Journées « Codage et Cryptographie »

Workshop on Coding and Cryptography

10 au 14 janvier 1999
January 10 to 14, 1999

Cercle National des Armées, Place Saint Augustin, Paris, France

—

Organisé par – *Organized by*

INRIA
Centre de Recherche des Écoles de Coëtquidan

Avec le parrainage de – *Sponsored by*

DGA, Délégation Générale pour l'Armement
SCSSI, Service Central de la Sécurité des Systèmes d'Information
Thomson-CSF Communication

—

Comité d'organisation – *Organizing committee*

Claude CARLET	Université de Caen, France
Gérard COHEN	Président du chapitre français de IEEE-IT, France
Caroline FONTAINE	INRIA Rocquencourt et Université Paris-Sud, France
Éric FILIOL	Centre de Recherche des Écoles de Coëtquidan, France
Sami HARARI	Université de Toulon et du Var, France
Nicolas SENDRIER	**Président**, INRIA Rocquencourt, France

Organisation locale – *Local organization*

Annick THEIS-VIEMONT	INRIA Rocquencourt, France
Claudie THÉNAULT	INRIA Rocquencourt, France

Comité de programme – *Program committee*

Daniel AUGOT	INRIA Rocquencourt, France
Joseph BOUTROS	ENST Paris, France
Thierry BERGER	Université de Limoges, France
Alister BURR	University of York, UK
Anne CANTEAUT	INRIA Rocquencourt, France
Claude CARLET	**Président**, Université de Caen, France
Pascale CHARPIN	INRIA Rocquencourt, France
Gérard COHEN	ENST-Paris, France
Jean-Marc COUVEIGNES	Université de Toulouse et DGA, France
Patrick FARRELL	University of Manchester, UK
Alain GLAVIEUX	ENST Bretagne, France
Marc GIRAULT	CNET Caen, France
Sami HARARI	Université de Toulon et du Var, France
Gregory KABATIANSKY	Russian Academy of Sciences, Russia
Dominique LEBRIGAND	Université Paris 6, France
Simon LITSYN	University of Tel Aviv, Israel
Harold MATTSON	University of Syracuse, USA
François MORAIN	DGA et École Polytechnique, France
Nicolas SENDRIER	INRIA Rocquencourt, France
Simon SHEPHERD	University of Bradford, UK
Henk VAN TILBORG	Eindhoven University of Technology, NL

Avant-propos

Soyez les bienvenus à ces Journées « Codage et Cryptographie ». Cet avant-propos me donne l'occasion de remercier tous les membres du comité de programme. Nous avons reçu soixante-seize soumissions. Chacune d'entre elles a été référée par deux rapporteurs. Quarante neuf ont été retenues. Je désire également remercier Christine Bachoc, Iwan Duursma, Eric Filiol, Philippe Gaborit, David Haccoun, Nicolas Lagorce, Pierre Loidreau, Lancelot Pecquet, Antoine Valembois, Patrick Solé et Ulrich Sorger pour leur aide précieuse dans le processus d'évaluation des soumissions. Moult mercis également à Nicolas Sendrier et Claudie Thénault, ainsi qu'à Eric Filiol, grâce à qui nous avons pu organiser ces journées au « Cercle Militaire ». Un grand merci, enfin, à Daniel Augot qui a eu la charge de ce livre de résumés.

Je vous souhaite un bon séjour à Paris !

Claude Carlet
Président du comité de programme

Foreword

I am very pleased to welcome you to this first Workshop on Coding and Cryptography in Paris. This foreword gives me the opportunity to thank all the members of the program committee. We considered seventy-six submissions. Each of them was reviewed by two referees. Forty-nine have been accepted. I wish also to thank Christine Bachoc, Iwan Duursma, Eric Filiol, Philippe Gaborit, David Haccoun, Nicolas Lagorce, Pierre Loidreau, Lancelot Pecquet, Antoine Valembois, Patrick Solé and Ulrich Sorger for their precious help in refereeing. Many thanks also to the organizers, and especially to Nicolas Sendrier and Claudie Thénault for their great job, and to Eric Filiol thanks to whom we managed to organize this workshop at the "Cercle Militaire". And, last but not least, Daniel Augot who was in charge of this book of abstracts.

I wish you a pleasant stay in Paris !

Claude Carlet
Program chairman

Sommaire – *Contents*

Conférences invitées – *Invited talks*

Contributions – *Contributed papers*

Code structure

Block codes, spherical codes and hash functions

Codes over Z4

Codes over rings

Lattices and designs

Convolutional codes

On Self-Dual and Formally Self-Dual Codes

Vera Pless
(co-authors-W.C. Huffman, J.Fields, P.Gaborit)

Self-dual codes are an interesting class of codes. The extended Golay codes over GF(2) and GF(3) are self-dual as is the [8,4,4] binary Hamming code and the [6,3,4] Hexacode over GF(4). The weight enumerators of these codes are given by the Gleason polynmials which also provide an upper bound on their minimum weights. Codes which meet this bound are called extremal and vectors of a fixed weight in them contain t-designs where t can be 5,3 or 1 (depending on the particular case).

In order to find the extremal codes, all self-dual codes of a particular length have been classified for many modest lengths. This has been possible because there are formulas for the number of self-dual codes of a fixed length.

Binary self-dual codes come in two types, Type II-the doubly-even codes where all weights are divisible by 4 and Type 1 where some weights are also equivalent to 2 (mod 4). The largest length where all Type II codes have been classified is 32. It is impractical to do this at longer lengths due to the large number of such codes. Attempts have been made (so far not successfully) to classsify only the extremal codes at longer lengths even though there are no formulas for the number of such codes.

If a binary code has the same weight distribution as its dual code, it is called formally self-dual (f.s.d.). These codes include the self-dual codes and when they are not self-dual, their weight enumerators are combinations of Gleason polynomials as are weight enumerators of Type 1 codes. Extremal f.s.d. codes exist which have higher minimum weights than self-dual codes with the same parameters. We describe recent work (as yet unpublished) which has classified the extremal f.s.d codes through length 30.

New permutation polynomials and applications to codes, sequences and Boolean functions

Hans Dobbertin

We present a new systematic technique to prove that certain polynomials are permutation polynomials (PP's) on finite fields of characteristic two. Using this method we find new PP's and new proofs for know PP's. In combination with results of the work at INRIA (Pascale Charpin, Anne Canteaut), some of these new PP's form important ingredients for a recently achieved progress on determining

- cross-correlation functions of binary m-sequences, or to put into other terms,
- the weight distribution of certain codes, resp. the non-linearity of certain power functions.

In our talk we shall describe the state of the art and discuss lines for future reasearch.

Metrics generated by linear codes in cryptography

Ernst M. Gabidulin

Each linear code can be treated as a subspace of a metric space. We refer to this metric as the original metric (very common case is the Hamming metric). On the other hand, a linear code generates a metric defined on the set of its syndroms. We refer to this metric as the generated metric. One can construct a code with syndroms as code words correcting errors in the generated metric. The general construction of a public-key cryptosystem is as follows. For a given linear code C_1, a legal user finds an (easy decodable) code C_2 in the generated metric and uses this code to modify a parity check metric of the original code. This modified matrix is published as the public key. Secret keys are fast decoding methods for the original metric and for the generated metric. Let the set of plaintexts be all the correctable errors in the original metric.

Encryption: for a given plaintext calculate a syndrome using the modified parity check matrix. This is a ciphertext.

Decryption: a ciphertext is a sum of a codeword of C_2 and a syndrome of a correctable error for C_1. The legal user decodes it using the generated metric and finds a syndrome of a correctable error for C_1. Then he applies the decoding algorithm for the original code C_1 and finds a correctable error, i.e., a plaintex t. We discuss general properties of this scheme and give examples.

Invited talk on cryptography

James Massey

Construction and classification of quasicyclic codes

Kristine Lally and Patrick Fitzpatrick
Department of Mathematics
National University of Ireland, Cork
Cork, Ireland
email: k.lally@ucc.ie, fitzpat@ucc.ie

Abstract

We use the theory of Gröbner bases to develop a new technique for constructing and classifying quasicyclic codes. A number of interesting results follow from this standard representation. For example, it is straightforward to determine the dimension of a quasicyclic code from its reduced Gröbner basis relative to a certain order.

1 Introduction

The theory of Gröbner bases of modules (developed in [1, 2]) has been applied [3, 4, 5, 6, 7] to decoding Reed-Solomon codes, to scalar rational interpolation, and to various other problems, such as Padé approximation, that can be represented as solving systems of polynomial congruences. In [9] the authors use the theory to develop machinery for analysis of Hermitian codes. The essential idea is to use a cyclic group of automorphisms of the code to represent it as a module over the polynomial ring $F[x]$ in one variable. In this paper we adopt the same approach to provide a new method of construction and classification of quasicyclic codes.

We assume the reader is familiar with the elementary theory of Gröbner bases. Let $A = F[x]$ for some finite field F, and let A^l be a free A–module. The standard basis elements are

$$\mathbf{e}_1 = (1, 0, \ldots, 0), \mathbf{e}_2 = (0, 1, 0, \ldots, 0), \ldots, \mathbf{e}_l = (0, 0, \ldots, 0, 1) \in A^l.$$

We define a term in A^l to be a vector of the form $\mathbf{X} = x^i \mathbf{e}_j$, $1 \leq j \leq l$. We fix a term order $<$ in A^l defined by $x^i \mathbf{e}_j < x^{i'} \mathbf{e}_{j'}$ if and only if $j > j'$ or $j = j'$ and $i < i'$. Note that $\mathbf{e}_1 > \mathbf{e}_2 > \cdots > \mathbf{e}_l$. (This is the "position over term" or POT ordering.)

Once a term order has been chosen, any $\mathbf{f} \in A^l$ can be expressed as

$$\mathbf{f} = a_1 \mathbf{X}_1 + a_2 \mathbf{X}_2 + \ldots + a_r \mathbf{X}_r$$

where $a_i \in F$, $\mathbf{X}_i$ is a term and $\mathbf{X}_1 > \mathbf{X}_2 > \ldots > \mathbf{X}_r$. The leading term of $\mathbf{f}$, denoted $\mathrm{lt}(\mathbf{f})$, is $\mathbf{X}_1$, and the leading coefficient of $\mathbf{f}$, denoted $\mathrm{lc}(\mathbf{f})$, is a_1. By definition, $\mathrm{lt}(\mathbf{0}) = \mathrm{lc}(\mathbf{0}) = \mathbf{0}$. The term $x^i\mathbf{e}_j$ divides $x^{i'}\mathbf{e}_{j'}$ provided $j = j'$ and $i \leq i'$. If $M \subseteq A^l$ is a submodule then a set of non-zero elements $\mathcal{G} = \{\mathbf{g}_1, \ldots, \mathbf{g}_t\}$ in M (with respect to the fixed term order) is a Gröbner basis of M if and only if for all $\mathbf{f} \in M$ there exists $i \in \{1, \ldots, t\}$ such that $\mathrm{lt}(\mathbf{g}_i)$ divides $\mathrm{lt}(\mathbf{f})$. A Gröbner basis $\mathcal{G} = \{\mathbf{g}_1, \ldots, \mathbf{g}_t\}$ is reduced if all $\mathbf{g}_i$ are monic and reduced with respect to $\mathcal{G}$, that is, no term in $\mathbf{g}_i$ is divisible by $\mathrm{lt}(\mathbf{g}_j)$ for any $j \neq i$. Such a reduced Gröbner basis for M is unique. If $\mathcal{G}$ is a Gröbner basis of M then each element $f \in A^l$ has a uniquely defined normal form relative to $\mathcal{G}$ denoted by $N_{\mathcal{G}}(f)$, and the set of cosets of M represented by terms which are in normal form is a vector space basis of the quotient A^l/M.

2 Quasicyclic codes

Let $I = \langle x^m - 1\rangle \subseteq A$ and denote by K the submodule of A^l generated by $\{(x^m - 1)\mathbf{e}_j \mid j = 1, \ldots, l\}$. If M is a submodule of A^l containing K then the image C of M under the natural homomorphism modulo K is a quasicyclic code of index l and length $n = ml$, and C may be regarded as an A/I-submodule of $(A/I)^l$. Conversely, every quasicyclic code may be obtained in this way from a submodule of A^l for suitable l, m.

Thus, a k-generator quasicyclic code C is an A/I-submodule generated by a set $\{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_k\} \subseteq (A/I)^l$ and its preimage (with the appropriate notational interpretation) is the A-submodule $M \subseteq A^l$ generated by $\{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_k, K\}$. Let $\mathcal{G}$ be the reduced Gröbner basis of M. Since K contains $(x^m - 1)\mathbf{e}_i$ for each i and since $\mathrm{lt}((x^m - 1)\mathbf{e}_i)$ is in the i^{th} position, there exists $\mathbf{g} \in \mathcal{G}$ with $\mathrm{lt}(\mathbf{g})$ dividing $\mathrm{lt}((x^m - 1)\mathbf{e}_i)$. It follows that the leading term of $\mathbf{g}$ is in the i^{th} position and as a consequence we have

Theorem 2.1 *The reduced Gröbner basis $\mathcal{G}$ of M contains precisely l elements $\mathbf{g}_i, i = 1, 2, \ldots, l$ where $\mathbf{g}_i$ has leading term in the i^{th} position. Moreover, $\mathbf{g}_i$ has the form $\mathbf{g}_i = (0, 0, \ldots, 0, g_i^i, g_i^{i+1}, \ldots, g_i^l) \in A^l$, where $g_i^i \neq 0$.*

Thus, the elements of $\mathcal{G}$ have the triangular form

$$\begin{aligned}
\mathbf{g}_1 &= (g_1^1, g_1^2, \ldots, g_1^l) \\
\mathbf{g}_2 &= (0, g_2^2, \ldots, g_2^l) \\
\mathbf{g}_3 &= (0, 0, g_3^3, \ldots, g_3^l) \\
&\vdots \\
\mathbf{g}_l &= (0, 0, 0, \ldots, 0, g_l^l).
\end{aligned}$$

where the nonzero polynomials g_i^i are called the *diagonal elements* of $\mathcal{G}$. Some additional properties are given in

Theorem 2.2

1. *Each* g_i^i *is monic and divides* $x^m - 1$.
2. *If* $\mathbf{f} \in M$ *has leading term in the* i^{th} *position then* g_i^i *divides the* i^{th} *component of* f.
3. *The degrees of non-zero off-diagonal components* $g_i^j (j \neq i)$ *satisfy*

$$\deg(g_i^j) < \deg(g_j^j) \leq m.$$

4. *If* $\mathbf{g}_i \in K$ *then* $\mathbf{g}_i = (x^m - 1)\mathbf{e}_i$.

A generating set $\mathcal{B}$ for the corresponding quasicyclic code C can be found by mapping $\mathcal{G}$ to $A^l/K \cong (A/I)^l$. Note that elements of $\mathcal{G} \cap K$ are sent to zero so $\mathcal{B}$ may contain fewer than l elements.

EXAMPLE 2.3 Let C be the quasicyclic code of index $l = 3$ and length $n = ml = 18$, $m = 6$ over F_2 generated by elements

$$\begin{aligned} \mathbf{a}_1 &= (x^5 + x^4 + x^3 + x^2 + x + 1, x^4 + x^3 + x + 1, x^5 + x^4 + x^3 + x^2 + x + 1) \\ \mathbf{a}_2 &= (x^5 + x^4 + x + 1, x^4 + x^3 + 1, x^5 + x^4 + x^3). \end{aligned}$$

The Gröbner basis of $M = \langle \mathbf{a}_1, \mathbf{a}_2, (x^6 - 1)\mathbf{e}_1, (x^6 - 1)\mathbf{e}_2, (x^6 - 1)\mathbf{e}_3 \rangle$ is $\mathcal{G} = \{\mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_3\}$ where

$$\begin{aligned} \mathbf{g}_1 &= (x + 1, x + 1, x^4 + x^3 + x + 1) \\ \mathbf{g}_2 &= (0, x^2 + x + 1, x^4 + x^2 + 1) \\ \mathbf{g}_3 &= (0, 0, x^6 - 1). \end{aligned}$$

Since $x^6 - 1 = (x + 1)^2(x^2 + x + 1)^2$ in $F_2[x]$ the diagonal elements $g_1^1 = x + 1$, $g_2^2 = x^2 + x + 1, g_3^3 = x^6 - 1$ are indeed divisors of $x^m - 1$. The corresponding generating set of C is $\mathcal{B} = \{\mathbf{g}_1, \mathbf{g}_2\}$ since $\mathbf{g}_3$ is mapped to zero.

□

We can determine the dimension k of the code from the diagonal elements g_i^i as follows.

