GENERATING GOPPA CODES

Thomas Risse

Institute for Informatics and Automation
Hochschule Bremen, University of Applied Sciences
Bremen, Germany
risse@hs-bremen.de

Abstract

Once quantum computers become operational all current public key crypto systems, PKCSs become obsolete. Fortunately, there are alternative quantum computer robust methods based on coding, on hashing, on multivariate polynomials, on lattices etc. One such candidate is the McEliece PKCS which is based on error correcting codes, e.g. Goppa codes. Hence, in order to implement the McEliece PKCS one needs to implement Goppa codes first. Here we show how to construct Goppa codes and especially how to construct binary irreducible Goppa codes which exhibit high error correction together with efficient encoding and decoding algorithms.

Keywords - postquantum cryptography, McEliece PKCS, Goppa codes, Goppa polynomial, parity check matrix, systematic generator matrix, irreducible polynomial, finite field

1 INTRODUCTION

Quantum computers are based on principles of quantum physics. They promise an improvement in performance previously unheard of. Once quantum computers become operational, all current, well established public key cryptographic systems, PKCSs become obsolete [2] because quantum computers can crack the notorious integer factorization problem or the discrete logarithm problem in minutes where traditional computers take hundred thousands of years [6], specifically factorization.

key length/bit	1024	2048	4096
classical computer	10^5 a	$5 \cdot 10^{15}$ a	$3 \cdot 10^{29}$ a
quantum computer	4.5 min	36 min	4.8 h

The key is a quantum fast Fourier transform [18]. Therefore, in order to be prepared, quantum computer robust alternatives have been investigated. These alternative methods are based on coding, on hashing, on multivariate polynomials, on lattices etc.
As a candidate the McEliece PKCS [9] is based on error correcting codes, most prominently on Goppa codes. Encoding is done by multiplying each block of the binary message stream with a big binary matrix which includes scrambling the data, then coding the scrambled data by a Goppa code, inserting errors to disguise and obscure the scrambled data and finally permute the coded scrambled data. This matrix serves as the public key. Decoding then includes the decoding of the modified message by e.g. the Patterson algorithm [12]. Moreover, [15] describes the McEliece PKCS in detail, [5] its security.

Here we want to specify – especially irreducible binary – Goppa codes. To do so we need to

- specify a Goppa code,
- determine a corresponding (systematic) generator matrix, and
- find an irreducible polynomial of given degree.

2 DEFINING GOPPA CODES

In the literature Goppa codes are defined in rather different ways (and the wikipedia article http://en.wikipedia.org/wiki/Goppa_code is classified as *'confusing or unclear and lacks appropriate citations'*)

- as linear codes whose code words satisfy some relation modulo some Goppa-Polynomial e.g. [2], [11], [17]
- as alternant generalized Reed-Solomon-Codes e.g. [16]
- as generalized algebraic geometry codes based on algebraic curves e.g. [13]
- using Fourier transform in finite fields e.g. [4]

Here, we will address and relate the first two most common definitions, only.

2.1 The Most Common Way

Most commonly we find the following

Definition Let $F = \mathbb{GF}_q$, $\Phi = \mathbb{GF}(q^m)$ and $L = \{\alpha_1, \dots, \alpha_n\} \subset \Phi$ be a set of pair wise different, so called *code locators* and let $g(x) \in \Phi[x]$ with $0 \notin g(L)$ be a Goppa-polynomial of degree t. Then

$C_{Goppa} = \{(c_1, \dots, c_n) \in F^n : \sum_{i=1}^{n} \frac{c_i}{x-\alpha_i} = 0 \, mod \, g(x)\}$ is a linear $[n, k, d]$ code over F. The code C_{Goppa} is called *irreducible* iff the Goppa polynomial g is irreducible.

Obviously, C_{Goppa} is a linear subspace of the vector space F^n over F, so it is a linear code. But, on the first glimpse we do not see the dimension k, the minimal distance d and how to generate code words from information words $u \in F^k$ because we lack a generator matrix G_{Goppa} of C_{Goppa} such that $c = u\, G_{Goppa} \in C_{Goppa}$. It would be even better to have a systematic generator matrix, i.e. G_{Goppa}, of the form $G_{Goppa} = (I|A)$ with $k \times k$ identity matrix I and $k \times (n-k)$ matrix A. It obviously saves memory space to store a systematic generator matrix, and it saves multiplications when computing the corresponding code word as the product $u\, G_{Goppa}$.

