式(57)和式(56)给出，所以其错误位置多项式为

$$L_4(z)=z^4+S_1z^3+\sigma_2z^2+\sigma_3z+\sigma_4 \tag{59}$$

通过以上对牛顿恒等式结合定理的数学推导，可以清楚地看出本文提出的(41, 21, 9)QR码的代数译码算法中，当接收码字中出现错误个数v=2, 3, 4的时候，其错误位置多项式的系数中均没有出现未知校验子。

3.2 简化v=1, 2, 3时的判断条件进一步优化前述的快速代数译码算法

在译码过程中，不同错误模式情况下的判断条件对译码时间的影响是非常大的，如果能改善出现错误时的判断条件，就能进一步降低译码时间。文献[12]的算法中，判定接收码字中出现错误个数v=1时的判断条件是$S_1^{41}=1$，而本文提出的对应的判断条件为$S_1{}^5=S_5$极大地降低了计算量；当接收码字中出现错误个数v=2或3时，Lin算法中是利用行列式的值是否为0来进行判断的，而本文提出了比Lin算法更加简化的行列式，可以降低译码仿真所需要的时间。本文提出算法的具体步骤如下：

(1)如果$S_1=0$，那么没有错误发生，$L_0(z)=0$；

(2)如果$S_1^5=S_5$，那么就有1个错误发生，$L_1(z)=z+S_1$；

(3)如果$\det(\boldsymbol{C}_3)=0$，那么就有2个错误发生，错误位置多项式为式(24)，其中，

$$\boldsymbol{C}_3=\begin{bmatrix} S_0 & S_8 & S_{40} \\ S_1 & S_9 & S_0 \\ S_{32} & S_{40} & S_{31} \end{bmatrix} \tag{60}$$

(4)如果$\det(\boldsymbol{C}_4)=0$，那么就有3个错误发生，错误位置多项式为式(26)，其中，

$$\boldsymbol{C}_4=\begin{bmatrix} S_1 & S_{18} & S_{25} & S_{33} \\ S_8 & S_{25} & S_{32} & S_{40} \\ S_{25} & S_1 & S_8 & S_{16} \\ S_{37} & S_4 & S_{23} & S_{31} \end{bmatrix} \tag{61}$$

(5)以上条件都不满足的情况下，则有4个错误发生，错误位置多项式为式(59)。

4　计算复杂度分析

为了说明本文提出不用计算未知校验子的代数译码算法的复杂度与Lin算法[12]相比有降低，现就针对$L_4(z)$分别对Lin算法和本文提出的算法的计算复杂度进行客观分析，因为在求解错误位置多项式时，相对于$L_2(z)$及$L_3(z)$的计算量，$L_4(z)$最大也最具代表性。

4.1　Lin算法$L_4(z)$的计算复杂度

当判定接收码字中有4个错时，Lin算法的错误位置多项式如式(62)：

$$\begin{aligned} L_4(z)=&z^4+\sigma_1z^3+\sigma_2z^2+\sigma_3z+\sigma_4=z^4+S_1z^3 \\ &+\frac{S_1A_7+S_3A_5}{S_1A_5+S_3A_3}z^2+(A_3+S_1\sigma_2)z \\ &+\frac{B_5+A_3\sigma_2}{S_1} \end{aligned} \tag{62}$$

其中，$A_3=S_1^3+S_3$，$A_5=S_1^5+S_5$，$A_7=S_1^7+S_7$，$B_5=S_1^2S_3+S_5$。

在求解错误位置多项式$L_4(z)$时，首先需要求解相关的未知校验子S_3，由文献[12]可知，S_3表达为

$$S_3=\det(\boldsymbol{C}_{40})/\det(\boldsymbol{C}_{41})=\det(\boldsymbol{C}_{40})\cdot[\det(\boldsymbol{C}_{41})]^{-1}$$

其中，

$$\boldsymbol{C}_{40}=\begin{bmatrix} S_0 & S_2 & S_8 & S_9 & S_{20} \\ S_1 & 0 & S_9 & S_{10} & S_{21} \\ S_{23} & S_{25} & S_{31} & S_{32} & S_2 \\ S_{31} & S_{33} & S_{39} & S_{40} & S_{10} \\ S_{37} & S_{39} & S_4 & S_5 & S_{16} \end{bmatrix} \tag{63}$$

$$\boldsymbol{C}_{41}=\begin{bmatrix} S_0 & S_8 & S_9 & S_{20} \\ S_{23} & S_{31} & S_{32} & S_2 \\ S_{23} & S_{39} & S_{40} & S_{10} \\ S_{37} & S_4 & S_5 & S_{16} \end{bmatrix} \tag{64}$$