Theorem 2.4 *The codimension* $n-k$ *of the code* C *is the number of monomials* $x^i\mathbf{e}_j \in A^l$ *in normal form modulo the Gröbner basis* $\mathcal{G}$ *of* M. *Thus*

$$\begin{aligned} k &= ml - \sum_{i=1}^{l} \deg(g_i^i) \\ &= \sum_{i=1}^{l} (m - \deg(g_i^i)). \end{aligned}$$

EXAMPLE 2.5 (Continued from example 2.3) The diagonal elements are $g_1^1 = x+1$, $g_2^2 = x^2+x+1$, and $g_3^3 = x^6-1$ and therefore the dimension k of the code C is

$$k = ml - \sum_{i=1}^{l} \deg(g_i^i) = 18 - (1+2+6) = 9.$$

EXAMPLE 2.6 Consider a quasicyclic code C of index $l = 5$ and length $n = ml = 60$, $m = 12$ over F_{16} . Let α be a root of the primitive polynomial x^4+x+1 over F_2 and let C be generated by elements

$$\begin{aligned}
\mathbf{a}_1 &= (\alpha^{10} + \alpha^{10}x + x^2 + x^3 + \alpha^{10}x^6 + \alpha^{10}x^7 + x^8 + x^9,\\
&\quad \alpha^5 + \alpha^{10}x + \alpha^{10}x^2 + x^3 + \alpha^5x^6 + \alpha^{10}x^7 + \alpha^{10}x^8 + x^9,\\
&\quad \alpha^{14} + \alpha^2x + \alpha^3x^2 + \alpha^{13}x^3 + \alpha^8x^4 + \alpha^3x^5 + \alpha^2x^6 + x^7,\\
&\quad \alpha^5 + x^2 + \alpha^5x^4 + x^6,\\
&\quad \alpha^{10} + x + \alpha^{10}x^4 + x^5)\\
\mathbf{a}_2 &= (\alpha^{10} + \alpha^{10}x + x^2 + x^3 + \alpha^{10}x^6 + \alpha^{10}x^7 + x^8 + x^9,\\
&\quad \alpha^5 + x + \alpha^5x^2 + \alpha^{10}x^3 + \alpha^{10}x^4 + \alpha^{10}x^5 + x^6 + \alpha^5x^7 + x^8,\\
&\quad \alpha^{14} + \alpha^2x + \alpha^3x^2 + \alpha^{13}x^3 + \alpha^8x^4 + \alpha^3x^5 + \alpha^2x^6 + x^7,\\
&\quad \alpha^5 + x^2 + \alpha^5x^4 + x^6,\\
&\quad 1 + x^4)
\end{aligned}$$

in $(A/I)^5$. The reduced Gröbner basis for $M = \langle \mathbf{a}_1, \mathbf{a}_2, (x^{12}-1)\mathbf{e}_1, \ldots, (x^{12}-1)\mathbf{e}_5 \rangle$ is $\mathcal{G} = \{\mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_3, \mathbf{g}_4, \mathbf{g}_5\}$ where

$$\begin{aligned}
\mathbf{g}_1 &= (\alpha^{10} + \alpha^{10}x + x^2 + x^3 + \alpha^{10}x^6 + \alpha^{10}x^7 + x^8 + x^9, 0,\\
&\quad \alpha^{14} + \alpha^2x + \alpha^3x^2 + \alpha^{13}x^3 + \alpha^8x^4 + \alpha^3x^5 + \alpha^2x^6 + x^7,\\
&\quad \alpha^5 + x^2 + \alpha x^4 + 1x^6, \alpha^5 + \alpha^5x + \alpha x^4 + \alpha^5x^5)\\
\mathbf{g}_2 &= (0, \alpha^5 + x + \alpha^5x^2 + \alpha^{10}x^3 + \alpha^{10}x^4 + \alpha^{10}x^5 + x^6 + \alpha x^7 + x^8,\\
&\quad 0, 0, \alpha^{10} + \alpha^5x + \alpha^{10}x^4 + \alpha^5x^5)\\
\mathbf{g}_3 &= (0, 0, 1 + \alpha^{10}x + \alpha^{10}x^4 + \alpha^5x^5 + \alpha^5x^8 + x^9,\\
&\quad 1 + \alpha^{10}x + \alpha^2x^2 + \alpha^{10}x^4 + \alpha^5x^5 + \alpha^{12}x^6 + \alpha^5x^8 + x^9 + \alpha^7x^{10},\\
&\quad \alpha^{12} + \alpha^2x + \alpha^{12}x^4 + \alpha^2x^5)\\
\mathbf{g}_4 &= (0, 0, 0,\\
&\quad \alpha^{10} + \alpha^5x + x^2 + \alpha^{10}x^3 + \alpha^5x^4 + x^5 + \alpha^{10}x^6 + \alpha^5x^7 + x^8 + \alpha^{10}x^9 + \alpha^5x^{10} + x^{11},\\
&\quad 0)\\
\mathbf{g}_5 &= (0, 0, 0, 0, 1 + x + x^2 + x^4 + x^5 + x^6).
\end{aligned}$$

The polynomial $x^{12}-1$ splits into 3 distinct factors, each of multiplicity 4 over F_{16},

$$x^{12}-1=(x+1)^4(x+\alpha^5)^4(x+\alpha^{10})^4.$$

and, as a consequence, the diagonal elements also split:

$$\begin{aligned}
g_1^1 &= \alpha^{10}+\alpha^{10}x+x^2+x^3+\alpha^{10}x^6+\alpha^{10}x^7+x^8+x^9\\
&= (x+1)^3(x+\alpha^5)^4(x+\alpha^{10})^2\\
g_2^2 &= \alpha^5+x+\alpha^5x^2+\alpha^{10}x^3+\alpha^{10}x^4+\alpha^{10}x^5+x^6+\alpha x^7+x^8\\
&= (x+1)(x+\alpha^5)^4(x+\alpha^{10})^3\\
g_3^3 &= 1+\alpha^{10}x+\alpha^{10}x^4+\alpha^5x^5+\alpha^5x^8+x^9\\
&= (x+1)^4(x+\alpha^5)(x+\alpha^{10})^4\\
g_4^4 &= \alpha^{10}+\alpha^5x+x^2+\alpha^{10}x^3+\alpha^5x^4+x^5+\alpha^{10}x^6+\alpha^5x^7+x^8+\alpha^{10}x^9+\alpha^5x^{10}+x^{11}\\
&= (x+1)^4(x+\alpha^5)^3(x+\alpha^{10})^4\\
g_5^5 &= 1+x+x^2+x^4+x^5+x^6\\
&= (x+1)^4(x+\alpha^5)(x+\alpha^{10}).
\end{aligned}$$

The corresponding generating set for C is $\mathcal{B}=\{\mathbf{g}_1,\mathbf{g}_2,\mathbf{g}_3,\mathbf{g}_4,\mathbf{g}_5\}$ and the dimension k of the code is

$$k=ml-\sum_{i=1}^{l}\deg(g_i^i)=60-(9+8+9+11+6)=17.$$

We note that the formula for dimension k of C can also be written

$$k=\sum_{\mathbf{g}_i\in\mathcal{B}}(m-deg(g_i^i))$$

since if $\mathbf{g}_i\in\mathcal{G}\setminus\mathcal{B}$ then $\mathbf{g}_i=(x^m-1)\mathbf{e}_i$ and $m-deg(g_i^i)=m-m=0$. A generator matrix for C can be constructed directly from $\mathcal{B}$.

Theorem 2.7 *The set*

$$\left\{x^k\mathbf{g}_i \bmod x^m-1 \mid \mathbf{g}_i\in\mathcal{B}, 1\le i\le l, 0\le k\le m-deg(g_i^i)-1\right\}$$

is a linearly independent set of $\sum_{\mathbf{g}_i\in\mathcal{B}}(m-deg(g_i^i))$ *vectors which spans the code* C.

Thus we can construct a generator matrix in the form of a $|\mathcal{B}|\times l$ block upper triangular matrix $[A_{ij}]$ with $A_{ij}=0$ for $i>j$ and for $i\le j$

$$A_{ij}=\begin{pmatrix} g_i^j \\ xg_i^j \\ \vdots \\ x^{m-\deg(g_i^i)-1}g_i^j \end{pmatrix}.$$

The specific form of the diagonal elements is

$$\begin{pmatrix} g_i^{i,0} & g_i^{i,1} & \cdots & g_i^{i,\deg(g_i^i)} & 0 & \cdots & 0 \\ 0 & g_i^{i,0} & g_i^{i,1} & \cdots & g_i^{i,\deg(g_i^i)} & \cdots & 0 \\ \vdots & & \ddots & \ddots & & \ddots & \vdots \\ 0 & 0 & \cdots & g_i^{i,0} & g_i^{i,1} & \cdots & g_i^{i,\deg(g_i^i)} \end{pmatrix}$$

where $g_i^i(x) = g_i^{i,0} + g_i^{i,1}x + \cdots + g_i^{i,\deg(g_i^i)}x^{\deg(g_i^i)} \in A/I$. The analogy with the generator matrix of a cyclic code is apparent.

EXAMPLE 2.8 Continuing from example 2.3 and 2.5. We have $l = 3$, $m = 6$ and the generators of C are

$$\begin{aligned} \mathbf{g}_1 &= (x+1, x+1, x^4+x^3+x+1) \\ \mathbf{g}_2 &= (0, x^2+x+1, x^4+x^2+1). \end{aligned}$$

The generator matrix is

$$\begin{pmatrix} 110000 & 110000 & 110110 \\ 011000 & 011000 & 011011 \\ 001100 & 001100 & 101101 \\ 000110 & 000110 & 110110 \\ 000011 & 000011 & 011011 \\ 000000 & 111000 & 101010 \\ 000000 & 011100 & 010101 \\ 000000 & 001110 & 101010 \\ 000000 & 000111 & 010101 \end{pmatrix}.$$

3 Structure of the Gröbner basis

Given an arbitrary set of k generators for a quasicyclic code C, we have seen how to construct a canonical generating set for C from the reduced Gröbner basis of the corresponding preimage module. We now study the general structure of this representation with the aim of constructing and classifying quasicyclic codes independently of a specific initial generating set.

Theorem 3.1 *The set* $\mathcal{G} = \{\mathrm{g}_1, \mathrm{g}_2, \ldots, \mathrm{g}_l\} \subseteq A^l$ *of the form*

$$\begin{aligned} \mathrm{g}_1 &= (g_1^1, g_1^2, \ldots, g_1^l) \\ \mathrm{g}_2 &= (0, g_2^2, \ldots, g_2^l) \\ \mathrm{g}_3 &= (0, 0, g_3^3, \ldots, g_3^l) \\ &\vdots \\ \mathrm{g}_l &= (0, 0, \ldots, 0, g_l^l) \end{aligned}$$

is a reduced Gröbner basis for a submodule M *containing* $K = \langle\{(x^m - 1)\mathbf{e}_i \mid i = 1, 2, \ldots, l\}\rangle$, *and so corresponds to a generating set for a quasicyclic code, if and only if:*

(i) *the diagonal elements* g_i^i *are monic polynomials*

(ii) $\deg(g_i^j) < \deg(g_j^j)$ *for all* $i < j$, $1 \leq i, j \leq l$.

(iii) $(x^m - 1)\mathbf{e}_i \in \langle\mathcal{G}\rangle$ *for all* $i = 1, \ldots, l$.

Conversely, every quasicyclic code can be represented in this way.

These conditions impose strong restrictions on the elements of the reduced Gröbner basis $\mathcal{G}$. For example, it is immediate that $g_i^i \mid x^m - 1$. Further divisibility conditions on the non-zero off-diagonal elements g_i^j, $j > i$, can also be specified. Two special cases are described next.

Theorem 3.2 *Let* $\mathcal{G} = \{\mathrm{g}_1, \mathrm{g}_2, \ldots, \mathrm{g}_l\}$ *be the reduced Gröbner basis for the submodule* M. *If some* $\mathrm{g}_i \in \mathcal{G}$, *is not contained in* K *but* $\mathrm{g}_j \in K$ *for all* $i < j \leq l$, *then* g_i *has the form*

$$\mathrm{g}_i = (0, 0, \ldots, 0, g_i^i, f_1(x)g_i^i, \ldots, f_{l-i}(x)g_i^i)$$

where

$$\deg(f_k(x)) < m - \deg(g_i^i),\ 1 \leq k \leq l - i.$$

We call $\mathcal{G}$ an *r–level* Gröbner basis of M if there is an index r such that $\mathrm{g}_r \notin K$ and $\mathrm{g}_j \in K$ for $r < j \leq l$. The corresponding generating set $\mathcal{B}$ of the code C is also called an r–level generating set. It contains at most r generators

$$\mathcal{B} = \{\mathrm{g}_i \in \mathcal{G} \mid \mathrm{g}_i \notin K, 1 \leq i \leq r\}.$$

We then have the following consequence.

Corollary 3.3 *A quasicyclic code C of index l and length ml has a 1-level generating set if it is generated by a single generator* $\mathbf{g}$ *of the form*

$$\mathbf{g} = (g, f_1 g, \ldots, f_{l-1} g)$$

where g divides $x^m - 1$ and $\deg(f_k) < m - \deg(g)$, $1 < k \leq l - 1$.

Finally, in the index 2 case we have

Theorem 3.4 *Let the index l be 2 and suppose that $\mathcal{G} = \{\mathbf{g}_1, \mathbf{g}_2\}$ is the reduced Gröbner basis for M. Then*

$$\begin{aligned} \mathbf{g}_1 &= (g_1^1, g_1^2) \\ \mathbf{g}_2 &= (0, g_2^2) \end{aligned}$$

where g_i^i divides $x^m - 1$, $i = 1, 2$, and

$$g_1^2 = \frac{f_1 g_2^2}{\gcd\left(\frac{x^m-1}{g_1^1}, g_2^2\right)}$$

where $\deg(f_1) < \deg(g_2^2) - \deg\left(g_2^2/\gcd\left(\frac{x^m-1}{g_1^1}, g_2^2\right)\right)$. If m is relatively prime to the characteristic of the field F then the non-diagonal entry simplifies to

$$g_1^2 = f_1 \gcd\left(g_1^1, g_2^2\right)$$

where $\deg(f_1) < \deg(g_2^2) - \deg(\gcd(g_1^1, g_2^2))$.

4 Diagonal elements of a 1-generator quasicyclic code

We now consider the particular case of a 1-generator quasicyclic code. This has been considered by several authors including [8],[10],[11]. The following theorem gives the structure of the diagonal elements of the Gröbner basis of of the preimage module (and hence those of the canonical generating set of the code).

Theorem 4.1 *A quasicyclic code C generated by $\mathbf{g} = (g_1, g_2, \ldots, g_l)$ has a corresponding preimage $M = \langle \mathbf{g}, (x^m - 1)\mathbf{e}_1, (x^m - 1)\mathbf{e}_2, \ldots, (x^m - 1)\mathbf{e}_l \rangle$ whose reduced Gröbner basis is $\mathcal{G} = \{\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_l\}$ where*

$$g_1^1 = \gcd(g_1, x^m - 1)$$

$$g_i^i = \frac{(x^m - 1)\gcd(g_1, g_2, \ldots, g_i, x^m - 1)}{\gcd(g_1, g_2, \ldots, g_{i-1}, x^m - 1)}, \quad i = 2, \ldots, l.$$

¿From theorem 2.4 the dimension of the code is

$$k = m - \deg(\gcd(g_1, g_2, \ldots, g_l, x^m - 1)).$$

which corresponds to the formula given by Séguin and Drolet[10].

This result can be used to develop a technique for constructing good 1-generator quasicyclic codes generated by an element of the form $\mathbf{g} = (f_1 g, f_2 g, \ldots, f_l g)$. We end with an analogue of the BCH bound in this case.

Theorem 4.2 *Let m be relatively prime to the characteristic of F and let C be a 1-generator quasicyclic code generated by*

$$\mathbf{g} = (f_1 g, f_2 g, \ldots, f_l g)$$

where $g \mid x^m - 1$, $\gcd(f_i, \frac{x^m-1}{g}) = 1$, and $\deg(f_i) < m - \deg(g(x)), 1 \leq i \leq l$. Then

$$l(\#ConsecutiveRoots(g) + 1) \leq d_{\min}(C) \leq l(\deg(g) + 1).$$

References

[1] W.W. Adams and P. Loustaunau, *An Introduction to Gröbner Bases*, Graduate Studies in Mathematics, vol. 3, American Mathematical Society, 1994.

[2] T. Becker and V. Weispfenning, *Gröbner Bases: A Computational Approach to Commutative Algebra*, Springer-Verlag, New York, 1993.

[3] P. Fitzpatrick and J. Flynn, A Gröbner basis technique for Padé approximation, *J. Symbolic Computation,* 13 (1992), 133–138.

[4] P. Fitzpatrick, New time domain errors and erasures decoding algorithm for BCH codes, *Elect. Lett.*, 30:2 (1994), 110–111.

[5] P. Fitzpatrick, On the key equation, *IEEE Transactions on Information Theory*, 41, no. 5 (1995), 1290–1302.

[6] P. Fitzpatrick, On the scalar rational interpolation problem, *Mathematics of Control, Signals, and Systems*, 9 (1996), 352–369.

[7] P. Fitzpatrick, Solving a multivariable congruence by change of term order, *J. Symbolic Computation*, 24 (1997), 505–510.