We observe that in $\Phi[x]/g(x)$ (which happens to be a field iff g is irreducible) we have

$$(x-\alpha)^{-1} = \frac{1}{x-\alpha} = -\frac{1}{g(\alpha)} \frac{g(x)-g(\alpha)}{x-\alpha}$$

because $(x-\alpha)\frac{-1}{g(\alpha)} \frac{g(x)-g(\alpha)}{x-\alpha} - 1 = -\frac{g(x)-g(\alpha)}{g(\alpha)} - 1 = -\frac{1}{g(\alpha)}\, g(x)$.

Let $g(x) = \sum_{i=o}^{t} g_i\, x^i$ be the Goppa polynomial. Then we have (best shown by induction in t, the degree of g)

$$\frac{g(x)-g(\alpha)}{x-\alpha} = g_t \sum_{i=o}^{t-1} \alpha^i x^{t-1-i} + g_{t-1} \sum_{i=o}^{t-2} \alpha^i x^{t-2-i} + \cdots + g_2(x+\alpha) + g_1$$

Then, $c \in C_{Goppa}$ iff $\sum_{i=1}^{n} \frac{c_i}{g(\alpha_i)} \frac{g(x)-g(\alpha)}{x-\alpha} = 0$ in $\Phi[x]$ and by comparison of coefficients $c \in C_{Goppa}$ iff $Hc^{\top} = 0$ with parity matrix

$$H = \begin{pmatrix} \frac{g_t}{g(\alpha_1)} & \frac{g_t}{g(\alpha_2)} & \cdots & \frac{g_t}{g(\alpha_n)} \\ \frac{g_{t-1}+\alpha_1 g_t}{g(\alpha_1)} & \frac{g_{t-1}+\alpha_2 g_t}{g(\alpha_2)} & \cdots & \frac{g_{t-1}+\alpha_n g_t}{g(\alpha_n)} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{g_1+\alpha_1 g_2+\cdots+\alpha_1^{t-1} g_t}{g(\alpha_1)} & \frac{g_1+\alpha_2 g_2+\cdots+\alpha_2^{t-1} g_t}{g(\alpha_2)} & \cdots & \frac{g_1+\alpha_n g_2+\cdots+\alpha_n^{t-1} g_t}{g(\alpha_n)} \end{pmatrix} = CXY$$

where

$$C = \begin{pmatrix} g_t & 0 & \cdots & 0 \\ g_{t-1} & g_t & & 0 \\ \vdots & \vdots & \ddots & \vdots \\ g_1 & g_2 & \cdots & g_t \end{pmatrix},\ X = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{t-1} & \alpha_2^{t-1} & \cdots & \alpha_n^{t-1} \end{pmatrix}$$ and

$$Y = \begin{pmatrix} \frac{1}{g(\alpha_1)} & & & \mathbf{0} \\ & \frac{1}{g(\alpha_2)} & & \\ & & \ddots & \\ \mathbf{0} & & & \frac{1}{g(\alpha_n)} \end{pmatrix}.$$

Here, the matrix C is lower triangular with non vanishing diagonal element(s) and hence invertible. Thus H and $\widehat{H} = XY$ both are parity matrices of the same linear code and, $\widehat{H}$ shows that this code is a generalized Reed-Solomon code C_{gRS} with (set of) code locators L and column multipliers $v_i = 1/g(\alpha_i)$. As a gRS code C_{gRS} is a $[n, k, d]$ code with dimension $k = n - t$ and minimal distance $d = t + 1$.

If we restrict code word elements to F we actually specify an alternant code $C_{Goppa} = C_{gRS} \cap F^n$. Its parity matrix H_{Goppa} is constructed from $\widehat{H}$ substituting each element of $\widehat{H}$ by the column vector in F^m according to some fixed basis of Φ over F. This procedure shows that as alternant code C_{Goppa} is a $[n, \geq n - (d-1)m, \geq d]$ code.

It remains to compute some generator matrix G_{Goppa} with $H_{Goppa}\, G_{Goppa}^{\top} = 0$ and if desired to turn it into a systematic generator matrix of the form $G_{Goppa} = (I|P)$ in order to reduce memory to store it and to save multiplications when encoding $u \in F^k$ to uG_{Goppa}.

