ICIT 2015 The 7th International Conference on Information Technology
doi:10.15849/icit.2015.0047

Groebner Bases and Coding

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

***Abstract*— In the past Groebner bases have been proved to be a very potent tool to solve a variety of problems first of all in mathematics but also in science and engineering. Hence, it is near at hand to study application of Groebner bases in coding, i.e. the encoding and especially the decoding of linear error correcting codes. This paper attempts an overview focusing on Reed-Solomon codes and Goppa codes together with their coding and decoding algorithms.**

Keywords*— *Groebner bases, error correcting codes, decoding, Reed-Solomon codes, Goppa codes

I. INTRODUCTION

A Groebner basis (according to Bruno Buchberger, 1965) or a standard basis (according to Heisuke Hironaka, 1964) is a finite generating set of an ideal I in the polynomial ring $R = K[x_1, \dots, x_n]$ over a field K. For any such ideal the (reduced) Groebner basis is unique and can be determined algorithmically. This basis allows solving some prominent mathematical problems, e.g. to decide whether some polynomial belongs to I or not, whether two ideals are identical, whether two varieties are identical or not etc. In his seminal thesis [5] Buchberger developed the theory and presented the necessary algorithms. He also investigated applications [7] of Groebner bases like solving systems of multivariate polynomial equations.

In the eighties the rapid development of computers spurred further investigation of Groebner bases which resulted in improvements of the algorithms and even more applications [7]. Especially Computer Algebra Systems benefitted [9] from Groebner bases. But Groebner bases also brought forth progress in coding and cryptography.

II. GROEBNER BASES

A. Definitions [11]

Let I be some ideal in $R = K[x_1, \dots, x_n]$. Then by Hilberts basis theorem, I is finitely generated, i.e. $I = \langle f_1, \dots f_s \rangle$. Now fix some monomial order on the monomials in R to be able to specify leading monomials $LM(f)$, leading terms $LT(f)$ and leading coefficients $LC(f)$ for any f in R. Then a Groebner basis G for I is a set $G = \{g_1, \dots, g_t\}$ with $I = \langle G \rangle$ so that the ideal generated by the leading terms of the elements in I is generated by the leading terms $LT(g)$ for g in G, i.e. $\langle LT(g_1), \dots, LT(g_t) \rangle = \langle LT(I) \rangle$. Equivalently, $G = \{g_1, \dots, g_t\}$ is a Groebner basis if and only if $LT(f)$ is divisible by one of the $LT(g_i)$ for any f in I. By the way, G then has the minimality properties of a proper ideal basis. Furthermore, G is unique, and any f in R can be written as $f = g + r$ for unique polynomials g and r with g in I and no term of r is divisible by any element of $LT(g_i)$.

B. Algorithms [11]

Buchberger's algorithm computes a (not reduced) Groebner basis for an ideal $I = \langle f_1, \dots f_s \rangle$.using sysygy- or S-polynomials $S(f,g) = \frac{LCM(LM(f),LM(g))}{LT(f)} f - \frac{LCM(LM(f),LM(g))}{LT(g)} g$ for any two polynomials f and g in R together with a generalization of the polynomial division algorithm for polynomials in one variable to the case of multivariate polynomials $f, f_1, \dots, f_s, r$ in R such that $f = a_1 f_1 + \cdots + a_s f_s + r$ where the remainder $r = \bar{f}^{\{f_1,\dots,f_s\}}$ is zero or a K-linear combination of monomials none of which is divisible by any $LT(f_1), \dots, LT(f_s)$ – all in the usual notation of [11][21] et al. With these definitions Buchberger's algorithm can now be specified.

```
input: F = (f_1, ..., f_s) ⊂ K[x_1, ..., x_n]
output: G = (g_1, ..., g_t) with ⟨F⟩ = ⟨G⟩
repeat
        G' := G
        for each {p, q} ⊂ G', p ≠ q do
                S := S(p, q)^G
                if S ≠ 0 then G := G ∪ {S}
until G = G'
```