其中，在v<4时已经计算出的已知校验子就作为已知量，当v=4时，这些已知校验子不再重复计算。并且是采用行(列)展开的方法进行运算的。

对$\boldsymbol{C}_{40}$的第2行展开，则有

$$\begin{aligned} \det(\boldsymbol{C}_{40})=&S_1\boldsymbol{C}_{401}+S_9\boldsymbol{C}_{402}+S_{10}\boldsymbol{C}_{403}+S_{21}\boldsymbol{C}_{404} \\ =&S_1\begin{bmatrix} S_2 & S_8 & S_9 & S_{20} \\ S_{25} & S_{31} & S_{32} & S_2 \\ S_{33} & S_{39} & S_{40} & S_{10} \\ S_{39} & S_4 & S_5 & S_{16} \end{bmatrix} \\ &+S_9\begin{bmatrix} S_0 & S_2 & S_9 & S_{20} \\ S_{23} & S_{25} & S_{32} & S_2 \\ S_{31} & S_{33} & S_{40} & S_{10} \\ S_{37} & S_{39} & S_5 & S_{16} \end{bmatrix} \\ &+S_{10}\begin{bmatrix} S_0 & S_2 & S_8 & S_{20} \\ S_{23} & S_{25} & S_{31} & S_2 \\ S_{31} & S_{33} & S_{39} & S_{10} \\ S_{37} & S_{39} & S_4 & S_{16} \end{bmatrix} \\ &+S_{21}\begin{bmatrix} S_0 & S_2 & S_8 & S_9 \\ S_{23} & S_{25} & S_{31} & S_{32} \\ S_{31} & S_{33} & S_{39} & S_{40} \\ S_{37} & S_{39} & S_4 & S_5 \end{bmatrix} \end{aligned} \tag{65}$$

对式(65)的第1项中的$\boldsymbol{C}_{401}$按第1行展开，则有

$$\begin{aligned} &\det(\boldsymbol{C}_{401}) \\ &=S_2\boldsymbol{C}_{4011}+S_8\boldsymbol{C}_{4012}+S_9\boldsymbol{C}_{4013}+S_{20}\boldsymbol{C}_{4014} \\ &=S_2\begin{bmatrix} S_{31} & S_{32} & S_2 \\ S_{39} & S_{40} & S_{10} \\ S_4 & S_5 & S_{16} \end{bmatrix}+S_8\begin{bmatrix} S_{25} & S_{32} & S_2 \\ S_{33} & S_{40} & S_{10} \\ S_{39} & S_5 & S_{16} \end{bmatrix} \\ &+S_9\begin{bmatrix} S_{25} & S_{31} & S_2 \\ S_{33} & S_{39} & S_{10} \\ S_{39} & S_4 & S_{16} \end{bmatrix}+S_{20}\begin{bmatrix} S_{25} & S_{31} & S_{32} \\ S_{33} & S_{39} & S_{40} \\ S_{39} & S_4 & S_5 \end{bmatrix} \end{aligned} \tag{66}$$

对式(66)的第1项中的$\boldsymbol{C}_{4011}$按第1行展开，则有

$$\begin{aligned} \det(\boldsymbol{C}_{4011})=&S_{31}\begin{bmatrix} S_{40} & S_{10} \\ S_5 & S_{16} \end{bmatrix}+S_{32}\begin{bmatrix} S_{39} & S_{10} \\ S_4 & S_{16} \end{bmatrix} \\ &+S_2\begin{bmatrix} S_{39} & S_{40} \\ S_4 & S_5 \end{bmatrix} \end{aligned} \tag{67}$$

在计算一个2阶行列式的时候，需要进行2次乘法，1次加法。由此，在计算式(67)即3阶行列式的时候，需要进行9次乘法，5次加法；然后，根据式(67)可以得到在计算式(66)即4阶行列式的时候，需要进行40次乘法，23次加法；最后可以得到在计算式(65)即$\det(\boldsymbol{C}_{40})$时需要的乘法和加法次数分别为：164次和95次。那么以同样的方法求4阶行列式$\det(\boldsymbol{C}_{41})$时需要的乘法和加法的次数分别为：40次和23次。

在有限域GF(2^{20})中，利用文献[15]提供的快速

求解逆元的方法可知，求一个元素的逆元时，需要的乘法次数为37次。所以在求解未知校验子S_3时，总共需要的乘法次数为：164+40+37+1=242次，加法次数为：95+23=118次。

为了求解错误位置多项式$L_4(z)$，还需先求解未知校验子$S_7=S_3^{16}$，计算S_7的乘法次数为15次。最后计算$L_4(z)$的系数σ_2时乘法进行了58次，加法进行了6次；计算σ_3时乘法进行了3次，加法进行了2次；计算σ_4时乘法进行了43次，加法进行了2次。