[8] T.A. Gulliver and V.K. Bhargava, Some best rate $1/p$ and rate $(p-1)/p$ systematic quasicyclic codess, *IEEE Trans. Inform. Theory*, IT-37 (1991), 552–555.

[9] J. Little, K. Saints, C. Heegard, On the structure of Hermitian codes, *J. Pure and Appl. Algebra*, 121 (1997), 293–314.

[10] G.E. Séguin and G. Drolet, The theory of 1-generator quasi-cyclic codes, preprint, Dept. of Electrical and Computer Engineering, Royal Military College of Canada, Kingston, Ontario, June 1990.

[11] H.C.A van Tilborg, On quasi-cyclic codes with rate $1/m$, *IEEE Trans. Inform. Theory*, IT-24 (1978), 628–630.

On the Complexity of Calculating the Minimum Norm of a Binary Code

Iiro Honkala
Department of Mathematics
University of Turku
20014 Turku, Finland
e-mail: honkala@utu.fi

Antoine Lobstein
Centre National de la Recherche Scientifique
Ecole Nationale Supérieure des Télécommunications
46 rue Barrault, 75634 Paris cédex 13, France
e-mail: lobstein@inf.enst.fr

Abstract

The problem of upper bounding the minimum norm of a binary code is shown to be co-NP-complete, and Π_2-complete if the code is known to be linear and the input consists of a parity check matrix for the code.

1 Introduction

The covering radius of a binary code $C \subseteq \mathbb{F}_2^n$ is the smallest integer R with the property that every point in the space $\mathbb{F}_2^n$ is withing Hamming distance R from at least one codeword. For the theory of covering codes, we refer to [2].

For $i = 1, 2, \ldots, n$, let $C_0^{(i)}$ (respectively, $C_1^{(i)}$) denote the set of codewords in which the i-th coordinate is 0 (respectively, 1). The integer

$$N^{(i)} = \max_{\mathbf{x} \in \mathbb{F}_2^n} \{d(\mathbf{x}, C_0^{(i)}) + d(\mathbf{x}, C_1^{(i)})\}$$

is called the norm of C with respect to the i-th coordinate and

$$N_{\min} = \min_i N^{(i)}$$

is called the minimum norm of C. This concept, introduced in [6], is important in building covering codes using the amalgamated direct sum construction.

The complexity of determining the minimum norm of a binary code is mentioned as an open problem in [3]. Here, we assume that the reader is familiar with NP-complete problems and the polynomial-time hierarchy (see, e.g., [5] or [1] for a whole theory of complexity, or [7] for a brief introduction to what will be necessary in the following).

We consider separately the following two variants:

> NAME: NORM1
>
> INSTANCE: A binary code C of length n (given as a list of codewords) and an integer w
>
> QUESTION: Is the minimum norm of C at most w?

and

> NAME: NORM2
>
> INSTANCE: A binary linear code C of length n and dimension k with a given parity check $k \times n$ matrix $\mathbf{H}$, and an integer w
>
> QUESTION: Is the minimum norm of C at most w?

In the former case the input consists of a list of all codewords (and therefore the size of an instance is $n \cdot |C|$) whereas in the second we assume that we are given a parity check matrix of the code and therefore the input size is essentially smaller (namely, $n \cdot k = n \cdot \log_2 |C|$). It is known that upper bounding the covering radius in these two cases is co-NP-complete [4], respectively Π_2-complete [7]. We prove that the same is true for the minimum norm. This is not surprising in view of the fact that it is easy to see that NORM1 belongs to co-NP and NORM2 belongs to Π_2, and intuitively it would seem that upper bounding the minimum norm is not easier than upper bounding the covering radius.

2 The Nonlinear Case

Before proving the co-NP-completeness of NORM1 (Theorem 1 below), we give the following definition and notation.

We say that a vector $\mathbf{v} = (v_1, v_2, \ldots, v_{2n}) \in \mathbb{F}_2^{2n}$ is *doubled* if and only if $v_{2i-1} = v_{2i}$ for all $i = 1, 2, \ldots, n$. If $\mathbf{u} \in \mathbb{F}_2^n$, we denote its complement $\mathbf{1} + \mathbf{u}$ by $\overline{\mathbf{u}}$; here $\mathbf{1}$ denotes the all-one vector of length n.

Let $\mathbf{u}(i) \in \mathbb{F}_2^{2i}$ denote the vector $(0101\ldots01)$. Let $Y_{2n}^1 = \{(01|\mathbf{u}(n-1)), (10|\mathbf{u}(n-1)), (01|\overline{\mathbf{u}}(n-1)), (10|\overline{\mathbf{u}}(n-1))\}$, where $|$ stands for concatenation, and $Y_{2n}^j = s_j(Y_{2n}^1)$, where s_j denotes the circular right shift of $2j-2$ bits, for $j = 2, 3, \ldots, n$. Finally, let $Y_{2n} = \bigcup_{j=1}^{n} Y_{2n}^j$. Then $|Y_{2n}| = 2n + 2$, since $(01|\mathbf{u}(n-1))$ and $(10|\overline{\mathbf{u}}(n-1))$ are invariant under all even circular shifts.

We are now ready to prove the following:

Theorem 1 *The problem NORM1 is co-NP-complete.*

Proof. We show that the complementary problem, lower bounding the minimum norm (co-NORM1), is NP-complete. Clearly, it belongs to NP; indeed, given n vectors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$, of length n, it is polynomial (in $n \cdot |C|$, the size of an instance of co-NORM1) to check that the n sums $d(\mathbf{x}_i, C_0^{(i)}) + d(\mathbf{x}_i, C_1^{(i)})$ are all greater than w.

Now we reduce 3-satisfiability, which is known to be NP-complete (see, e.g., [5, p. 48]), to co-NORM1, using the same code C as in [4].

> NAME: 3-satisfiability (3-SAT)
>
> INSTANCE: A boolean formula E, in conjunctive normal form, with exactly three distinct literals in each clause
>
> QUESTION: Can E be satisfied?

Starting from any instance of 3-SAT, we construct, in polynomial time, an instance of co-NORM1 in such a way that the positive and negative instances correspond in co-NORM1 and 3-SAT. Let $E = \mathcal{C}_1 \wedge \mathcal{C}_2 \wedge \ldots \wedge \mathcal{C}_m$ be an instance of 3-SAT, each clause $\mathcal{C}_j$, defined over the set of variables $\{x_1, x_2, \ldots, x_n\}$, consisting of exactly three distinct literals. For each clause $\mathcal{C}_j$, let $\mathbf{z}(\mathcal{C}_j) = (z_1, \ldots, z_{2n}) \in \mathbb{F}_2^{2n}$ be the vector defined by

$z_{2i-1} = z_{2i} = 1$ if $\mathcal{C}_j$ contains the literal x_i;

$z_{2i-1} = z_{2i} = 0$ if $\mathcal{C}_j$ contains the literal $\overline{x}_i$;

$z_{2i-1} = 0$ and $z_{2i} = 1$ otherwise.

We then define $C \subseteq \mathbb{F}_2^{2n+2}$ as follows:

$$C = \{(\mathbf{z}(\mathcal{C}_j)|00) : 1 \leq j \leq m\} \cup Y_{2n+2}.$$

Finally, let $w = 2n + 1$. The code C has length $2n + 2$ and cardinality $m + 4 + 2n$, which is polynomial in nm, the size of the instance of 3-SAT. We now have to show that E can be satisfied if and only if C has minimum norm at least $2n + 2$.

First suppose that E can be satisfied; a truth assignment to the variables $\{x_1, x_2, \ldots, x_n\}$ satisfying E can be represented by a vector $\mathbf{v} = (v_1, v_2, \ldots, v_n) \in \mathbb{F}_2^n$. Let $\mathbf{v}^*$ be the doubled vector $(v_1, v_1, v_2, v_2, \ldots, v_n, v_n, 0, 0) \in \mathbb{F}_2^{2n+2}$. Then for all $\mathbf{c} \in Y_{2n+2}$, $d(\mathbf{c}, \mathbf{v}^*) = n + 1$. Moreover, in each clause $\mathcal{C}_j$ there is at least one literal which is set true by $\mathbf{v}$; it is easy to see then that $d((\mathbf{z}(\mathcal{C}_j)|00), \mathbf{v}^*) \leq 2 + 2 + 0 + (n - 3) = n + 1$. Consequently, for all $\mathbf{c} \in C$, $d(\mathbf{c}, \mathbf{v}^*) \leq n + 1$. This implies that $d(C, \overline{\mathbf{v}^*}) \geq 2n + 2 - (n + 1) = n + 1$, so

$R \geq n+1$. Trivially $N_{\min} \geq 2n+2$, and in fact equality holds in the last two inequalities, because the code has length $2n+2$ and contains a word–complement pair.

We claim that conversely, if $N_{\min} \geq 2n+2$ (i.e., $N_{\min} = 2n+2$), then E can be satisfied. By the assumption, $N^{(2n+2)} = 2n+2$, and therefore there exists a vector $\mathbf{v} = (v_1, v_2, \ldots, v_{2n+1}, v_{2n+2})$ such that

$$d(\mathbf{v}, C_0^{(2n+2)}) + d(\mathbf{v}, C_1^{(2n+2)}) = 2n+2.$$

Then

$$v_1 = v_2, v_3 = v_4, \ldots, v_{2n-1} = v_{2n}; \tag{2}$$

indeed, if for example $v_1 = 1$ and $v_2 = 0$, the fact that $10\,01\,01\ldots01$ and $10\,10\,10\ldots10$ belong to the code would imply that

$$d(\mathbf{v}, C_0^{(2n+2)}) + d(\mathbf{v}, C_1^{(2n+2)}) \leq d(\mathbf{v}, 101010\ldots10) + d(\mathbf{v}, 100101\ldots01) = 2n.$$

Assume first that $v_{2n+1} = v_{2n+2}$. Since all the codewords from $C \setminus Y_{2n+2}$ belong to $C_0^{(2n+2)}$, we know by (2) that $d(\mathbf{v}, C_1^{(2n+2)}) = n+1$, and $d(\mathbf{v}, C_0^{(2n+2)}) = 2n+2-d(\mathbf{v}, C_1^{(2n+2)}) = n+1$; consequently $d(\mathbf{v}, C) = n+1$.

Assume now that $v_{2n+1} = 0$ and $v_{2n+2} = 1$. Then $d(\mathbf{v}, C_1^{(2n+2)}) = n$ and $d(\mathbf{v}, C_0^{(2n+2)}) = 2n+2-d(\mathbf{v}, C_1^{(2n+2)}) = n+2$. But then the vector $\mathbf{v}'$ obtained by changing the last 1 to 0 in $\mathbf{v}$ satisfies the conditions $d(\mathbf{v}', C_1^{(2n+2)}) = n+1$ and $d(\mathbf{v}', C_0^{(2n+2)}) = n+1$, and therefore $d(\mathbf{v}', C) = n+1$. The last case $v_{2n+1} = 1$ and $v_{2n+2} = 0$ is similar.

We have shown that in all cases there exists a vector ($\mathbf{v}$ or $\mathbf{v}'$), which is *doubled* and has distance at least $n+1$ to all the codewords in C.

Complementing $\mathbf{v}$ or $\mathbf{v}'$ yields a doubled vector $\mathbf{v}^* = (v_1'', v_1'', \ldots, v_{n+1}'', v_{n+1}'') \in \mathbb{F}_2^{2n+2}$ such that $d(\mathbf{v}^*, \mathbf{c}) \leq n+1$ for all $\mathbf{c} \in C$. In particular, for all j, $d(\mathbf{v}^*, (\mathbf{z}(\mathcal{C}_j)|00)) \leq n+1$, which implies $d((v_1'', v_1'', \ldots, v_n'', v_n''), \mathbf{z}(\mathcal{C}_j)) \leq n+1$. Let $\mathbf{v}'' = (v_1'', v_2'', \ldots, v_n'')$. The structure of $\mathbf{z}(\mathcal{C}_j)$ shows that there exists $i \in \{1, 2, \ldots, n\}$ such that $z_{2i-1} = z_{2i}$ and $d((v_i'', v_i''), (z_{2i-1}, z_{2i})) = 0$. This means that the truth assignment defined by $\mathbf{v}''$ satisfies the clause $\mathcal{C}_j$: if $z_{2i-1} = z_{2i} = v_i'' = 1$ (respectively, 0), then the variable x_i is set true (respectively, false) and x_i (respectively, $\overline{x}_i$) belongs to $\mathcal{C}_j$. Therefore E is satisfied.

Together with co-NORM1$\in$NP, this shows that co-NORM1 is NP-complete or, equivalently, that NORM1 is co-NP-complete. □

3 The Linear Case

Theorem 3 *The problem NORM2 is* Π_2*-complete.*

Proof. First we prove that the complementary problem (lower bounding the minimum norm) is in co-Π_2. Roughly speaking, a problem is in co-Π_2 if it is solvable in polynomial time by a nondeterministic algorithm with access to an oracle providing single-step solutions to some problem in NP. To this end, we choose the Linear Decoding (LD) problem: given a linear code, a vector $\mathbf{v}$ and an integer p, is there a codeword at distance at most p from $\mathbf{v}$? Now, given a guess consisting of n vectors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$, of length n, we have to check that the n sums $d(\mathbf{x}_i, C_0^{(i)}) + d(\mathbf{x}_i, C_1^{(i)})$ are all greater than w. This can be done polynomially in n, with the help of an oracle solving LD in one step.

Next, to NORM2 we reduce the following problem, which is Π_2-complete. We use the same matrix $\mathbf{H}$ as in [7].

> NAME: $\forall\exists$-3-dimensional matching ($\forall\exists$-3-DM)
>
> INSTANCE: Two disjoint subsets M_1 and M_2 of $X_1 \times X_2 \times X_3$, where X_1, X_2 and X_3 are three disjoint sets of the same cardinality
>
> QUESTION: Is it true that $\forall S_1 \subseteq M_1, \exists S_2 \subseteq M_2$, such that $S_1 \cup S_2$ is a matching?

A *matching* S is a subset of $M_1 \cup M_2$ with $|X_1|$ elements such that no two triples in S agree in any coordinate. Notice that we can assume, without loss of generality, that every element of $X_1 \cup X_2 \cup X_3$ is contained in at least one triple in $M_1 \cup M_2$ (otherwise, the answer is trivially NO).

Starting from any instance of $\forall\exists$-3-DM, we construct, in polynomial time, an instance of NORM2 in such a way that the positive and negative instances correspond in $\forall\exists$-3-DM and NORM2.

Let M_1, M_2, X_1, X_2, X_3 be an instance of $\forall\exists$-3-DM. Let $M = M_1 \cup M_2$ (so that $|M| = |M_1| + |M_2|$), $p = |X_i|$, and $X_i = \{x_{i,1}, x_{i,2}, \ldots, x_{i,p}\}$ for $i = 1, 2, 3$. Let $w = 2p + 1$ and

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_1 \\ \mathbf{H}_2 \end{bmatrix}$$

be a (binary) matrix of dimensions $(3p + |M_1|) \times 8|M|$, where $\mathbf{H}_1$ and $\mathbf{H}_2$ have dimensions $3p \times 8|M|$ and $|M_1| \times 8|M|$, respectively. If $t = (x_{1,i}, x_{2,j}, x_{3,k})$ belongs to M, then we let $\mathbf{H}_1$ contain one column, $\mathbf{H}_1^{t(0)}$, with exactly three 1's in the positions corresponding to $x_{1,i}, x_{2,j}$ and $x_{3,k}$, and seven columns, $\mathbf{H}_1^{t(1)}$, $\mathbf{H}_1^{t(2)}$, ..., $\mathbf{H}_1^{t(7)}$, obtained from $\mathbf{H}_1^{t(0)}$ by replacing 1's by 0's in all possible ways. So each triple $t \in M$ is associated with eight columns in $\mathbf{H}$.

Each row in $\mathbf{H}_2$ corresponds to a triple in M_1, with 1's in the eight columns associated with this triple and 0's elsewhere.

Notice that the assumption that every element of $X_1 \cup X_2 \cup X_3$ is contained in at least one triple in M guarantees that the matrix $\mathbf{H}$ is of full rank.

This construction is polynomial in the size of the instance of $\forall\exists$-3-DM. We now have to show that $\forall S_1 \subseteq M_1, \exists S_2 \subseteq M_2$, such that $S_1 \cup S_2$ is a matching, if and only if the above described matrix $\mathbf{H}$ is the parity check matrix of a code C with minimum norm at most w.