Code snippet 1. Computation of G with $\langle G \rangle = \langle F \rangle$

Obviously, this very simple version of Buchberger's algorithm extends the given set F to G. A reduction step removes superfluous elements from G resulting in the unique reduced Groebner basis of I. There are improved versions [11] to compute the unique, reduced Groebner basis of I efficiently.

doi:10.15849/icit.2015.0047

C. Applications [11]

First, one should note that the concept of Groebner bases generalizes both Euclid's algorithm to compute the greatest common divisor, gcd, of two polynomials as well as Gauß's algorithm to solve a system of linear equations.

Euclids algorithm

```
in: f, g ∈ K[x]; out: h = gcd(f, g)
h := f; s := g;
while s ≠ 0
        r = remainder(h, s);
        h := s; s := r;
```

Code snippet 2. $h = gcd(f, g)$ for any $f, g \in K[x]$

Regard each equation of a system of linear equations in the unknowns $x_1, \dots, x_n$ as a linear polynomial f_i in $K[x_1, \dots, x_n]$. Then, the reduced Groebner basis $G = \{g_1, \dots, g_t\}$ of $I = \langle f_1, \dots, f_n \rangle$ consists of linear, non-zero polynomials whose coefficients correspond to the non-zero rows in the reduced echelon form of the coefficient matrix of the system of linear equations. In this sense, computation of the reduced Groebner basis is equivalent to Gauß's algorithm.

As one of the very many applications of Groebner bases consider the problem to solve a system of multivariate polynomial equations $f_1 = f_2 = \cdots = f_s = 0$ for f_i in R. Here we use $I = \langle f_1, \dots f_s \rangle = \langle G \rangle$ for the reduced Groebner basis $G = \{g_1, \dots, g_t\}$ of I. It turns out that the set of equations $g_1 = g_2 = \cdots = g_t = 0$ is easier to solve because using the lexicographic order (*lex*), variables are eliminated in that order in the Groebner basis so that a process like back substitution generates the variety $V(I) = V(\langle G \rangle) = V(g_1, \dots, g_t)$.
Elimination theory [11] provides the proofs and [20] more examples.

III. Error Correcting codes

Here we consider linear block codes [17] only. An alphabet is some finite field $\mathbb{F} = \mathbb{GF}(p^m)$ for prime p and $m \in \mathbb{N}$ and information words u of length k are in $\mathbb{F}^k$. Code words over $\mathbb{F}$ are of the form uG for a $n \times k$ *generator matrix* G. Hence the code $\mathcal{C} = \{uG : u \in \mathbb{F}^k\}$ is a linear subspace of $\mathbb{F}^n$. $\mathcal{C}$ can also be characterized as the kernel space $\mathcal{C} = \{c \in \mathbb{F}^n : Hc^\top = 0\}$ of the *parity matrix* H, i.e. $HG^\top = 0$. If any two code words have a Hamming distance of at least d then at most $(d-1)/2$ errors in a transmitted code word can be corrected. Such a code is called a (linear) $[n, k, d]$ code.

Encoding an information word u to $c = uG \in \mathcal{C}$ is easy whereas decoding a corrupted word $y = c + e$ with an error vector $e \in \mathbb{F}^n$ with no more than $(d-1)/2$ non-zero elements to the original c (and then to the original information word u) is difficult. In fact, it is NP-complete [2]. However, for many specific (linear) codes there exist efficient decoding algorithms.