综上，该算法求解$L_4(z)$时需要的乘法总次数为：242+15+58+3+43=361次；需要进行的加法总次数为：118+0+6+2+2=128次。

4.2 本文提出算法$L_4(z)$的计算复杂度

本文提出的快速译码算法直接由已知校验子就可以求出错误位置多项式$L_4(z)$。且在v<4时已经计算出的已知校验子作为已知量，当v=4时，这些已知校验子就不再重复计算。

从式(31)—式(54)可以计算出σ_4的乘法和加法次数，其中有些变量多次出现，比如S_1^2出现了9次，$a_{211}a_{302}$出现了15次，$a_{202}a_{311}$出现了13次，等等。此类变量只需计算一次，在接下来的计算中就当作已知量使用，从而可以减少乘法的运算次数，进而减少译码时间。以上可知只需251次乘法和98次加法就可以计算出σ_4。由式(57)可知在计算σ_3时仅需1次乘法。由式(58)可以知道计算σ_2时有一个逆元的计算，所以需要3+37+1+1=42次乘法和3次加法。

综上，不用计算未知校验子的算法求解$L_4(z)$时需要的乘法总次数为：42+1+251=294次；需要进行的加法总次数为：3+0+98=101次。

4.3 计算$L_4(z)$的复杂度比较

由上可知，Lin算法和本文提出的算法在计算$L_4(z)$复杂度比较如表1。

表 1 两种译码算法计算$L_4(z)$复杂度的比较

算法	乘法	加法
Lin算法	361	128
本文算法	294	101
降低百分比(%)	18.56	21.09

由表1可知，在求解错误位置多项式$L_4(z)$时，本文提出的算法相对于Lin算法的运算量上，乘法量和加法量分别降低了18.56%和21.09%。

5 仿真结果

建立仿真的平台为64位的Windows 7操作系统，CPU为英特尔奔腾Pentium(R)Dual-Core，内存为4.00 GB，利用C++语言进行编程实现，采用BPSK调制，对(41, 21, 9)QR码的所有能够求解的错误模式集合进行穷举测试，对比了Lin算法和本文提出的代数译码算法仿真时在不同错误个数时的译码平均时间对比表如表2，所有的错误模式总共有$\sum_{i=1}^{4}C_{41}^{i}=112791$个。除此之外，令调制后的码字通过加性高斯白噪声(AWGN)信道，求解出(41, 21, 9)QR码的BER (Bit Error Rate)和FER (Frame Error Rate)，再利用MATLAB软件画出了Lin算法和本文提出算法的BER、FER性能曲线图如图1。

表 2 两种译码算法平均译码时间(μs)

错误个数	错误模式总数	Lin算法	本文算法
1	41	24.20	3.94
2	820	142.62	52.79
3	10660	275.60	231.73
4	101270	562.01	477.78

从表2中可以看出当接收码字中出现错误个数v=1的时候，Lin算法平均译码时间需要24.2 μs，而本文提出的算法只需要3.94 μs，本文算法相对于Lin算法，译码时间减少了83.72%；当接收码字中出现错误个数v=2, 3, 4的时候，本文算法相对于Lin算法，译码时间分别减少了62.99%, 15.92%, 14.98%，在v=4时，其时间的降低率与其计算复杂度的降低率相当。并且，从图1中可以看出两种算法的BER和FER的性能曲线完全一致，从而验证了本文提出算法的正确性和可靠性。

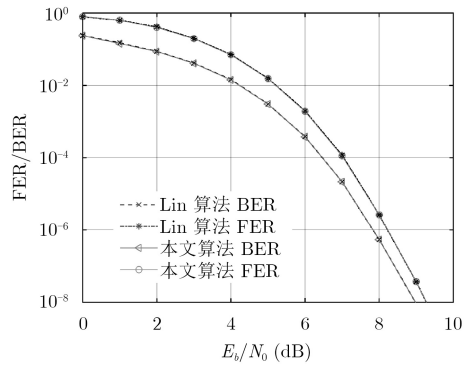

图 1 两种译码算法的BER与FER性能曲线图

6 结论

本文提出了(41, 21, 9) QR码在判定接收码字中分别出现了2, 3, 4个错的时候，不需要计算未知校验子直接求得对应的错误位置多项式系数的代数译码算法，并简化了文献[12]判定接收码字中发生

不同错误时的判断条件，从而降低了译码复杂度。对该码进行穷举仿真，验证了本文提出算法的可行性与正确性及维持原有译码性能不变，降低了译码时间。

参 考 文 献

[1] PRANGR E. Cyclic error-correcting codes in two symbols, AFCRC-TN-57-103[R]. Cambridge, MA: AirForce Cambridge Research Center, 1957.