First suppose that for all $S_1 \subseteq M_1$, there exists $S_2 \subseteq M_2$, such that $S_1 \cup S_2$ is a matching. Let $\mathbf{y}$ be any vector of length $3p + |M_1|$. The last $|M_1|$ coordinates of $\mathbf{y}$ correspond to the triples in M_1 and the 1's in these locations select a subset S_1 of M_1. Choose $S_2 \subseteq M_2$, such that $S = S_1 \cup S_2$ is a matching. Let $\mathbf{y}'$ be the vector of length $3p$ obtained from $\mathbf{y}$ by taking its first $3p$ components. Let $*$ stand for the componentwise product. Because $\sum_{t \in S} \mathbf{H}_1^{t(0)}$ is the all-one vector of length $3p$, we get: $\mathbf{y}' = \left(\sum_{t \in S} \mathbf{H}_1^{t(0)}\right) * \mathbf{y}' = \sum_{t \in S} \left(\mathbf{H}_1^{t(0)} * \mathbf{y}'\right)$. Since $\mathbf{H}_1^{t(0)} * \mathbf{y}' = \mathbf{H}_1^{t(j)}$ for some j between 0 and 7, this means that $\mathbf{y}'$ is the sum of $|S| = p$ columns of $\mathbf{H}_1$. But the way $\mathbf{H}$ was constructed and S_1 was chosen from $\mathbf{y}$ also shows that the sum of these same p columns of $\mathbf{H}$ is equal to $\mathbf{y}$, i.e., $\mathbf{y} = \mathbf{H}\mathbf{x}^T$, with $wt(\mathbf{x}) = p$. Since $\mathbf{y}$ was arbitrary, we obtain, by the characterization of the covering radius of a linear code in terms of its parity check matrix (see, e.g. [2, Th. 2.1.9]), that $R(C) \leq p$.

Now let us consider the minimum distance of C. The first eight columns of $\mathbf{H}$, corresponding to the first triple in M, consist of the four rows

$$\begin{array}{cccccccc} 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ a & a & a & a & a & a & a & a \end{array}$$

(where $a = 1$ or 0 according to the membership of the triple to M_1), together with all-zero rows — in a certain order, specified by the triple. Anyway, the sum of the first, fifth, sixth and seventh columns is zero, and therefore C has minimum distance at most four. We know that every such code is normal, i.e., if its covering radius is R, then its minimum norm is at most $2R + 1$ (see, e.g., [2, Th. 4.2.2]). Here, since C has covering radius at most p, it has minimum norm at most $2p + 1 (= w)$.

Conversely, assume that C has minimum norm at most $2p + 1$. Then its covering radius is at most p, i.e., for all $\mathbf{y} \in \mathbb{F}_2^{3p+|M_1|}$, there exists $\mathbf{x} \in \mathbb{F}_2^{8|M|}$, such that $\mathbf{H}\mathbf{x}^T = \mathbf{y}$ and $wt(\mathbf{x}) \leq p$. For any set $S_1 \subseteq M_1$, let $\mathbf{y}$ be the vector with 1's in the first $3p$ coordinates and with 1's in those coordinates among the last $|M_1|$ that correspond to triples in S_1. Then $\mathbf{H}_1\mathbf{x}^T$ is the all-one vector of

length $3p$ and is the sum of at most p columns of $\mathbf{H}_1$, each column containing at most three 1's. Thus, $\mathbf{H}_1\mathbf{x}^T$ is the sum of exactly p columns of $\mathbf{H}_1$, each column containing exactly three 1's. So $\mathbf{H}\mathbf{x}^T$ is the sum of p columns $\mathbf{H}^{t(0)}$ of $\mathbf{H}$ and these p columns select a matching S. Since this sum has 1's in just those positions among the last $|M_1|$ that correspond to triples in S_1, it follows that the triples in M_1 that are contained in S are just those in S_1. Therefore $S = S_1 \cup S_2$, where $S_2 \subseteq M_2$. Thus the $\forall\exists$-3-dimensional matching property holds.

This, together with NORM2$\in \Pi_2$, shows that NORM2 is Π_2-complete. □

References

[1] J. P. Barthélemy, G. Cohen, A. Lobstein: Algorithmic Complexity and Communication Problems. UCL Press, London, 1996.

[2] G. Cohen, I. Honkala, S. Litsyn, A. Lobstein: Covering Codes. Elsevier, Amsterdam, 1997.

[3] G. Cohen, S. Litsyn, A. Lobstein, H. F. Mattson, Jr.: Covering radius 1985-1994, *Applicable Algebra in Engineering, Communication and Computing*, Special Issue, vol. 8, No. 3, 67 pp., 1997.

[4] M. Frances, A. Litman: On covering problems of codes, *Theory of Computing Systems*, vol. 30, No. 2, pp. 113–119, 1997.

[5] M. R. Garey, D. S. Johnson: Computers and Intractability, a Guide to the Theory of NP-Completeness. Freeman, New York, 1979.

[6] R. L. Graham, N. J. A. Sloane: On the covering radius of codes, *IEEE Trans. Inform. Th.*, vol. 31, pp. 385–401, 1985.

[7] A. McLoughlin: The complexity of computing the covering radius of a code, *IEEE Trans. Inform. Th.*, vol. 30, pp. 800–804, 1984.

Perfect binary codes components *

F.I. Solov'eva
Sobolev Institute of Mathematics
Koptyug pr.4, Novosibirsk 630090, Russia
sol@math.nsc.ru

Abstract

Special components of perfect binary codes are investigated. We call such components i-components. The existence of maximal cardinality nonisomorphic i-components of different perfect codes of length n for all $n = 2^k - 1, k > 3$, is proved. A class of perfect codes of length n with nonextremal cardinality i-components is constructed for all admissible $n > 7$.

1 Introduction

Let C be a single-error correcting perfect binary code with distance 3

(briefly a perfect code) and M be a subset of C. Exchanging the bit in the i'th coordinate of all vectors of M with the opposite bit we obtain a new set, denoted by $M \oplus i$. We identify with i the vector having an one in the i'th coordinate and zeroes elsewhere. If $C' = (C \setminus M) \cup (M \oplus i)$ is a perfect code, we call the set M an *i-component* of the code C and say that C' is obtained from C by *switching* of an i-component M. An i-component is *minimal* if it cannot be subdivided into smaller i-components. We omit the word minimal because we consider below only minimal i-components.

It is known [3, 4] that upper and lower bounds on the number m of i-components of an arbitrary perfect code of length n, $n = 2^k - 1$, are given by

$$2 \leq m \leq 2^{\frac{n}{2}}/(n). \tag{1}$$

Both of these bounds can be achieved, see [1, 3, 4]. The cardinality of the minimal i-components can vary from $2^{(n-1)/2}$ to $2^{n-1}/(n)$. A perfect code of length n with i-components of different structures and cardinalities was presented in [5] for all admissible $n > 7$.

We prove that there exist maximal cardinality nonisomorphic i-components of different perfect codes of length n for all $n = 2^k - 1, k > 3$. We construct a class of nonextremal cardinality i-components of perfect codes of length n for every such n. Constructions are given by using simultaneously the switching [1]

and the concatenation [2] constructions of perfect codes.

*This research was supported by the Russian Foundation for Basic Research under grant 97-01-01104

2 Construction

We further identify a vector $x = (x_1, \ldots, x_n) \in E^n$ with its block presentation $(i_1, \ldots, i_k)$ of all those positions of coordinates of vector x, which equaled to 1.

Consider the Hamming code H^n of length n given by Vasil'ev construction [1]

$$H^n = \{(\alpha, \alpha \oplus \beta, |\alpha|) : \alpha \in E^{(n-1)/2}, \beta \in H^{(n-1)/2}\}, \tag{2}$$

where $|\alpha| = \alpha_1 \cdots \alpha_{(n-1)/2} \pmod 2$ for $\alpha = (\alpha_1, \ldots, \alpha_{(n-1)/2})$ and $H^{(n-1)/2}$ be the Hamming code of length $(n-1)/2 = 2^k - 1, k \geq 2$.

According to Glagolev's lemma, see in [6], for any binary linear code of length n, cardinality k and distance d there exists a binary linear code with the same parameters and a basis among codewords of weight d. Hamming codes are unique up to equivalence, therefore there exists a basis of the Hamming code among codewords of weight 3.

We construct such a basis for the Hamming code by induction. If $k = 2$ the Hamming code H^3 is generated by codeword $(1, 2, 3)$. Let $V_n = \{(i, j, k)\}$ be a basis of the Hamming code H^n, $|V_n| = n - log(n1)$. Let $V_{2n1} = V_n \cup \{(s, n, 2n) : s = 1, \ldots, n\}$. According to (2) the vector $(s, n, 2n)$ belongs to H^{2n}, $s = 1, \ldots, n$, and $V_n \subset H^{2n}$. Therefore $|V_{2n1}| = 2n - log(n1)$ and the set V_{2n1} is a basis of weight 3 codewords of the Hamming code H^{2n}. We have then,

Proposition 1. *There exists a basis of weight 3 codewords of the Hamming code of length n for any $n = 2^k - 1, k \geq 2$.*

Proposition 2. *There exist Hamming codes H_1^n and H_2^n of length n, $n = 2^k - 1, k \geq 3$, with basis intersected exactly by one codeword.*

Proposition 3. *(see [7]) There exist Hamming codes H_3^n and H_4^n of length n, $n = 2^k - 1, k \geq 3$, with nonintersected basis.*

Consider now a concatenation construction of perfect codes given in [2]. Let $C_0, C_1, \ldots, C_n$ and $C'_0, C'_1, \ldots, C'_n$ be any partitions of the vector space E^n into perfect codes (constructions of nontrivial partitions of E^n can be found in [2]). Let φ be a permutation on the set $\{0, 1, \ldots, n\}$. Then the code

$$C^{2n} = \{(x, y, |y|) : x \in C_i, y \in C'_{\varphi(i)}, i = 0, 1, \ldots, n\} \tag{3}$$

is a perfect code of length $2n$.

Let H_1^n and H_2^n be Hamming codes given above and φ be an identity permutation. Using for the concatenation construction (3) partitions of E^n into cosets $H_1^n \oplus i$ and $H_2^n \oplus i$ of Hamming codes H_1^n and H_2^n respectively, $i = 0, 1, \ldots, n$, we obtain the perfect code, denote it by K^{2n}. In our case H_1^n is not equal to H_2^n then the code K^{2n} is not a linear code, see [8, 2].

Using for the same construction (3) cosets $H_3^n \oplus i$ and $H_4^n \oplus i$ of Hamming codes H_3^n and H_4^n respectively, $i = 0, 1, \ldots, n$, we obtain another nonlinear perfect code of length $2n$. Denote it by R^{2n}, see [7].

Theorem 1. *Every perfect code K^{2n} and R^{2n} is partitioned into two $(2n)$-components of maximal cardinality for $n = 2^k - 1, k \geq 3$.*

A mapping ψ between two perfect codes C and C' such that $d(x,y) = d(\psi(x), \psi(y))$ for all codewords $x, y \in C$ is called an *isometry* from C to C'.

Comparing structures of perfect codes K^{2n} and R^{2n} we prove Theorem 2.

Theorem 2. *There exist maximal cardinality nonisometric i-components of different perfect codes of length n for all $n = 2^k - 1, k > 3$.*

¿From Theorem 2 one can obtain Theorem 3.

Theorem 3. *There exist maximal cardinality nonisomophic i-components of different perfect codes of length n for all $n = 2^k - 1, k > 3$.*

Let $H_5^{(n-1)/2} \cap H_6^{(n-1)/2} = H^t$, where $H_5^{(n-1)/2}$ and $H_6^{(n-1)/2}$ are Hamming codes of length $(n-1)/2, n > 7$, and H^t is the Hamming code of length t, $t = 2^s - 1, s = 2, \ldots, log(n1)/2 - 1$. Using the construction (3) for codes $H_5^{(n-1)/2}$ and $H_6^{(n-1)/2}$ we construct a class of perfect codes of length n with nonextremal cardinality i-components for all $n = 2^k - 1, k > 3$.

Theorem 4. *There exists a perfect code of length n with minimal i-components of cardinality $(t1)2^{n-t}/(n1)$ for every $n = 2^k - 1, k > 3$ and $t = 2^s - 1$, where $s = 2, \ldots, log(n1)/2 - 1$.*

However, the question of enumerating all possible sizes of minimal i-components of perfect binary codes remains open.

References

[1] *Vasil'ev Y.L.,* On nongroup close-packed codes, Problems of Cybernetics 8 (1962) 375-378 (in Russian).

[2] *Solov'eva F.I.,* On binary nongroup codes, Methody Discretnogo Analiza 37 (1981) 65-76 (in Russian).

[3] *Solov'eva F.I.,* Factorization of code-generating disjunctive normal forms, Methody Discretnogo Analiza 47 (1988) 66-88. (in Russian).

[4] *Solov'eva F.I.,* Exact bounds on the connectivity of code-generating disjunctive normal forms, Inst. Math. of the Siberian Branch of Acad. of Sciences USSR, Preprint 10 (1990) 15 (in Russian).

[5] *Avgustinovich S.V., Solov'eva F.I.,* On projections of perfect binary codes, Proc. Seventh Joint Swedish-Russian Workshop on Information Theory, St.-Petersburg, Russia. June (1995) 25-26.

[6] *Kurljandchik Ja.M.,* On logariphmic asimptotics length of maximal cycle of spread $r > 2$, Methody Dickretnogo Analiza 19 (1971) 48-55.

[7] *Vasil'ev Yu.L., Solov'eva F.I.,* Codegenerating factorization on n-dimensional unite cube and perfect binary codes, Problems of Information Transmission 33 (1) (1997) 64-74.

[8] *Heden O.,* A new construction of group and nongroup perfect codes, Inform. and Control 34 (4) (1977) 314-323.

Permutation groups of error-correcting codes

Nicolas Sendrier*, Gintaras Skersys†

Introduction

The problem of finding the automorphism (or permutation) group of a code (and an isomorphism (or permutation) between two equivalent codes) is difficult (see [PR97]). An algorithm was developed by J. Leon to address this problem (see [Leo82], [Leo84] [Leo91], [Leo92], [Leo97]). However, in order to find the automorphism group $Aut(C)$ of a linear code C, this algorithm must first compute a set W of vectors invariant under the action of $Aut(C)$ which is "reasonably small" but which "contains enough structure" ([Leo82], [Leo92]). Often it will be the set of minimal weight vectors of C or of its dual. Therefore the algorithm of Leon is limited to the codes for which one can "easily" find such a set W (for example, in binary case to the codes of length up to 50–60 or dimension up to 25–30 — with few exceptions).

We present here an algorithm for finding the permutation group for some other codes. Our algorithm is limited by the size of the hull of a code (the intersection of a code with its dual), for we have to calculate its weight enumerator. The hull of almost all linear code is small (see the note in the section 2.4), therefore, theoretically, our algorithm can find the permutation group of almost all linear codes. In practice we can deal with codes of length as high as a few thousands as long as the hull has dimension less than about 25. Unfortunately, in some important families of linear codes (for instance cyclic codes) many codes have a big hull, so our algorithm is not applicable to them.

We begin by several **notations**.

1. For any finite set E, we denote $|E|$ the cardinality of E.
2. We denote by $\mathbf{F}_q$ the finite field with q elements.
3. We denote by Ω the finite set used to index the coordinates of a given code C. We let n denote $|\Omega|$ (the length of C).
4. For any integer $k > 0$, we denote $I_k = \{1, \ldots, k\}$.

*Projet CODES, INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 Le Chesnay CEDEX, FRANCE. E-mail: Nicolas.Sendrier@inria.fr

†LACO, Département de Mathématiques, Faculté des Sciences de Limoges, 123 avenue Albert Thomas, 87060 LIMOGES cedex, FRANCE. E-mail: Gintaras.Skersys@unilim.fr

5. $Sym(\Omega)$ denotes the symmetric group on Ω.

In this paper codes will be over the finite field $\mathbf{F}_q$.

Definition 1

- *We call two codes C and C'* equivalent *if they are obtained one from another by the permutation of the support, i.e. if exists such a permutation $\sigma \in Sym(\Omega)$ that $C' = C^\sigma$, where $C^\sigma = \{(x_{i^{\sigma^{-1}}})_{i\in\Omega} \mid (x_i)_{i\in\Omega} \in C\}$.*
- *The* permutation group *of a code C, denoted by $Perm(C)$, is the subgroup of all the elements σ of $Sym(\Omega)$ such that $C^\sigma = C$.*

The most general form of Leon's algorithm (see [Leo91], [Leo97]) computes a subset K of $Sym(\Omega)$ described by a *property* $\mathcal{P}$ — i.e. $g \mapsto \mathcal{P}(g)$ is a boolean-valued function on $Sym(\Omega)$ such that $\mathcal{P}(g)$ is true if and only if $g \in K$ — by use of the so-called elementary refinement processes, related to this property (see Def.4).

The main difficulty lies precisely in the choice of these refinements, and this is where our algorithm differs.

For a given code C we will denote $\mathcal{A}_C$ the property such that $\mathcal{A}_C(g)$ holds exactly when g is in the permutation group of C.

1 Partitions, refinements and backtrack search

In this section we will define the elementary $\mathcal{P}$-refinement process and we will explain its significance in the backtrack search method used in Leon's algorithm.

1.1 Partitions

We must start with a few notations and definitions of the partitions ([McK78], [Leo91]).