A. (Generalized) Reed-Solomon Codes

(Generalized) Reed-Solomon codes, RS and gRS, are an important class of codes comprising many other important codes. Such code $\mathcal{C}_{gRS}$ is specified by its n distinct non-zero *code locators* $\alpha_1, \dots, \alpha_n \in \mathbb{F}$ and n *column multipliers* $v_1, \dots, v_n \in \mathbb{F}$. Then the parity matrix H_{gRS} of $\mathcal{C}_{gRS}$ is defined by

$$H_{gRS} = \begin{pmatrix} \alpha_1^0 & \alpha_2^0 & \cdots & \alpha_n^0 \\ \alpha_1^1 & \alpha_2^1 & \cdots & \alpha_n^1 \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{n-k-1} & \alpha_2^{n-k-1} & \cdots & \alpha_n^{n-k-1} \end{pmatrix} \begin{pmatrix} v_1 & & & 0 \\ & v_2 & & \\ & & \ddots & \\ 0 & & & v_n \end{pmatrix}.$$

Then, $\mathcal{C}_{gRS}$ is a (linear) $[n, k, d]$ code with $d = n - k + 1$. (Such codes attain the Singleton bound $d \leq n - k + 1$ and are called maximum distance separable, MDS codes.) For gRS codes there are efficient decoding algorithms: e.g. solving linear equations [17], using Euclid's algorithm [22] or linear recurrences in case of the famous Berlekamp-Massey algorithm [3][17][18]. List decoding of e.g. (generalized) Reed-Solomon codes relaxes the assumption on the number of allowed errors and returns a list of possible code words.

B. Goppa-Codes

Goppa-codes, alternant gRS codes, play an important role e.g. in the McEliece Public Key Crypto System, PKCS [18][19]. Let $F = \mathbb{GF}_q$, $\mathrm{K} = \mathbb{GF}(q^m)$ and $L = \{\alpha_1, \dots, \alpha_n\} \subset \mathrm{K}$ be a set of pair wise different *code locators* and let $g(x) \in \mathrm{K}[x]$ with $0 \notin g(L)$ be a *Goppa*-polynomial of degree t. Then

$$C_{Goppa} = \{(c_1, \dots, c_n) \in F^n : \sum_{i=1}^n \frac{c_i}{x - \alpha_i} = 0 \bmod g(x)\}$$

is a linear $[n, k, d]$ code over F. The code C_{Goppa} is called *irreducible* iff the Goppa polynomial g is irreducible. 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 $\mathrm{K}[x]$ and by comparison of coefficients $c \in C_{Goppa}$ iff $Hc^\top = 0$ with parity matrix $H =$

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

where

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

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

Such codes correct up to $\frac{t}{2}$ errors, even up to t errors in the binary case, i.e. if C_{Goppa} is a code over $\mathbb{F} = \mathbb{GF}(2)$.

doi:10.15849/icit.2015.0047

Early methods used Euclid's algorithm for decoding or list decoding [23]. Later Patterson's algorithm [16] provided an efficient method to decode received words when using a Goppa encoding [18]. On top one can correct approximately up to $\frac{t^2}{n}$ errors [4].

C. Cyclic Codes

Cyclic codes [17] are linear codes C when in addition with any code word $(c_o, \dots, c_{n-1}) \in C$ also $(c_{n-1}, c_o, \dots, c_{n-2}) \in C$, i.e. the shifted word is again a code word. Conventional Reed-Solomon codes (code locators $\alpha_j = \alpha^{j-1}$ are powers of an element $\alpha \in \mathbb{F}$ of multiplicative order n) as well as BCH codes (alternant codes of conventional Reed-Solomon codes) are prominent examples of cyclic codes. Cyclic codes feature efficient encoding (multiplication by the generator polynomial g of the code), syndrome computation (remainder of division by g) and decoding (sequentially by Meggitt decoder) via rather simple hardware.

IV. Applying Groebner Algorithms to Coding

There are several ways [10] to transform the decoding problem into a problem of solving a system of multivariate polynomial equations. A straightforward way is to consider the (unknown) entries e_i of the error vector e as variables E_i. If H consists of rows $h_1, \dots, h_r$ with redundancy $r = n - k$ then the vector equation $s = He^\top$ is equivalent to the r linear equations

$$\sum_{j=1}^{n} (h_i)_j \, E_j - s_i = 0 \text{ for } i = 1, \dots, r \qquad (1)$$

We can formulate the condition that e has at most $t = \lfloor \frac{d-1}{2} \rfloor$ non-zero entries by the $\binom{n}{t+1}$ equations of multidegree $t + 1$

$$E_{j_1} \cdot E_{j_2} \cdot \dots \cdot E_{j_{t+1}} = 0 \text{ for } 1 \le j_1 < j_2 < \dots < j_{t+1} \le n \qquad (2)$$

Let the two sets of equations together generate the ideal I. Then the Groebner basis of I allows to read off the solution E, $E = (E_1, \dots, E_n)$ i.e. the one element in the variety $V(I)$.