[2] LEE Chongdao, CHANG Yaotsu, and CHANG Hohsuan. Unusual general error locator polynomial for the (23, 12, 7) golay code[J]. *IEEE Communications Letters*, 2010, 14(4): 339–341. doi: 10.1109/LCOMM.2010.04.091969.

[3] REED I S, TRUONG T K, CHEN Xuemin, *et al*. The algebraic decoding of the (41, 21, 9) quadratic residue code[J]. *IEEE Transactions on Information Theory*, 1992, 38(3): 974–986. doi: 10.1109/18.135639.

[4] CHEN Xuemin, REED I S, HELLESETH T, *et al*. Use of gröbner bases to decode binary cyclic codes up to the true minimum distance[J]. *IEEE Transactions on Communication*, 1994, 40(5): 1654–1661. doi: 10.1109/18.333885.

[5] CHIEN R. Cyclic decoding procedure for Bose-Chaudhuri-Hocquenghem codes[J]. *IEEE Transactions on Information Theory*, 1964, 10(4): 357–363. doi: 10.1109/TIT.1964.1053699.

[6] CHEN Yanhaw, HUANG Chingfu, and CHANG J. Decoding of binary quadratic residue codes with hash table[J]. *IET Communications*, 2016, 10(1): 122–130. doi: 10.1049/iet-com.2015.0546.

[7] LIN Tsungching, LEE Chongdao, CHEN Yanhaw, *et al*. Algebraic decoding of cyclic codes without error-locator polynomials[J]. *IEEE Transactions on Communications*, 2016, 64(7): 2719–2731. doi: 10.1109/TCOMM.2016.2569078.

[8] ZHANG Pengwei, LI Yong, CHANG Hsinchiu, *et al*. Fast decoding of the (47, 24, 11) quadratic residue code without determining the unknown syndromes[J]. *IEEE Communications Letters*, 2015, 19(8): 1279–1282. doi: 10.1109/LCOMM.2015.2440263.

[9] 陈高明, 黎勇, 董灿, 等. 一种(71, 36, 11)QR码的快速代数译码算法[J]. 重庆邮电大学学报(自然科学版), 2015, 27(6): 781–785. doi: 10.3979/j.issn.1673-825X.2015.06.013.
CHEN Gaoming, LI Yong, DONG Can, *et al*. A fast algebraic decoding algorithm of the (71, 36, 11) quadratic residue code[J]. *Journal of Chongqing University of Posts and Telecommunications* (*Natural Science Edition*), 2015, 27(6): 781–785. doi: 10.3979/j.issn.1673-825X.2015.06.013.

[10] LIN Tsungching, CHANG Hsinchiu, LI Yong, *et al*. Algebraic decoding of the (71, 36, 11) quadratic residue code[J]. *IET Communications*, 2016, 10(6): 734–738. doi: 10.1049/iet-com.2015.0159.

[11] HUANG Chingfu and CHEN Yanhaw. Efficient software method for decoding of the (71, 36, 11) quadratic residue code[C]. Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), Adelaide, Australia, 2015: 45–48.

[12] LIN Tsungching, TRUONG T K, LEE Hungpeng, *et al*. Algebraic decoding of the (41, 21, 9) quadratic residue code[J]. *Information Sciences*, 2009, 179(19): 3451–3459. doi: 10.1016/j.ins.2009.06.002.

[13] FENG G and TZENG K. A new procedure for decoding cyclic and BCH codes up to actual minimum distance[J]. *IEEE Transactions on Information Theory*, 1994, 40(5): 1364–1374. doi: 10.1109/18.333854.

[14] MACWILLIMS F J and SLOANE N J A. The Theory of Error Correcting Codes[M]. New York: North Holland, 1977: 244–245.

[15] WANG C C, TRUONG T K, SHAO H M, *et al*. VLSI architectures for computing multiplications and inverses in GF(2^m)[J]. *IEEE Transactions on Computers*, 1985, C-34(8): 709–717. doi: 10.1109/TC.1985.1676616.

吴　怡：女，1970 年生，教授、博士生导师，研究方向为无线自组织网、信道编码.

罗春兰：女，1994 年生，硕士生，研究方向为信道编码.

张新球：男，1954 年生，博士，研究方向为差错控制编码.

林　潇：男，1981 年生，助理研究员，研究方向为无线网络通信.

徐哲鑫：男，1985 年生，副教授，研究方向为无线自组织网络.