Definition 2 *A* partition *of Ω is a collection $\mathbf{\Pi}$ of disjoint non-empty subsets of Ω whose union is Ω. The elements of $\mathbf{\Pi}$ are called its* cells. *An* ordered partition *of Ω is a sequence $(\Pi_1, \Pi_2, \ldots, \Pi_k)$ for which $\{\Pi_1, \Pi_2, \ldots, \Pi_k\}$ is a partition. The set of all ordered partitions of Ω will be denoted by $Partn(\Omega)$.*

If $\mathbf{\Pi}$ is a partition (ordered or not), the number of cells of $\mathbf{\Pi}$ is denoted by $|\mathbf{\Pi}|$. $\mathbf{\Pi}$ is called discrete *if $|\mathbf{\Pi}| = n$. If $|\mathbf{\Pi}|+1 \leq i \leq n$, Π_i will denote the empty set.*

We will need to compare the ordered partitions. For this we define an ordering.

Definition 3 *If $\mathbf{\Pi} = (\Pi_1, \Pi_2, \ldots, \Pi_k)$ and $\mathbf{\Sigma} = (\Sigma_1, \Sigma_2, \ldots, \Sigma_m)$ are ordered partitions, we define $\mathbf{\Pi} \leq \mathbf{\Sigma}$ to mean*

1. $|\Pi| \geq |\Sigma|$,

2. $\Pi_i \subseteq \Sigma_i$ *for* $i \leq |\Sigma|$, *and*

3. Π_i, $i > |\Sigma|$, *is contained in some cell of* Σ.

If $\Pi \leq \Sigma$ *and* $\Pi \neq \Sigma$, *we write* $\Pi < \Sigma$ *and say that* Π *is* finer *than* Σ.

1.2 Refinements

If G is a permutation group, then $G_{\mathcal{P}}$ will denote $\{g \in G \mid \mathcal{P}(g)\}$.

The purpose of Leon's algorithm is to find a strong generating set of $Sym(\Omega)_{\mathcal{P}}$ — a strong generating set is a particular set of generators of a group of permutations, introduced by Sims [Sim71a], [Sim71b]. This set will be obtained by means of the elementary $\mathcal{P}$-refinement processes and by a programming technique called backtrack search.

In practice, the solutions to our problem can be represented by a tree. The nodes of this tree are labeled by pairs of ordered partitions. The root of the tree is labeled by $((\Omega),(\Omega))$, where (Ω) is the ordered partition with one cell. As we progress in the tree the partitions become finer and finer, in particular by use of the elementary $\mathcal{P}$-refinement processes, until we reach the leaves labeled by (Π, Π^g) with Π (and thus Π^g) discrete and $g \in Sym(\Omega)_{\mathcal{P}}$ (i.e. there is exactly one leave for each element of $Sym(\Omega)_{\mathcal{P}}$, and we may recover this element from the pair of partitions (Π, Π^g)).

The tree is searched by the backtrack search method, and pruned by the properties of the strong generating set.

The searched tree may contain useless nodes, depending on the "power" of the elementary $\mathcal{P}$-refinement processes.

We give the formal definition of an elementary $\mathcal{P}$-refinement process.

Definition 4 *[Leo91, Def.8] If* $\mathcal{P}$ *is a property, an* elementary $\mathcal{P}$-refinement process $\mathfrak{E}$ *on* G *is a pair* $(\mathfrak{E}_{\mathsf{L}}, \mathfrak{E}_{\mathsf{R}})$ *of mappings of* $Partn(\Omega)$ *into* $Partn(\Omega)$ *such that the following hold for every* Π *and* Σ *in* $Partn(\Omega)$*:*

(a) $\mathfrak{E}_{\mathsf{L}}(\Pi) \leq \Pi$ *and* $\mathfrak{E}_{\mathsf{R}}(\Pi) \leq \Pi$.

(b) $|\mathfrak{E}_{\mathsf{L}}(\Pi)| \leq |\Pi| + 1$ *and* $|\mathfrak{E}_{\mathsf{R}}(\Pi)| \leq |\Pi| + 1$.

(c) If $g \in G_{\mathcal{P}}$ *and if* $\Pi^g = \Sigma$, *then* $\mathfrak{E}_{\mathsf{L}}(\Pi)^g = \mathfrak{E}_{\mathsf{R}}(\Sigma)$.

Note [Leo91, p.542] The subscripts "L" and "R" are formal symbols chosen to suggest left and right, corresponding to the appearance of $\mathfrak{E}_{\mathsf{L}}$ and $\mathfrak{E}_{\mathsf{R}}$ on the left side and the right side, respectively, of the equation in (c) above. Note that (a) and (b) imply that either $\mathfrak{E}_{\mathsf{z}}(\Pi) = \Pi$ or $|\mathfrak{E}_{\mathsf{z}}(\Pi)| = |\Pi| + 1$ for $\mathsf{z} \in \{\mathsf{L}, \mathsf{R}\}$; that is, each component of an elementary $\mathcal{P}$-refinement process either leaves Π unchanged or splits exactly one of its cells.

An elementary $\mathcal{P}$-refinement process is a pair of mappings; in each node of the search tree, $\mathfrak{E}_\mathsf{L}$ is applied to the left partition and $\mathfrak{E}_\mathsf{R}$ to the right partition. But in our case (as we try to calculate the permutation group) it is simpler, any elementary $\mathcal{P}$-refinement process may be expressed as a single mapping of $Partn(\Omega)$ into itself.

Definition 5 *[Leo91, Def.10] An elementary $\mathcal{P}$-refinement process $\mathfrak{E} = (\mathfrak{E}_\mathsf{L}, \mathfrak{E}_\mathsf{R})$ is* symmetric *if $\mathfrak{E}_\mathsf{L} = \mathfrak{E}_\mathsf{R}$.*

One result from [Leo91, Lem.6] :

Lemma 1 *If $G_\mathcal{P}$ is a subgroup of G, then any elementary $\mathcal{P}$-refinement process is symmetric.*

In our case $G = Sym(\Omega)$, $\mathcal{P} = \mathcal{A}_C$ for a given code C and $G_\mathcal{P} = Perm(C)$. Thus, any elementary $\mathcal{A}_C$-refinement process is symmetric.

Definition 6 *[Leo91, Def.11] If $\mathfrak{E} = (\mathfrak{E}_\mathsf{L}, \mathfrak{E}_\mathsf{R})$ is a symmetric elementary $\mathcal{P}$-refinement process, we will write $\mathfrak{E}(\Pi)$ in place of $\mathfrak{E}_\mathsf{z}(\Pi)$, $\mathsf{z} = \mathsf{L}$ or R. (Essentially we are treating $\mathfrak{E}$ as a mapping of $Partn(\Omega)$ into itself, rather than a 2-tuple whose components are such mappings.)*

2 Signatures

2.1 Definitions

We begin by reminding some definitions.

Definition 7 *Let C be a code and let J be a subset of Ω. The code C* punctured *in J, denoted C_J, consists of all codewords of C where the coordinates indexed by J are replaced by zeroes, i.e.*

$$C_J = \{(x'_i)_{i\in\Omega} \mid (x_i)_{i\in\Omega} \in C, \text{ where } x'_i \text{ denotes } x_i, \text{ if } i \notin J, \text{ and } 0 \text{ else } \}.$$

Note that this definition is different from usual definition of punctured code (for example, see [MS78, Ch.1.§9.]). In our case C_J remains in the same space $\mathbf{F}_q^n$ as C.

We will denote $C_i \stackrel{\text{def}}{=} C_{\{i\}}$.

Definition 8 *[Sen97a] A* signature *S over a set F maps a code C and an element i of Ω into an element of F and is such that for all permutations σ on Ω, $S(C, i) = S(C^\sigma, i^\sigma)$.*

Signature associate a given code and each coordinate of its support with an element of F. This allows to differentiate the coordinates of a given code. See [Sen97a] for how to differentiate further the coordinates and for other properties of signatures.

2.2 Generalized signature

Let C be a code of length n, let S be a signature over a set E, let Π be an ordered partition of Ω, let L be a subset of I_n and let $i \in \Omega$.

We define

$$\mathsf{T}_{S,L}(\Pi, C, i) = S(C_{\bigcup_{j \in L} \Pi_j}, i). \tag{1}$$

We remark one property of $\mathsf{T}_{S,L}$.

Lemma 2

$$\mathsf{T}_{S,L}(\Pi^\sigma, C^\sigma, i^\sigma) = \mathsf{T}_{S,L}(\Pi, C, i) \text{ for all } \sigma \in Sym(\Omega).$$

Mappings that satisfy the lemma 2 will be called *generalized signatures.*

Definition 9 *Let C be a code, let Π be an ordered partition of Ω, let E be a set and let $i \in \Omega$. A mapping* R *which associates Π,C and i to an element of E is called* generalized signature *if*

$$\mathsf{R}(\Pi^\sigma, C^\sigma, i^\sigma) = \mathsf{R}(\Pi, C, i) \text{ for all } \sigma \in Sym(\Omega).$$

Like a signature, a generalized signature also differentiates the coordinates of a given code C, but it depends yet on a third argument — partition.

When we have some generalized signatures, we can construct other generalized signatures in the following manner.

Proposition 1 *Let* P *and* Q *be generalized signatures. Then* $\mathsf{P}^\perp$*, defined by* $\mathsf{P}^\perp(\Pi, C, i) = \mathsf{P}(\Pi, C^\perp, i)$*, and* $\mathsf{P} \times \mathsf{Q}$*, defined by* $\mathsf{P} \times \mathsf{Q}(\Pi, C, i) = (\mathsf{P}(\Pi, C, i), \mathsf{Q}(\Pi, C, i))$*, are generalized signatures.*

Corollary 1 *If* R *is a generalized signature, then* $\bar{\mathsf{R}} \stackrel{\text{def}}{=} \mathsf{R} \times \mathsf{R}^\perp$ *is also a generalized signature.*

We hope that $\bar{\mathsf{R}}$ will be able to better differentiate the coordinates of a given code than R.

2.3 Signature-based refinements

By means of the generalized signature we will define our elementary $\mathcal{A}_C$-refinement processes. But before we give one useful notation (from [Leo91, Def.15] and [Leo97, Def.6]). To describe the action of a symmetric elementary $\mathcal{P}$-refinement process on a partition, it suffices to precise which cell is split and which elements are moved to the new cell.

If $\Phi = (\Phi_1, \ldots, \Phi_k) \in Partn(\Omega)$, $i \in I_n$ and $\Gamma \subset \Omega$, then $\mathcal{E}_{i,\Gamma}(\Phi)$ is defined by

$$\mathcal{E}_{i,\Gamma}(\Phi) = \begin{cases} (\Phi_1, \ldots, \Phi_i \setminus \Gamma, \ldots, \Phi_k, \Phi_i \cap \Gamma) & \text{provided } i \leq k \text{ and } \emptyset \subsetneq \Phi_i \cap \Gamma \subsetneq \Phi_i, \\ \Phi & \text{otherwise.} \end{cases}$$

Now we define our elementary $\mathcal{A}_C$-refinement process.

Proposition 2 *Let C be a code of length n, let $\Pi \in Partn(\Omega)$, let $i \in I_n$, let* R *be a generalized signature over a set E and let $e \in E$. The mapping $Q_{C,\mathsf{R},i,e}$ of $Partn(\Omega)$ into $Partn(\Omega)$, defined by*

$$Q_{C,\mathsf{R},i,e}(\Pi) = \mathcal{E}_{i,\Phi}(\Pi), \text{ where } \Phi = \{j \in \Omega \mid \mathsf{R}(\Pi, C, j) = e\},$$

is an elementary $\mathcal{A}_C$-refinement process.

2.4 Hull and related signature

We saw that one may construct an elementary $\mathcal{A}_C$-refinement process from a generalized signature, and a generalized signature from a signature (see (1)). Now we will present one good signature that we use in our algorithm.

Definition 10 *The* hull *of a linear code C is defined to be its intersection with its dual. We will denote $\mathcal{H}(C) = C \cap C^{\perp}$.*

Let $\mathcal{W}(C)$ denote the weight enumerator of C. In [Sen97a] Sendrier propose the signature

$$\mathbf{S} : (C, i) \mapsto \left(\mathcal{W}(\mathcal{H}(C_i)), \mathcal{W}(\mathcal{H}((C^{\perp})_i))\right) \tag{2}$$

(we puncture C and $C^{\perp}$ in i and compute the weight enumerators of the hulls of punctured codes).

Note [Sen97b] The average dimension of the hull of a q-ary $[n, k]$ code tends to a small positive constant when the size of the code goes to infinity. For example, for binary codes this constant is equal to 0.7645, for ternary 0.4041, for quaternary 0.2794, etc.

So the weight enumerator of the hull and the signature **S** are (rather) easy to compute.

Let's denote $\mathbf{Q}_{C,L,i,e} = Q_{C,\overline{\mathbf{T}}_{\mathbf{S},L},i,e}$. Therefore, we use Leon's algorithm with the set $\mathbf{\Omega}_C$ of elementary $\mathcal{A}_C$-refinement processes, where $\mathbf{\Omega}_C$ is defined by $\mathbf{\Omega}_C = \{\mathbf{Q}_{C,L,i,e} \mid L \subset I_n,\ i \in I_n,\ e \in W_n^4\}$ (W_n denotes the set of weight enumerators of all codes of length n. Note that W_n is a finite set).

2.5 Final notes

And now two final notes.

First, we speak about the permutation group of a code, and not about the automorphism group. If we define codes C and C' to be *equivalent (in general sense)* [MS78, p.40] when there exist a permutation σ on Ω and a sequence $(\pi_i)_{i\in\Omega}$ of n permutations on $\mathbf{F}_q$ such that $C' = \phi(C)$ where $\phi : (x_i)_{i\in\Omega} \mapsto ((x_{i\sigma^{-1}})^{\pi_{i\sigma^{-1}}})_{i\in\Omega}$, then the *automorphism group* of C consists of all such functions

ϕ with $C = \phi(C)$. To prove that $\mathbf{S}$, defined in (2), is a signature, we use the fact that if C and C' are equivalent, then $\mathcal{H}(C)$ and $\mathcal{H}(C')$ are also equivalent. But in general case this does not hold, i.e. if C and C' are equivalent (in general sense), then $\mathcal{H}(C)$ and $\mathcal{H}(C')$ are not necessarily equivalent (in general sense). Thus in this case we cannot define a signature using the hull, consequently we cannot define an elementary $\mathcal{A}_C$-refinement process $\mathbf{Q}_{C,L,i,e}$ and use Leon's algorithm. That's why we can use the hull when we compute the permutation group of a code and not the automorphism group.

Second, we can prove that if C and C' are two linear codes, then $Q_{C,C',\mathsf{R},i,e} = (Q_{C,\mathsf{R},i,e}, Q_{C',\mathsf{R},i,e})$ is an elementary $\mathcal{A}_{C,C'}$-refinement process, where $\mathcal{A}_{C,C'}$ is a property such that $\mathcal{A}_{C,C'}(g)$ holds exactly when $C' = C^g$. Thus we can find a permutation between two equivalent codes or we can prove that two codes are not equivalent. The procedure is analogous to the one described above, and uses a version of Leon's algorithm described in [Leo91, Fig.8].

3 Comparison with Leon's algorithm

What do we gain in comparison with the Leon's original algorithm ? Leon's algorithm for finding the automorphism group of a given code C uses the elementary refinement processes for finding the automorphism group of a matrix. It takes a set W of vectors invariant under the action of $Aut(C)$ and treats it as a matrix. W may be the set of all codewords of C, $C^\perp$ or $\mathcal{H}(C)$, the set of minimum weight (or constant weight) codewords of C, $C^\perp$ or $\mathcal{H}(C)$, the union of some of these sets, etc. But W must be "reasonably small" (otherwise W is too large as a matrix) and must "represent well" the code C, i.e. the automorphism group of W must not be much bigger than $Aut(C)$ (otherwise the search tree to process, defined by W, will be much larger than the solution tree).

In practice, when such a set W is available, Leon's algorithm is a little faster than ours. But there exist many cases when such W is not available (thus Leon's algorithm cannot work), and our algorithm work. For instance, when C and $C^\perp$ are too large to calculate the set of minimum weight codewords, and $\mathcal{H}(C)$ is too small. And it occurs very often, e.g. random linear codes have, on average, a small hull (see note in the section 2.4), all cyclic codes of length $n = q^m + 1$, $m = 1, 2, \ldots$ have the hull reduced to $\{0\}$, etc.

4 Running time

Binary case of our algorithm was fully implemented in language C. We present an example of the running time of our algorithm. We try all binary non degenerate narrow sense BCH codes of length $n = 2^m + 1$, $m = 4, 5, \ldots, 10, 11$. In table 1, we give for all m the number of such codes, the minimal, average and maximal

m	n	Number of codes	Running time (in CPU sec)		
			minimal	average	maximal
4	17	1	0.016	0.016	0.016
5	33	2	0.027	0.028	0.029
6	65	4	0.055	0.060	0.064
7	129	8	0.121	0.150	0.190
8	257	15	0.253	0.400	0.586
9	513	28	0.757	1.468	2.170
10	1025	50	2.619	5.795	10.152
11	2049	92	11.878	20.894	31.446

Table 1: The running time

running time of our algorithm on DEC Alpha EV56 500/400 (400 MHz).