In addition, [10][21] present alternatives to (2) with less equations of lower multidegree so that the Groebner basis is faster to compute.

A. RS and gRS codes

Decoding RS and gRS codes means to solve the key equations. Hence in general a formulation of the decoding problem using Groebner bases is near at hand. But exploiting the fact that Groebner bases help to determine the corresponding variety $V(I)$ of some ideal $I = \langle G \rangle$ for the reduced Groebner basis G of I explains why Groebner bases support list decoding naturally. [15] gives an overview over existing methods.

B. Goppa codes

[14] is most promising to decode Goppa codes. However, [14] shows '*that one can, at least in theory, decode these codes up to half the true minimum distance by using the theory of Groebner bases*'. Therefore, what is lacking is the transfer of the solution of [14] into praxis.

C. Cyclic codes

[12] gives an algorithm to decode cyclic codes using Groebner bases. The decoding problem is represented as a system of $n - k$ linear equations together with n quadratic equations in at most $n + d$ unknowns, i.e. error locations and error values. Because the number of errors is not known beforehand, the algorithm then starts with assumed $t = 0$ errors and increases t as long as the variety $V(I) = \emptyset$ where I is the ideal generated by equations specified above. Once $V(I) \neq \emptyset$ it contains the unique solution. However, the viability of the algorithm is limited because on one hand there are aforesaid efficient decoding methods and on the other hand the cost to compute a Groebner basis might be prohibitive.

Conclusion

This article is meant to set the stage for Groebner bases in coding. In the light of the very many application of Groebner bases in science and engineering [7] it is to be expected that further research will reveal even better algorithms for the decoding of linear (and non-linear) error-correcting codes. (Also, Groebner bases have spurred the specification and investigation of new linear codes [13][14].) The exact average complexity of determining the reduced Groebner basis of an ideal is not known right now. Once it has been determined [10] one will be able to set objectives and to identify limits of the approach to apply Groebner bases for coding.

References

[1] E.R. Berlekamp, Goppa Codes, IEEE Transactions Information Theory, Vol 19, No 5, September 1973, 590–592 http://infosec.seu.edu.cn/space/kangwei/senior_thesis/Goppa.pdf

[2] .R. Berlekamp, R.J. McEliece, H.C.A. van Tilborg, On the inherent intractability of certain coding problems, IEEE Trans. Inform. Theory, 24 (1978) 384–386

[3] E.R. Berlekamp, Algebraic Coding Theory, Aegean Park Press 1984

[4] D.J. Bernstein, List decoding for binary Goppa codes, 2008 http://cr.yp.to/ntheory/goppalist-20080706.pdf

[5] B. Buchberger, Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal, Universität Innsbruck, Dissertation, 1965 http://www.ricam.oeaw.ac.at/Groebner-Bases-Bibliography/gbbib_files/publication_706.pdf

[6] B. Buchberger, An Algorithmic Criterion for the Solvability of a System of Algebraic Equations, Aequationes Mathematicae 4 (1970), 374–383 http://www.ricam.oeaw.ac.at/Groebner-Bases-Bibliography/gbbib_files/publication_699.pdf

[7] B. Buchberger, Groebner Bases – A Short Introduction for System Theorists, Proceedings of Computer Aided Systems Theory, EURO-CAST, 1–19, 2001 http://people.reed.edu/~davidp/pcmi/buchberger.pdf