2.2 The Alternant Way

Now, Goppa Codes are defined as special alternant generalized Reed-Solomon codes by the following

Definition Let $F = \mathbb{GF}_q$, $\Phi = \mathbb{GF}(q^m)$ and $L = \{\alpha_1, \ldots, \alpha_n\} \subset \Phi$ be the set of pair wise distinct code locators and let $g(x) \in \Phi[x]$ with $0 \notin g(L)$ a Goppa polynomial of degree t. Now, define a generalized Reed-Solomon-Code C_{gRS} by its (canonical) parity matrix with column multipliers $1/g(\alpha)$ for every $\alpha \in L$, i.e.

$$H_{gRS} = \left(\alpha_j^{i-1}\right) diag\left(\frac{1}{g(\alpha_1)}, \frac{1}{g(\alpha_2)}, \ldots, \frac{1}{g(\alpha_n)}\right) = \begin{pmatrix} 1/g(\alpha_1) & 1/g(\alpha_2) & \cdots & 1/g(\alpha_n) \\ \alpha_1/g(\alpha_1) & \alpha_2/g(\alpha_2) & \cdots & \alpha_n/g(\alpha_n) \\ \vdots & \vdots & & \vdots \\ \alpha_1^{t-1}/g(\alpha_1) & \alpha_2^{t-1}/g(\alpha_2) & \cdots & \alpha_n^{t-1}/g(\alpha_n) \end{pmatrix}$$

Then C_{gRS} is a linear $[n, K, D]$ code with $d = n - K + 1$. The corresponding alternant code $C_{Goppa} = C_{Goppa}(L, g) = C_{gRS} \cap F^n$ is a $[n, k, d]$ Goppa code with $k \geq n - mt$.

Selecting some base of Φ and representing each entry of H_{gRS} as a column vector in F^m results in the parity matrix H_{Goppa} of C_{Goppa} with (canonical) generator matrix $G_{Goppa} = ker(H_{Goppa})$.

3 PROPERTIES OF GOPPA CODES

Let C_{Goppa} be a $[n, k, d]$ code with Goppa polynomial g of degree $t < n = |L|$, the length of C_{Goppa}. By construction we have

- $n - mt \leq k \leq n - t$, the dimension k of C_{Goppa} [Hoffmann], p 24
- $t + 1 \leq d$, the *designed minimum distance* of C_{Goppa}
- A systematic generator matrix can be determined methodically [16].

Now, C_{Goppa} is called *binary* if $F = \mathbb{GF}_2$.

If C_{Goppa} is binary and g is irreducible or – more precisely – if g has no multiple roots in any extension field then $d \geq 2t + 1$, i.e. the code can correct up to t errors.

Binary Goppa codes with irreducible Goppa polynomial have not only very good error correcting capability but there are also efficient decoding algorithms [12], [15].

4 IRREDUCIBLE POLYNOMIALS

For our purposes it suffices to guarantee [3] that there are enough irreducible polynomials over Φ of given degree t.

Let $\mathcal{I}(t, q)$ be the number of irreducible monic polynomials over $\mathbb{GF}_q$ of degree t.

Then $\mathcal{I}(t, q) = \frac{1}{t} \sum_{s|t} \mu(s) q^{t/s}$ [7], [8], [11], [16] with the Möbius function $\mu: \mathbb{N} \to \{-1, 0, 1\}$ given by $\mu(n) = \mu\left(\prod_{j=1}^{r} p_j^{e_j}\right) = \begin{cases} 1 & if\ n = 1 \\ (-1)^r\ if\ e_j = 1\ for\ j = 1, \dots, r. \\ 0 & otherwise \end{cases}$

Also, the product of all irreducible monic polynomials in $\mathbb{GF}_q[x]$ whose degree divides t is $x^{q^t} - x$. (Alternatively, minimal polynomials also provide a way to come up with irreducible polynomials [8].)

Unfortunately, there is no known deterministic polynomial time algorithm for generating irreducible polynomials [1].

Irreducible polynomials are tabulated [8], [14] or can be computed at random. This is attractive because in our case, the factorization of $x^{(q^m)^t} - x$ is much too expensive.

Then, $1/\mathcal{I}(t, q^m)$ is the probability that a polynomial chosen at random is irreducible where this probability is positive because of $q^t - 2q^{t/2} \leq t\,\mathcal{I}(t, q^m) \leq q^t$ [3].