References

[Leo82] Jeffrey S. Leon. *Computing Automorphism Groups of Error-Correcting Codes.* IEEE Transactions on Information Theory, Vol. IT–28, No. 3, May 1982, pp. 496–511.

[Leo84] Jeffrey S. Leon. *Computing automorphism groups of combinatorial objects.* In: Computational group theory, 321–335, Academic Press, London-New York, 1984.

[Leo91] Jeffrey S. Leon. *Permutation Group Algorithms Based on Partitions,I: Theory and Algorithms.* J.Symbolic Computation (1991) 12, 533–583.

[Leo92] Jeffrey S. Leon. *Partition backtrack programs: User's manual.* http://www.math.uic.edu/~leon/

[Leo97] Jeffrey S. Leon. *Partitions, Refinements, and Permutation Group Computation.* DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Vol. 28, 1997; 123–158.

[McK78] B.D. McKay. *Computing automorphisms and canonical labellings of graphs.* Lecture Notes in Math. 686, 223–232, 1978.

[MS78] F.J. MacWilliams, N.J.A.Sloane. *The Theory of Error-Correcting Codes.* North-Holland, 1978.

[PR97] E. Petrank and R.M.Roth. *Is code equivalence easy to decide?* IEEE Transactions on Information Theory, 43(5):1602–1604, September 1997.

[Sen97a] Nicolas Sendrier. *Finding the permutation between equivalent binary codes.* In: IEEE Conference, ISIT'97, Ulm, Germany, June 1997.

[Sen97b] Nicolas Sendrier. *On the dimension of the hull.* SIAM Journal on Applied Mathematics, 10(2):282–293, May 1997.

[Sim71a] C.C. Sims. *Determining the conjugacy classes of a permutation group.* Proc. of the Symposium on Computers in Algebra and Number Theory, Amer. Math. Soc., New York, 1971, pp. 191–195.

[Sim71b] C.C. Sims. *Computation with permutation groups.* Proc. of the Second Symposium on Symbolic and Algebraic Manipulation, Assoc. for Computing Mach., New York, 1971, pp. 23–28.

Recognition of a Binary Linear Code as a Vector-Subspace

Antoine Valembois *†‡

1 Introduction

This paper deals with the following situation: someone intercepts a noisy binary signal and wants to understand it. He knows the message has been encoded with a linear code. He also knows the length and the dimension of this code. He even manages to be synchronized with the (erroneous) codewords. But he just doesn't know which code has been used and wants to recognize the most likely one (given the signal), not as an element of a well studied family of decodable codes, but just as a vector subspace.

There are not many publications dealing with this problem. One can find sketches of solutions for the convolutional codes [Pla96, Fil97] and try to adapt it to the block codes. But this paper may fill the gap in the literature dealing with this specific problem.

This is only an abstract of the full paper [Val99] with complete proofs, which will be soon availiable.

First we will formalize this problem and call it the "Maximum Likelihood Code Recognition" problem. It will lead us to another one, the "Rank Reduction" problem, which we prove to be NP-complete. Then we give a general solution to this "Rank Reduction" problem by designing an algorithm that will retrieve the dual code. We then evaluate the computation cost of the latter. Finally we will give an example of implementation and the limits above which the problem becomes intractable.

2 Presentation of the problem

2.1 The Maximum Likelihood Code Recognition Problem

Let $(X_i)_{i=1,\dots,N}$ be the N received words of length n.

*INRIA, Projet CODES, Domaine de Voluceau, Rocquencourt, BP 105, 78153 Le Chesnay Cedex, France (e-mail: antoine.valembois@inria.fr)

†Thomson-CSF Communications, 66 Rue du Fossé Blanc, 92231 Gennevilliers

‡This work was partially supported by the *DGA*

For every linear code C of length n, let $\Pr(C|X)$ denote the probability that C is the linear code from which the received words $(X_i)_{i=1,\dots,N}$ have stemmed from.

We will calculate this probability under the assumptions that:

- All the vector-subspaces are *a priori* equally likely to be used,
- all the codewords are *a priori* equally likely to be transmitted,
- the binary channel is memoryless, symmetric and of bit error rate (b.e.r.) $\tau < 1/2$.

Let $w_H(.)$ denote the Hamming weight function, $d(.,.)$ the Hamming distance and for every code C, $d(x,C) = \min_{c \in C}(d(x,c))$ the Hamming distance of a word x to the code C, we prove [Val99] that:

$$\begin{aligned} \Pr(C|X) &= \frac{\Pr(C)}{\Pr(X)} 2^{-dim(C)N} (1-\tau)^{nN} \prod_{i=1}^{N} \left(\sum_{c \in C} \left(\frac{\tau}{1-\tau} \right)^{d(X_i,c)} \right) \\ &\approx \frac{\Pr(C)}{\Pr(X)} 2^{-dim(C)N} (1-\tau)^{nN} \left(\frac{\tau}{1-\tau} \right)^{\sum_{i=1}^{N} d(X_i,C)} \end{aligned}$$

(the term $2^{-dim(C)N}$ is the probability related to the messages before encoding).

The problem of Maximum Likelihood Code Recognition (MLCR) consists in finding the vector-subspace C for which the probability $\Pr(C|X)$ is maximum.

Thus, when the above approximation is valid (which is the case for "normal" values of τ), C has to be such that $\sum_{i=1}^{N} d(X_i,C) + \frac{N}{\log_2(\tau^{-1}-1)} dim(C)$ is minimum.

If the dimension $dim(C) = k$ is known, we will then have to reduce to k the rank of the matrix X, by changing as few of its coefficients as possible. This is our main problem which is described specifically in the following section.

2.2 Rank Reduction

Let $\mathbf{F}_2$ be the field with two elements, and let $\mathcal{M}_{r,c}(\mathbf{F}_2)$ denote the set of $r \times c$ matrices over $\mathbf{F}_2$.

The Hamming weight $w_H(M)$ of a matrix M will denote the number of its coefficients equal to 1. Its rank will be written $rk(M)$.

> *Let N and n be two positive integers, and X a matrix in $\mathcal{M}_{N,n}(\mathbf{F}_2)$. Let $k \leq rk(X)$ be another positive integer. The Rank Reduction problem (RR) may be formulated this way:*
>
> **RR**(X,k): *Find a matrix $E^\star$ of minimum weight in $\mathcal{M}_{N,n}(\mathbf{F}_2)$ such that $rk(X+E^\star) = k$.*
>
> *if ω is an integer, the associated decision problem (DRR) is:*
>
> **DRR**(X,k,ω):*is there a matrix $E^\star \in \mathcal{M}_{N,n}(\mathbf{F}_2)$ such that* $\begin{cases} rk(X+E^\star)=k \\ w_H(E^\star) \leq \omega \end{cases}$?

2.3 Complexity

The DRR problem is clearly NP since, if the answer to DRR(X, k, ω) is true, given a solution $E^\star$ (as a certificate), it requires only polynomial time to check that $rk(X + E^\star) = k$ and that $w_H(E^\star) \leq \omega$.

We will prove here that DRR is NP-complete by using the NP-completeness [BMvT78] of the Complete-Decoding decision problem (DCD):

> *Let n and k be two positive integers, $k < n$, G a matrix in $\mathcal{M}_{k,n}(\mathbf{F}_2)$.*
> *Let x be a vector from $\mathbf{F}_2^n$, and ω an integer.*
>
> **DCD**(G, x, ω): *is there a vector $e^\star \in \mathbf{F}_2^n$ such that* $\begin{cases} x + e^\star \in span(G) \\ w_H(e^\star) \leq \omega \end{cases}$?
>
> *(with $span(G)$ denoting the subspace generated by G's rows)*

Theorem 1 *Let n and k be two positive integers, $k < n$. Let G be a matrix in $\mathcal{M}_{k,n}(\mathbf{F}_2)$. Let x be a vector from $\mathbf{F}_2^n$, and ω an integer. Let X be the matrix from $\mathcal{M}_{k+1,n}(\mathbf{F}_2)$ whose first k rows are those of G, and whose last one is x. Then the answer to DCD(G, x, ω) is the same as the answer to DRR(X, k, ω).*

Proof: Let's assume that the answer to DRR(X, k, ω) is true.

Let $(G_1, \cdots, G_k)$ be the rows of G, and $E^\star$ be a matrix such that: $\begin{cases} rk(X + E^\star) = k \\ w_H(E^\star) \leq \omega \end{cases}$ with rows $(E_1^\star, \cdots, E_{k+1}^\star)$.

Since $rk(X + E^\star) = k$, there exists $(\lambda_1, \cdots, \lambda_k) \in \mathbf{F}_2^k$ such that

$$x + E_{k+1}^\star = \sum_{i=1}^{k} \lambda_i (G_i + E_i^\star).$$

Hence $x + \sum_{i=1}^{k} \lambda_i E_i^\star + E_{k+1}^\star \left(= \sum_{i=1}^{k} \lambda_i G_i \right) \in span(G)$.

But $w_H(\sum_{i=1}^{k} \lambda_i E_i^\star + E_{k+1}^\star) \leq w_H(E^\star) \leq \omega$, therefore the answer to DCD(G, x, ω) is true.

Reciprocally if the answer to DRR(X, k, ω) is false, we know that:

$$\forall E \in \mathcal{M}_{k+1,n}(\mathbf{F}_2) \qquad rk(X + E) = k \Rightarrow w_H(E) > \omega$$

Let $e \in \mathbf{F}_2^n$ be such that $x + e \in span(G)$, and let $E \in \mathcal{M}_{k+1,n}(\mathbf{F}_2)$ be the matrix whose first k rows are zero vectors and whose last one is e. We have $rk(X + E) = k$ hence $w_H(e) = w_H(E) > \omega$. This proves that the answer to DCD(G, x, ω) is false. □

We can then affirm that DCD is not "harder" than DRR and, since DCD is NP-complete [BMvT78] and DRR NP, that DRR is NP-complete.

3 A method to solve the Maximum Likelihood Code Recognition Problem in any case

In most cases, a recognition algorithm has to choose between all possible solutions by applying a criterion on certain properties of these solutions.

To design such an algorithm, we have to find some significant properties (cf. §3.1), then to find a criterion to apply on these properties which is very unlikely to be satisfied by a wrong solution (cf. §3.2 and §3.3), and, on the contrary, very likely to be satisfied by the good solution (cf. §3.4).

3.1 Presentation of the method

The received matrix X hides the structure of a k-dimensional subspace, the original code C, by addition of a sparse matrix E which is the transmission error and whose density is to be equal to the b.e.r. τ.

To retrieve the k-dimensional subspace, we will use the fact that the parity relationship between the codewords symbols are likely to be somehow still apparent: Actually if H is a parity check matrix of the original code, then HX^t is equal to HE^t whose low density ($< 1/2$) may be deduced from E's one.

Let x and h be rows of X and H respectively, the scalar product $< h, x >$ is a coefficient of XH^t, it is equal to 1 if and only if x have an odd number of transmission errors inside h's support, which happens with probability [Val99]

$$\begin{aligned} \Pr(< h, x >= 1 | h \in C^\perp) &= \sum_{i=0}^{\lfloor (w_H(h)-1)/2 \rfloor} \binom{w_H(h)}{2i+1} \tau^{2i+1} (1-\tau)^{w_H(h)-2i-1} \\ &= \frac{1-(1-2\tau)^{w_H(h)}}{2} \end{aligned} \tag{1}$$

Hence, we can expect hX^t to have a weight close to $\frac{1-(1-2\tau)^{w_H(h)}}{2} N$, which, for large N, would be very unlikely if X and h were not correlated (with fixed n and $\alpha < 1/2$, the probability that a random $[N, n]$-code comprises a word of weight less than αN tends to zero when N tends to infinity).

Our goal will then be to find $n-k$ independent vectors in $span(X^t)$ satisfying some low weight criteria (to be defined), and, if yX^t is such a vector, to assume that y belongs to $C^\perp$. This will allow us to reconstruct C.

3.2 False Alarm Probability

We will call false alarm the event of one (or more) of the $n-k$ found vectors $h_1, \ldots, h_{n-k}$ not being in $C^\perp$, which would lead to a wrong decision about C.

The low weight criterion must be designed so that, if y is not in $C^\perp$, then yX^t is very unlikely to satisfy it, in order to minimize the false alarm probability P_{fa}.

Let's calculate the latter in terms of the weights of h_i and $h_i X^t$ $(i = 1, \ldots, n-k)$:

$$P_{fa} = 1 - \prod_{i=1}^{n-k} \Pr(h_i \in C^\perp | (w_H(h_i X^t) \text{ and } w_H(h_i)))$$

With the approximation that the weight distributions of $C^\perp$ and of $span(X^t)$ are binomial, we find [Val99] that:

$$\Pr(y \in C^\perp | (w_H(y) = \omega_h \text{ and } w_H(yX^t) = \omega_v)) \approx \left(1 + \frac{2^k - 1}{(1-(1-2\tau)^{\omega_h})^{\omega_v}(1+(1-2\tau)^{\omega_h})^{N-\omega_v}}\right)^{-1}$$

which leads to:

$$P_{fa} \approx 1 - \prod_{i=1}^{n-k} \left(1 + \frac{2^k - 1}{(1-(1-2\tau)^{w_H(h_i)})^{w_H(h_i X^t)}(1+(1-2\tau)^{w_H(h_i)})^{N-w_H(h_i X^t)}}\right)^{-1} \quad (2)$$

Remark 1 *The False Alarm Probability we are talking about is a formal object; the real failure probability is directly linked to the implementation and to the context, but this approximation will allow us to define a pertinent criterion.*

3.3 A Criterion that will make the False Alarm Probability close to zero

Definition 1 *For all $\epsilon > 0$, we define $T_{N,\tau,\epsilon}$ as the function:*

$$\omega \mapsto \left\lfloor \frac{N\log(1+(1-2\tau)^\omega) + \log\epsilon - \log(n-k) - \log(2^k-1)}{\log(1+(1-2\tau)^\omega) - \log(1-(1-2\tau)^\omega)} \right\rfloor$$

Proposition 1 *For all $\epsilon > 0$ we have:*

$$\left(\forall i \in \{1, \ldots, n-k\} \quad w_H(h_i X^t) \leq T_{N,\tau,\epsilon}(w_H(h_i))\right) \Rightarrow P_{fa} < \epsilon$$

Proof: From (2), we can say that:

$$P_{fa} < \sum_{i=1}^{n-k} \frac{2^k - 1}{(1-(1-2\tau)^{w_H(h_i)})^{w_H(h_i X^t)}(1+(1-2\tau)^{w_H(h_i)})^{N-w_H(h_i X^t)}}$$

In other respects, we can prove easily that:

$$\omega_v \leq \mathrm{T}_{N,\tau,\epsilon}(\omega_h) \Leftrightarrow \frac{2^k - 1}{(1-(1-2\tau)^{\omega_h})^{\omega_v}(1+(1-2\tau)^{\omega_h})^{N-\omega_v}} \leq \frac{\epsilon}{n-k}$$

which gives directly the proposition. □

This suggests that, if we want P_{fa} to be less than a given value ϵ, we should choose the low weight criterion to be: $w_H(yX^t) \leq \mathrm{T}_{N,\tau,\epsilon}(w_H(y))$

3.4 Detection Probability

We call the detection probability P_{det} the probability that the code might be detected, that is the probability that there exists $n-k$ independent words of $C^{\perp}$ satisfying the criterion.

Let $P_c^{N,\tau,\epsilon}(\omega)$ denote the probability that a word of $C^{\perp}$ of weight ω satisfies the criterion:

$$\begin{aligned} P_c^{N,\tau,\epsilon}(\omega) &= \Pr(w_H(yX^t) \leq \mathrm{T}_{N,\tau,\epsilon}(\omega) | (y \in C^{\perp} \text{ and } w_H(y) = \omega)) \\ &= \sum_{i=0}^{\mathrm{T}_{N,\tau,\epsilon}(w)} \binom{N}{i} \left(\frac{1-(1-2\tau)^{\omega}}{2}\right)^{i} \left(\frac{1+(1-2\tau)^{\omega}}{2}\right)^{N-i} \end{aligned}$$

We then have:

$$P_{det} = 1 - \prod_{B \text{ basis of } C^{\perp}} \left(1 - \prod_{h \in B} P_c^{N,\tau,\epsilon}(w_H(h))\right)$$

and:

$$\sum_{B \text{ basis of } C^{\perp}} \left(\prod_{h \in B} P_c^{N,\tau,\epsilon}(w_H(h))\right) > P_{det} > \max_{B \text{ basis of } C^{\perp}} \left(\prod_{h \in B} P_c^{N,\tau,\epsilon}(w_H(h))\right)$$

It certainly is very difficult to know exactly the value of one of these three expressions, but we can say anyway that if a systematic basis of $C^{\perp}$ has a reasonable probability to satisfy the criterion, then P_{det} must be very close to 1; and since the elements of a systematic basis of $C^{\perp}$ have an average weight of $k/2$, we should have:

$$P_c^{N,\tau,\epsilon}(k/2)^{n-k} > 1/2 \Rightarrow P_{det} \simeq 1$$

We should then choose N such that $P_c^{N,\tau,\epsilon}(k/2)^{n-k} > 1/2$ in order to have a chance to detect the code.