[8] B. Buchberger, H. Engl, Workshop D1: Groebner Bases in Cryptography, Coding Theory, and Algebraic Combinatorics, May 1st – May 6th, 2006 with papers on coding, cryptography, algebraic combinatorics, etc. http://www.ricam.oeaw.ac.at/specsem/srs/groeb/schedule_D1.html

[9] B. Buchberger, A. Maletzky (organizers), session 'Software for Groebner Bases', 4th Int. Congress on Mathematical Software, ICMS, Seoul, August 5th – 9th, 2014
http://www.risc.jku.at/people/amaletzk/ICMS2014-GB.html

[10] S. Bulygin, R. Pelikaan, Bounded distance decoding of linear error-correcting codes with Groebner bases, J. Symbolic Computation 44 (2009) 1626–1643
http://www.sciencedirect.com/science/article/pii/S0747717108001776

[11] D. Cox, J. Little, D. O'Shea, Ideals, Varieties, and Algorithms, Springer 2007 http://www.math.ku.dk/~holm/download/ideals-varieties-and-algorithms.pdf

[12] M. de Boer, R. Pellikaan, Gröbner bases for error-correcting codes and their decoding. http://www.risc.jku.at/Groebner-Bases-Bibliography/gbbib_files/publication_590.pdf

[13] C. Di, Z. Liu, Construction of a Class of Algebraic-Geometric Codes via Groebner Bases, MM Research Preprints, 42–48 No. 16, April 1998. Beijing http://www.mmrc.iss.ac.cn/pub/mm16.pdf/cdi.pdf

[14] J. Fitzgerald , R.F. Lax Decoding affine variety codes using Groebner bases, Designs, Codes and Cryptography, 13, 147–158 (1998)
https://www.math.lsu.edu/~lax/designscodescrypt.pdf or
http://www.ricam.oeaw.ac.at/Groebner-Bases-Bibliography/gbbib_files/publication_277.pdf

[15] H. O'Keeffe, P. Fitzpatrick, A Groebner basis approach to list decoding of Reed-Solomon and Algebraic Geometry Codes, see [8]
https://www.ricam.oeaw.ac.at/specsem/srs/groeb/download/OKeeffe.pdf

[16] N. J. Patterson: The Algebraic Decoding of Goppa Codes; IEEE Trans. on Information Theory, Vol IT-21, No 2, March 1975, 203–20

[17] R.M. Roth, Introduction to Coding Theory, Cambridge 2006

[18] Th. Risse, How SAGE helps to implement Goppa Codes and McEliece PKCSs, Int. Conf. on Information Technologies 2011, ICIT'11, May 11th – 13th, 2011, Amman
http://www.weblearn.hs-bremen.de/risse/papers/ICIT11/

[19] Th. Risse, Generating Goppa Codes, Int. Conf. on Information Technologies 2013, ICIT'13, May 8th – 10th, 2013, Amman
http://sce.zuj.edu.jo/icit13/index.php/accepted-papers/2-uncategorised/41-applied-mathematics

[20] Th. Risse, Groebner-Basen, 12. Workshop Mathematik für Ingenieure, Hafen City Universität, Hamburg 12. – 13.2.2015,
http://www.weblearn.hs-bremen.de/risse/papers/MathEng12/

[21] M. Sala, T. Mora, L. Perret, S. Sakata, C. Traverso (Editors), Groebner Bases, Coding, and Cryptography, Springer 2009
http://xa.yimg.com/kq/groups/24626876/489439549/name/ggp496.pdf

[22] P. Shankar, Decoding Reed–Solomon Codes Using Euclid's Algorithm, Resonance April 2007, 37–51
http://www.ias.ac.in/resonance/Volumes/12/04/0037-0051.pdf

[23] Y. Sugiyama, M. Kasahara, S. Hirasawa, T. Namekawa, A Method for Solving Key Equation for Decoding Goppa Codes, Information and Control 27 (1975) 87–99
http://www.sciencedirect.com/science/article/pii/S001999587590090X