To test irreducibility we use e.g. the [7]

Rabin test The polynomial $f(x) \in \mathbb{GF}_q[x]$ of degree t is irreducible iff $f(x) | (x^{q^t} - x)$ and for all divisors $d | t$ one has $gcd\left(f(x), x^{q^d} - x\right) = 1$.

The proof is rather obvious:

$'' \Rightarrow ''$ Any irreducible f is a factor of $x^{q^t} - x$ and $gcd\left(f(x), x^{q^d} - x\right) \neq 1$ contradicts the irreducibility of f.

$'' \Leftarrow ''$ Let $f = \prod_{i=1}^{r} f_i$ be the product of all irreducible factors f_i of f. Then $d = deg(f_1) | t$ because $f_1(x) | (x^{q^t} - x)$, which leads to the contradiction $1 \neq f_1(x) | gcd\left(f(x), x^{q^d} - x\right)$.

5 CONCLUSION

Summarizing

- We have seen that Goppa Codes efficiently provide error correcting codes for the McEliece PKCS.
- The Patterson algorithm allows for efficient decoding [15].
- The generation consists of choosing an irreducible Goppa polynomial, generating a generator matrix for the corresponding Goppa code and possibly transforming it to a systematic generator matrix.
- We have illustrated how to generate Goppa codes and how to do so with minimal memory requirements.
- We have pointed out how to generate irreducible Goppa polynomials at random.

In this context especially the spectral analysis by the Fourier transform in finite fields [4] may promise further improvements.

References

[1] Leonard M. Adleman, Hendrik W. Lenstra Jr.: Finding Irreducible Polynomials over Finite Fields; Symposium on Theory of Computing, STOC 1986, pp 350—355

[2] Daniel J. Bernstein, Johannes Buchmann, Erik Dahmen (Eds.): Post-Quantum Cryptography; Springer 2009

[3] Markus Bläser, Chandan Saha: Computational Number Theory and Algebra; www.mpi-inf.mpg.de/~csaha/lectures/lec9.pdf

[4] Richard E. Blahut: Algebraic Codes on Lines, Planes and Curves; Cambridge University Press 2008

[5] H. Dinh, C. Moore, A. Russell: The McEliece Cryptosystem Resists Quantum Fourier Sampling Attacks; 2010, http://arxiv.org/pdf/1008.2390v2.pdf

[6] Robert Feldmann: Quantenalgorithmen; Universität Leipzig, 2002 www.physik.uni-leipzig.de/#feldmann/Documents/Quantenalgorithmen.pdf

[7] Tanja Lange: Finite Fields; Coding Theory and Cryptology I, Fall 2011 Eindhoven University of Technology, http://hyperelliptic.org/tanja/teaching/CCI11/online-ff.pdf

[8] Rudolf Lidl, Harald Niederreiter: Finite Fields; Encyclopedia of Mathematics and its Applications, Cambridge University Press 1997

[9] Robert McEliece: Public Key Cryptosystem based on Algebraic Coding; DSN Progress Report 42-44, January/February 1978, 114 – 116

[10] Alfred J. Menezes, Paul C. van Oorschot, Scott A.\ Vanstone: Handbook of Applied Cryptography; CRC Press, 1996 http://cacr.uwaterloo.ca/hac

[11] Todd K. Moon: Error Correction Coding; Wiley 2005

[12] N. J. Patterson: The Algebraic Decoding of Goppa Codes; IEEE Transactions on Information Theory, Vol. IT-21, No 2, March 1975, 203 – 207

[13] Antonino Pecorella, Alberto Picone: On GAG-Codes and Geometric Goppa Codes; http://documents.ct.infn.it/record/178/files/Manoscritto.pdf

[14] W. Wesley Peterson, E. J. Weldon: Error Correcting Codes; MIT-Press 1972

[15] Thomas Risse: How SAGE helps to implement Goppa Codes and the McEliece Public Key Crypto System; Ubiquitous Computing and Communication Journal, UbiCC, ISSN 1992-8424, Special Issue on 5th International Conference on Information Technology, ICIT'11, Amman 2011

[16] Ron M. Roth: Introduction to Coding Theory; Cambridge University Press 2006 www.cs.technion.ac.il/~ronny

[17] Nicolas Sendrier: Binary Goppa Codes; Indocrypt 2009 tutorial http://indocrypt09.inria.fr/goppa.pdf

[18] Peter W. Shor: Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer; Proceedings of the 35th Annual Symposium on Foundations of Computer Science, Santa Fe, NM, Nov. 1994, 20 – 22