Example 1 $n = 128$, $k = 64$, $\tau = 2\%$. *We want P_{fa} to be less than 1%. Figure 1 indicates that in order to have $P_{det} \simeq 1$, we should choose N greater than 2218.*

With $\tau = 1\%$, N greater than 560 should be enough; with $\tau = 0.5\%$, it goes down to 266; and $N = n$ is OK if $\tau < 0.1\%$.

4 Limits of the Algorithm

In theory, the presented method can solve the problem for all (n, k, τ) provided that we have enough words to put in the matrix X and enough time to run the algorithm, which will not always be the case.

For given values of the parameters (n, k, τ) of the problem, and for a given upper bound for the False Alarm probability, the algorithm will compute the

number N of needed words, make all the initializations and then start the search for low weight vectors.

The algorithm has been implemented in language C on a 300 MHz DEC Alpha computer, using a variant of Stern algorithm [CC98] to search for low weight vectors in a code; and it appears (but we had already guessed it) that this step becomes intractable when N and n grow.

4.1 Cost of the Search

The number of words to find to have $n-k$ independent words is in average:

$$n-k+\frac{1}{2^{n-k}-2}+\frac{3}{2^{n-k}-4}+\cdots+\frac{2^j-1}{2^{n-k}-2^j}+\cdots+\frac{2^{n-k-1}-1}{2^{n-k-1}}$$

which we can upperbound by $n-k+2$.

An algorithm to find low weight vectors in a code of length N generally works this way: At each iteration it picks up a set of candidate codewords and computes their weight. It is characterized by the cost C_{it} of an iteration, and by the probabilities $(P_s(\omega))_{\omega=1,\ldots,N}$ that the algorithm calculates the weight of a given word at a given iteration, if the word is of weight ω.

Assuming once again that the weight distribution of $C^\perp$ is binomial, the average number E_{it} of words we will find at each iteration is [Val99]:

$$\sum_{\omega_h=1}^{n}\frac{\binom{n}{\omega_h}}{2^k}\sum_{\omega_v=1}^{\mathrm{T}_{N,\tau,\epsilon}(\omega_h)}\Pr(w_H(hX^t)=\omega_v|(h\in C^\perp \text{ and } w_H(h)=w_h))P_s(\omega_v)=$$

$$\sum_{\omega_h=1}^{n}\frac{\binom{n}{\omega_h}}{2^k}\sum_{\omega_v=1}^{\mathrm{T}_{N,\tau,\epsilon}(\omega_h)}\binom{N}{\omega_v}\left(\frac{1-(1-2\tau)^{\omega_h}}{2}\right)^{\omega_v}\left(\frac{1+(1-2\tau)^{\omega_h}}{2}\right)^{N-\omega_v}P_s(\omega_v)$$

And the cost of the search will be in average less than $(n-k+2)C_{it}/E_{it}$.

4.2 An implementation

We consider here a variant of the Stern algorithm [CC98]. At each iteration, it considers a systematic basis of the code, and computes the weight of those of the $\binom{n}{4}$ sums of 4 basis vectors that are null inside a set of $\sigma = log_2(\frac{N-n}{2})$ redundancy positions.

The cost of each iteration is [Val99]:

$$C_{it}=\frac{(n+64)(N-n)}{2}+n\left(\frac{n}{2}-1\right)\left(\sigma+\frac{n}{2}\left(\frac{n}{2}-1\right)\left(1-\frac{\sigma}{N-n}\right)+\frac{64}{4}\right)$$

And we have $P_s(\omega_v)=\begin{cases}\binom{n}{4}\binom{N-n-\sigma}{\omega_v-4}/\binom{N}{\omega_v} & \text{if } 4\le\omega_v\le N-n-\sigma+4,\\ 0 & \text{else.}\end{cases}$

We have improved the algorithm in the following manner: after the computation of N, the algorithm evaluates how much time it will need to run the search, and, in case this seems impossible, suggests to make some adjustments on P_{fa} or P_{det}.

5 Conclusion

There is a newcomer in the class of NP-complete problems. Its applications are rather obvious (electronic war or any non cooperative context), but to find a good way to solve it is not.

The solution we have given works with parameters that are commonly used (cf. Figure 2), but could be insufficient in some case (interleavers, product codes, concatenated codes may give rise to very long linear code) and should be extended to non binary codes

The possible adjustment of the False Alarm probability may be very useful when the algorithm works serially with other recognition algorithms (of modulation, of ambient space, of length, of dimension, of decodable structure, of codewords/messages correspondence...) offering the same possibility. Actually this should give ability to get the global process under control.

A means of recognizing codes with parameter intractable with this method could be to use soft information; research using this approach are under progress.

References

[BMvT78] E.R. Berlekamp, R.J. McEliece, and H.C. van Tilborg. On the inherent intractability of certain coding problems. *IEEE Transactions on Information Theory*, 24(3), May 1978.

[CC98] A. Canteaut and F. Chabaud. A new algorithm for finding minimum-weight words in a linear code: Application to McEliece's cryptosystem and to narrow-sense BCH codes of length 511. *IEEE Transactions on Information Theory*, 44(1):367–378, January 1998.

[Fil97] E. Filiol. Reconstruction of convolutional encoders over gf(q). In P. Charpin and G. Cohen, editors, *Cryptography and Coding*, number 1355 in LNCS, pages 100–109. Springer-Verlag, 1997.

[Pla96] G. Planquette. *Identification de trains binaires codés.* Thèse de doctorat, Université de Rennes 1, December 1996.

[Val99] A. Valembois. Recognition of a binary linear code. In preparation, 1999.

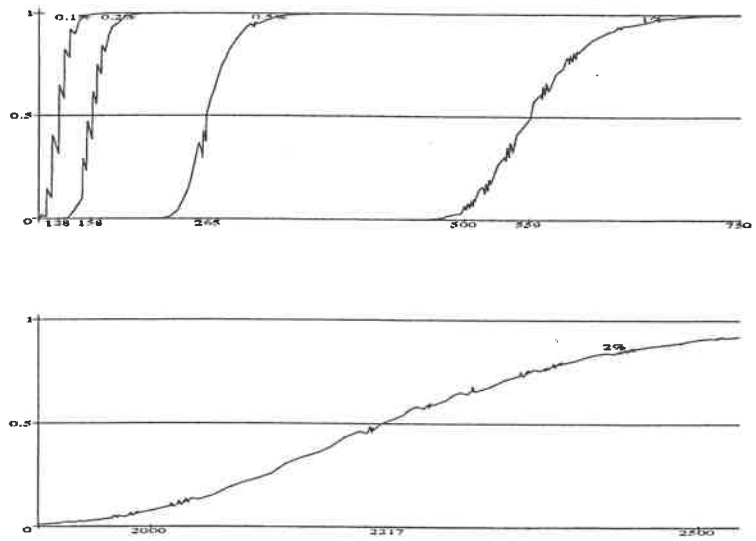


Figure 1: $P_c(k/2)^{N,\tau,\epsilon}(k/2)^{n-k}$ with respect to N, for $n = 128, k = 64, \epsilon = 1\%, and \tau = 0.1\%, 0.2\%, 0.5\%, 1\%, 2\%$

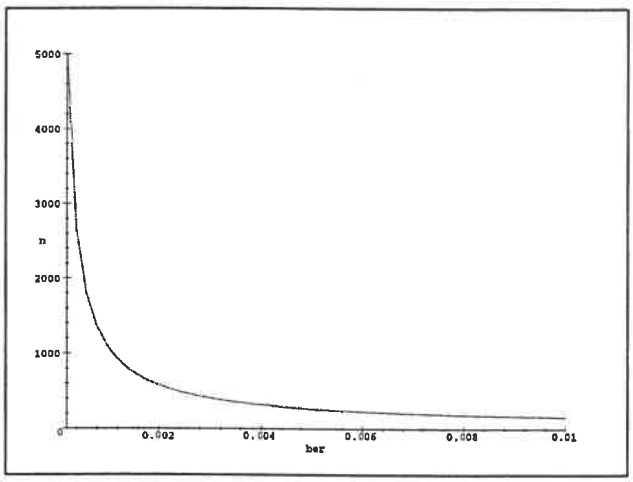


Figure 2: Reachable area in $\{\tau\} \times \{n\}$, with $k = \lfloor n/2 \rfloor$ and $\epsilon = 1\%$.

Bounds on the Sizes of Ternary Weight-Constrained Codes

Mattias Svanström*
Dept. of Electrical Engineering,
Linköping University, SE-581 83 Linköping, Sweden
Email: mattias@isy.liu.se

Abstract

We derive upper and lower bounds on the size of a ternary block code with the restrictions of constant weight and constant composition. We also present tables of the best known bounds for these two types of codes.

1 Introduction

Denote the maximal size of a q-ary block code with length n and minimum Hamming distance d by $A_q(n,d)$. If every codeword has Hamming weight w, we have a *constant-weight code* of weight w and we denote its maximal size by $A_q(n,d,w)$. If, in addition, the number of occurrences of each symbol in the codeword is the same for all codewords, we have a *constant-composition* code. Let the *composition vector*

$$\mathbf{w}_c = [w_0, w_1, \ldots, w_{q-1}]$$

be a vector containing the number of occurrences of each symbol in a codeword, where we use subscript c for denoting constant-composition. We denote the maximal size of a q-ary constant-composition code by $A_q(n,d,\mathbf{w}_c)$.

By interchanging symbols i and j everywhere they occur in a constant-composition code, the entries w_i and w_j in the composition vector are swapped. Because of this property it is enough to investigate codes in *standard form*, which we define as codes with

$$w_0 \geq w_1 \geq \ldots \geq w_{q-1}.$$

We sometimes mention the *weight* of a constant-composition code, referring to the Hamming weight

$$w = \sum_{i=1}^{q-1} w_i.$$

The problem of determining upper and lower bounds on the sizes of binary unrestricted codes and binary constant-weight codes has been investigated to some

*This work was supported by the Swedish Research Council for Engineering Sciences under grant 271-97-532.

extent. An excellent survey of the problem as well as a compilation of tables containing the best known binary bounds has been done by Brouwer et. al. [5].

A table of results on ternary unrestricted codes first appeared in an article by Vaessens, Aarts and van Lint [13], and recently an updated table of bounds on $A_3(n,d)$ was published in [4]. The first table of ternary constant-weight codes can be found in the thesis by Tarnanen [12]. Further results on ternary constant-weight and constant-composition codes have come from Fu, Vinck and Shen [6], Bogdanova [1], [2], Bogdanova and Ocetarova [3] and the author [9], [10] and [11].

2 Tables

Tables 2 and 3 contain the results on lower and upper bounds on $A_3(n,d,\mathbf{w}_c)$ and $A_3(n,d,w)$ respectively. Each entry in the table is followed by a pair of subscripts indicating the lower and upper bound used to derive the value. Keys to the indices are presented in Tables 4 and 5. Wherever a dash occurs, a code with the corresponding parameters can only have one codeword.

Some of the bounds presented by Bogdanova and Ocetarova [3] were found by exhaustive search. We give explicit proofs of these bounds in case the proofs are known to us. We have found several new codes, which are presented here. We also give new descriptions of codes with the same parameters as codes that have appeared earlier, but with more structure than has previously been indicated.

Remark: The references [2] and [3] contain some unfortunate errors. The most important correction is that we have established the value $A_3(8,3,[4,2,2]) = 84$, which is greater than the upper bound stated in [3].

3 Upper Bounds

For binary constant-weight codes, the Johnson bound [7], often turns out to be the strongest bound. For ternary constant-weight codes, we presented several forms of the Johnson bounds in [10]. For general q-ary constant-composition codes, the recursive version of the Johnson bound takes this form:

Theorem 1 (c. f. Theorem 3.14 in [10]) *Let $\mathbf{w}_c^{(i)}$ denote the composition vector of a code with composition vector $\mathbf{w}_c$ shortened with respect to the symbol i,*

$$\mathbf{w}_c^{(i)} = [w_0, w_1, \ldots, w_{i-1}, w_i - 1, w_{i+1}, \ldots, w_{q-1}].$$

The maximal number of codewords in a q-ary constant-composition code with composition $\mathbf{w}_c$ fulfills the inequality

$$A_q(n,d,\mathbf{w}_c) \leq \left\lfloor \frac{n}{w_i} A_q(n-1,d,\mathbf{w}_c^{(i)}) \right\rfloor$$

for all i, $0 \leq i \leq q-1$.

Ternary constant-weight codes with $d = 2w-1$ were investigated in [11]. We also have some results on constant-composition codes with $d = 2w-1$.

Theorem 2 *The maximal number of codewords in a ternary constant-composition code with $d = 2w - 1$ and $w_2 = 1$ is*

$$A_3(n, 2w-1, [n-w, w-1, 1]) = \left\lfloor \frac{n}{w-1} \right\rfloor .$$

Proof: The upper bound follows by application of Theorem 1 with $i = 1$.

We construct a code meeting this bound by taking $0^{n-w}1^{w-1}2$ as the first codeword. Shift this tuple cyclically $w-1$ steps and take the resulting tuple as the next codeword. Repeat this until the code contains $\lfloor n/(w-1) \rfloor$ codewords. The supports of two codewords either do not overlap or overlap in exactly one position, in which one codeword is 1 and the other codeword is 2, thus the minimum distance is $d = 2w - 1$. ■

The codes constructed in the proof of Theorem 2 have the property that, when considering the order in which the codewords were constructed, codewords following each other have distance d or $d+1$, and codewords that do not follow each other have distance $d+1$. This observation is useful in the following juxtaposition construction.

Theorem 3 *For a code in standard form, if n is even, if w is even and if $n \geq 5w - 10$, the maximal number of codewords in a ternary constant-composition code with $d = 2w - 1$ and $w_2 = 2$ is*

$$A_3(n, 2w-1, [n-w, w-2, 2]) = \left\lfloor \frac{n}{w-2} \right\rfloor .$$

Proof: The upper bound follows by application of Theorem 1 with $i = 1$.

We construct codes meeting this bound with equality as follows. Take the $M = \lfloor n/(w-2) \rfloor$ codewords of an $(n/2, w-1, [n/2-w/2, w/2-1, 1])$ code as constructed in the proof of Theorem 2. Index the codewords by 0 to $M-1$. By their construction as cyclic shifts, codewords whose indices differ by one (as well as codewords 1 and M if $w-2$ divides n) have distance $w-1$. Codewords whose indices differ by two or more have distance w. A code with minimum distance $2w-1$ can therefore be constructed by juxtaposition, while making sure that codewords that differ by $w-1$ in one half differ by w in the other half.

If M is odd, we juxtapose a codeword with index j and a codeword with index $2j \pmod M$. In this way, if two codewords have adjacent indices on their left halves, the indices of their right halves will not be adjacent.

If M is even, we divide the indices into two ordered sets, $I_1 = \{0, 1, \ldots, (M-1)/2\}$ and $I_2 = \{M/2, M/2+1, \ldots, M-1\}$. We take indices 0 to $M-1$ in order from low to high for the left halves. For the right halves, we alternate between taking indices from I_1 and I_2, taking indices from low to high from index group I_1 and taking alternately the lowest and the highest index left in group I_2. The only thing we need to check is that index $M/2$ is not chosen immediately before or after index $M/2$ and that index $M-1$ is not chosen after index 0 or as the last index. By assumption we have $M \geq 6$, so none of these situations occur. ■

Theorem 4 *The maximal number of codewords of a ternary $(n, d, w) = (6, 5, 5)$ constant-weight code is*

$$A_3(6, 5, 5) = 3.$$

Proof: Assume that there exists a $(6,5,5)$ code C with $M = 4$ codewords. Let column k contain $v_0(k)$ zeros, $v_1(k)$ ones and $v_2(k)$ twos. We count in two ways $\sum_{\mathbf{c}_1 \in C} \sum_{\mathbf{c}_2 \in C} d(\mathbf{c}_1, \mathbf{c}_2)$ to find that

$$\sum_{k=1}^{n} 2v_0(k)v_1(k) + 2v_0(k)v_2(k) + 2v_1(k)v_2(k) \geq M(M-1)d = 60.$$

The only possible distribution of symbols over the columns fulfilling this is to have $[v_0(k), v_1(k), v_2(k)] = [2,1,1]$, $[1,2,1]$ or $[1,1,2]$. However, this implies that the codeword matrix contains at least six zeros, whereas it can only contain four. We have a contradiction and conclude that $M \leq 3$. ■

4 Codes from Jacobsthal Matrices

Let p be an odd prime. Consider the Galois field F_{p^m} with elements $\alpha_1, \alpha_2, \ldots, \alpha_{p^m}$ and let S be the set of nonzero square elements $S = \{\, \alpha_i^2 : \alpha_i \in F_{p^m},\ \alpha_i \neq 0 \,\}$. A $p^m \times p^m$ Jacobsthal matrix $Q = (q_{ij})$ is defined by

$$q_{ij} = \begin{cases} 0 & \text{if } \alpha_i = \alpha_j, \\ 1 & \text{if } \alpha_i - \alpha_j \in S, \\ -1 & \text{if } \alpha_i - \alpha_j \notin S \text{ and } \alpha_i \neq \alpha_j. \end{cases}$$

Define the maps $f_1 : \{-1,0,+1\} \to \{0,1,2\}$ and $f_2 : \{-1,0,+1\} \to \{0,1,2\}$ by

$$f_1 : \begin{array}{ccc} -1 & \mapsto & 0 \\ 0 & \mapsto & 2 \\ +1 & \mapsto & 1, \end{array} \qquad \text{and} \qquad f_2 : \begin{array}{ccc} -1 & \mapsto & 1 \\ 0 & \mapsto & 2 \\ +1 & \mapsto & 0. \end{array}$$

Fu, Vinck and Shen [6, Proposition 5.3] state that

$$A_3\left(p^m, \frac{p^m+3}{2}, p^m - 1\right) = p^m.$$

Let C_1 be the code we get by applying f_1 to the rows of Q. It follows by the properties of Jacobsthal matrices that C_1 is a ternary constant-composition code, showing that

$$A_3\left(p^m, \frac{p^m+3}{2}, \left[\frac{p^m-1}{2}, \frac{p^m-1}{2}, 1\right]\right) = p^m.$$

By interchanging zeros and twos we get the code derived by Fu, Vinck and Shen. However, we can get more results out of the Jacobsthal matrices.

Let C_2 be the code we get by applying f_2 to the rows of the same Jacobsthal matrix. The code $C_1 \cup C_2$ is a constant composition code, showing that

$$A_3\left(p^m, \frac{p^m+1}{2}, \left[\frac{p^m-1}{2}, \frac{p^m-1}{2}, 1\right]\right) \geq 2p^m.$$

This code is optimal at least for p^m in the range covered by the tables. Also, the code

$$\{1\,\mathbf{c} : \mathbf{c} \in C_1\} \cup \{2\,\mathbf{c} : \mathbf{c} \in C_2\} \cup \{0\ 1^{(p^m)}\} \cup \{0\ 2^{(p^m)}\}$$

is an optimal constant-weight code, showing that

$$A_3\left(p^m + 1, \frac{p^m+3}{2}, p^m\right) = 2p^m + 2.$$

5 Cyclic Codes

The codes in Table 1 are used to obtain some of the lower bounds. The codes contain all cyclic shifts of the orbit representatives, which are given in base-9 notation.

In some cases, sporadic single codewords can be added to the cyclic codes without decreasing the minimum distance. We have $A_3(8,3,6) \geq 163$ by taking the cyclic $(8,3,6)$ code and adding the codewords 1344, 3414 and 4134. We also get $A_3(8,4,7) \geq 25$ by taking the $(8,4,[4,3,1])$ code, swapping zeros and twos, and adding the codeword 1444.

A cyclic code can also be described as a variant of a group code, with the group generated by a cyclic shift of the coordinates, and multiple generators. Generalizing this technique, we find that $A_3(8,3,[4,3,1]) = 56$ from the code we get by applying the permutation group generated by $\{(2\ 3\ 4\ 5\ 6\ 7\ 8), (1\ 4\ 2)(5\ 6\ 7)\}$ to the tuple $0^4 1^3 2$.

The $(7,7,[3,2,2])$ code with $M = 2$ and the $(8,8,[3,3,2])$ code with $M = 2$ are trivial to construct (by cyclically shifting $0^{w_0}1^{w_1}2^{w_2}$).

A $(10,8,[5,3,2])$ code with $M = 4$ codewords is

$$\begin{bmatrix} 1200\ 1200\ 01 \\ 0120\ 0120\ 10 \\ 0012\ 0012\ 01 \\ 2001\ 2001\ 10 \end{bmatrix}.$$

6 Codes from Unrestricted Codes

In [10] we derived ternary constant-weight codes from ternary unrestricted codes by taking all codewords having a certain weight. The same technique can also be used to derive ternary constant-composition codes.

The entries in Table 2 that originate from unrestricted codes are $A_3(6,4,[2,2,2]) = 15$ and $A_3(9,6,[3,3,3]) = 24$ from Hadamard codes, plus a number of codes with minimum distance 6, which we can derive from the extended Golay code.

In what follows, shortening refers to shortening in the first position. Let C_{12} be the $(n,d) = (12,6)$ Golay code with generator matrix given in [10]. C_{12} has complete weight enumerator

$$W(z_0,z_1,z_2) = z_0^{12}+z_1^{12}+z_2^{12}+22(z_0^6z_1^6+z_0^6z_2^6+z_1^6z_2^6)+220(z_0^6z_1^3z_2^3+z_0^3z_1^6z_2^3+z_0^3z_1^3z_2^6).$$

The codewords of composition $[6,3,3]$ form a constant-composition code proving $A_3(12,6,[6,3,3]) = 220$. Let C_{11} be C_{12} shortened with respect to zeros. C_{11} has complete weight enumerator

$$W(z_0,z_1,z_2) = z_0^{11} + 11(z_0^5z_1^6 + z_0^5z_2^6) + 55(z_0^2z_1^6z_2^3 + z_0^2z_1^3z_2^6) + 110z_0^5z_1^3z_2^3.$$

Select the words of C_{11} having composition $[2,3,6]$. After swapping zeros and twos to put it in standard form, the resulting code proves $A_3(11,6,[6,3,2]) = 55$. Shorten this code with respect to ones. The resulting code gives us $A_3(10,6,[6,2,2]) = 15$. Shorten this code with respect to zeros. The code we arrive at shows us that $A_3(9,6,[5,2,2]) = 9$.

The codewords of C_{11} having composition $[5,3,3]$ form a code that proves the value $A_3(11,6,[5,3,3]) = 110$. Shorten this code with respect to twos. The resulting code gives us $A_3(10,6,[5,3,2]) = 30$.

Let C_{10} be C_{11} shortened with respect to zeros. C_{10} has complete weight enumerator

$$W(z_0,z_1,z_2) = z_0^{10} + 5(z_0^4z_1^6 + z_0^4z_2^6) + 10(z_0z_1^6z_2^3 + z_0z_1^3z_2^6) + 50z_0^4z_1^3z_2^3.$$

Select the words of C_{10} having composition $[1,3,6]$. After swapping zeros and twos to put it in standard form, the resulting code proves $A_3(10,6,[6,3,1]) = 10$. Shorten this code with respect to zeros. From the resulting code we have $A_3(9,6,[5,3,1]) = 6$.

The codewords of C_{10} having composition $[4,3,3]$ form a code that proves the value $A_3(10,6,[4,3,3]) = 50$. Shorten this code with respect to twos. The resulting code gives us $A_3(9,6,[4,3,2]) = 15$. Shorten this code with respect to twos. We get a code proving $A_3(8,6,[4,3,1]) = 4$. By instead shortening the $(9,6,[4,3,2])$ code with respect to ones we get a code proving that $A_3(8,6,[4,2,2]) = 5$. Shorten this code with respect to zeros. The resulting code gives us $A_3(7,6,[3,2,2]) = 3$.

A different construction, based on binary unrestricted codes, is the one that is used in [8] to find perfect ternary $(n,d,w) = (2^r,3,2^r-1)$ constant-weight codes with $M = 2^{2^r-1}$ codewords. This construction is based on concatenation of cosets of binary Hamming codes. Shortened versions of these codes are also either optimal or very good. In this way, the entries for $n = 6,7,8,9$ and 10 in Table 3 are obtained.

7 Juxtaposition

We *juxtapose*, i. e. put side by side, m copies of the $(n,d,\mathrm{w}_c) = (3,3,[1,1,1])$ code $\{012,201,120\}$ to get a $(3m,3m,[m,m,m])$ code with $M = 3$ codewords.

We juxtapose two $(5,4,[2,2,1])$ codes to create a code proving $A_3(10,8,[4,4,2]) = 5$. We also juxtapose a $(5,4,[2,2,1])$ code and a $(5,4,[2,1,2])$ code to get a code proving that $A_3(10,8,[4,3,3]) = 5$. We juxtapose a $(5,4,[2,2,1])$ code and a $(3,3,[1,1,1])$ code to get an $(8,7,[3,3,2])$ code with $M = 3$. Finally, we juxtapose a $(6,5,[3,2,1])$ code and a $(3,3,[1,1,1])$ code to get a $(9,8,[4,3,2])$ code with $M = 3$.

8 Lexicographic Codes

In order to have lower bounds for all entries in the tables, we have extended the lexicographic searches from [3]. All new codes presented in our tables are lexicographic codes with standard lexicographic ordering of the space but using the seed giving the best possible lexicographic code. Since only two of these codes are optimal we present the rest without giving the details. The optimal codes prove $A_3(6,5,4) = 4$ by a lexicographic code with seed 44 and $A_3(10,5,[6,3,1]) = 13$ by a lexicographic code with seed 54, both in base-9 notation. We expect most lexicographic codes to be beaten by codes with more structure.

9 Summary

We have derived some new upper bounds and constructed new optimal ternary constant-composition and constant-weight codes. The ternary Golay code not only gives us many optimal constant-weight codes but also several optimal constant-composition codes.

References

[1] G. T. Bogdanova. Bounds for the maximum size of ternary constant composition weight codes. In *Proceedings of the II International Workshop on Optimal Codes '98*, pages 15–18. Institute of Mathematics and Informatics, Bulgaria, Jun 1998.

[2] G. T. Bogdanova. Some lexicogrphic codes. In *Proceedings of the II International Workshop on Optimal Codes '98*, pages 9–14. Institute of Mathematics and Informatics, Bulgaria, Jun 1998.

[3] G. T. Bogdanova and D. S. Ocetarova. Some ternary constant-composition codes. In *Proceedings of the Sixth International Workshop on Algebraic and Combinatorial Coding Theory*, pages 41–45. Pskov, Russia, Sep 1998.

[4] A. E. Brouwer, H. O. Hämäläinen, P. R. J. Östergård, and N. J. A. Sloane. Bounds on mixed binary/ternary codes. *IEEE Transactions on Information Theory*, 44(1):140–161, Jan 1998.

[5] A. E. Brouwer, J. B. Shearer, N. J. A. Sloane, and W. D. Smith. A new table of constant weight codes. *IEEE Transactions on Information Theory*, IT-36(6):1334–1380, Nov 1990.

[6] F.-W. Fu, A. J. H. Vinck, and S.-Y. Shen. On the constructions of constant-weight codes. *IEEE Transactions on Information Theory*, 44(1):328–333, Jan 1998.

[7] S. M. Johnson. A new upper bound for error-correcting codes. *IRE Transactions on Information Theory*, IT-8(3):203–207, Apr 1962.

[8] M. Svanström. A class of perfect ternary constant-weight codes. *Submitted to Designs, Codes and Cryptography.*

[9] M. Svanström. A lower bound for ternary constant weight codes. *IEEE Transactions on Information Theory*, 43(5):1630–1632, Sep 1997.

[10] M. Svanström. *Ternary Codes with Constant Weight.* Licentiate thesis, Linköping University, Sep 1997. Thesis No. 646.

[11] M. Svanström. Constructing optimal ternary constant-weight codes by placing pieces on chessboards. In *Proceedings 1998 IEEE International Symposium on Information Theory*, page 66. Cambridge, MA, USA, Aug 1998.

[12] H. Tarnanen. *An Approach to Constant Weight and Lee Codes by Using the Methods of Association Schemes.* Number 182 in Ann. Univ. Turku Ser. A I. University of Turku, 1982.

[13] R. J. M. Vaessens, E. H. L. Aarts, and J. H. van Lint. Genetic algorithms in coding theory—a table for $A_3(n, d)$. *Discrete Applied Mathematics*, 45:71–87, 1993.

Table 1: Cyclic Weight-Constrained Codes

Bound	Orbit Representatives
$A_3(6,3,[3,2,1]) = 12$	15, 71
$A_3(6,3,[2,2,2]) = 30$	48, 84, 125, 167, 172
$A_3(7,3,[4,2,1]) = 21$	17, 121, 132
$A_3(7,3,[3,3,1]) = 28$	45, 164, 241, 337
$A_3(7,3,[3,2,2]) = 42$	48, 84, 165, 237, 352, 367
$A_3(8,3,[5,2,1]) = 24$	15, 71, 321
$A_3(8,3,[4,2,2]) = 84$	57, 75, 138, 264, 325, 367, 521, 532, 635, 712, 1212
$A_3(8,3,[3,3,2]) \geq 96$	148, 254, 355, 472, 517, 564, 724, 841, 1127, 1152, 1235, 1365
$A_3(10,3,[7,2,1]) = 40$	15, 71, 1021, 1032
$A_3(6,4,[3,2,1]) = 6$	17
$A_3(7,4,[3,2,2]) = 21$	55, 264, 372
$A_3(8,4,[4,3,1]) = 24$	135, 214, 337
$A_3(8,4,[4,2,2]) = 36$	57, 235, 325, 642, 1212
$A_3(8,4,[3,3,2]) = 56$	177, 348, 465, 724, 841, 1235, 1422
$A_3(10,4,[7,2,1]) = 20$	17, 1061
$A_3(7,5,[3,2,2]) = 7$	118
$A_3(8,5,[4,3,1]) = 8$	115
$A_3(8,5,[3,3,2]) \geq 16$	375, 834
$A_3(8,6,[3,3,2]) = 8$	542
$A_3(7,3,3) = 28$	17, 68, 121, 132
$A_3(7,3,4) \geq 56$	45, 57, 128, 171, 224, 252, 265, 364
$A_3(7,3,5) \geq 70$	145, 187, 274, 355, 377, 424, 538, 572, 675, 768
$A_3(8,3,4) = 112$	48, 84, 117, 165, 237, 252, 314, 368, 412, 505, 628, 634, 671, 821, 1212
$A_3(8,3,5) \geq 152$	145, 157, 274, 354, 377, 472, 568, 688, 765, 781, 824, 837, 852, 1128, 1167, 1217, 1265, 1342, 1522
$A_3(8,3,6) \geq 160$	445, 457, 574, 588, 777, 785, 848, 854, 1155, 1174, 1278, 1428, 1472, 1517, 1658, 1684, 2245, 2287, 2375, 2424, 2828
$A_3(9,3,3) = 48$	15, 67, 308, 311, 632, 2062
$A_3(6,4,3) = 8$	17, 222
$A_3(7,4,3) = 14$	34, 82
$A_3(8,4,3) = 16$	17, 322
$A_3(8,4,4) \geq 42$	57, 235, 314, 368, 821, 1111
$A_3(8,4,5) \geq 72$	147, 285, 374, 468, 542, 658, 781, 1164, 2238

Table 2: Bounds on $A_3(n, d, \mathbf{w}_c)$

$(n, d, \mathbf{w}_c)$	$d=3$	$d=4$	$d=5$	$d=6$	$d=7$	$d=8$
$(4, d, [2,1,1])$	4_{dw}	2_{dd}	–	–	–	–
$(5, d, [3,1,1])$	5_{dw}	2_{dw}	–	–	–	–
$(5, d, [2,2,1])$	10_{j1}	5_{j1}	2_{dw}	–	–	–
$(6, d, [4,1,1])$	6_{dw}	3_{dd}	–	–	–	–
$(6, d, [3,2,1])$	12_{c2}	6_{c1}	3_{d1}	2_{dd}	–	–
$(6, d, [2,2,2])$	30_{c1}	15_{u1}	3_{hx}	3_{xw}	–	–
$(7, d, [5,1,1])$	7_{dw}	3_{dd}	–	–	–	–
$(7, d, [4,2,1])$	21_{c1}	9_{bx}	3_{d1}	2_{dd}	–	–
$(7, d, [3,3,1])$	28_{c1}	14_{j1}	7_{j1}	2_{aw}	2_{dw}	–
$(7, d, [3,2,2])$	42_{c1}	21_{c1}	7_{c0}	3_{uw}	2_{tw}	–
$(8, d, [6,1,1])$	8_{dw}	4_{dd}	–	–	–	–
$(8, d, [5,2,1])$	24_{c2}	12_{b1}	4_{d1}	2_{dd}	–	–
$(8, d, [4,3,1])$	56_{c1}	24_{c1}	8_{c1}	4_{u0}	2_{d1}	2_{dd}
$(8, d, [4,2,2])$	84_{c1}	36_{c1}	12_{b1}	5_{uw}	2_{aw}	2_{dd}
$(8, d, [3,3,2])$	$96-112_{c1}$	56_{c1}	$16-18_{c1}$	8_{c1}	3_{xw}	2_{tw}
$(9, d, [7,1,1])$	9_{dw}	4_{dd}	–	–	–	–
$(9, d, [6,2,1])$	$34-36_{b1}$	18_{b1}	4_{d1}	3_{dd}	–	–
$(9, d, [5,3,1])$	$65-72_{b1}$	$31-36_{b1}$	$10-12_{b1}$	6_{u1}	3_{d1}	2_{dd}
$(9, d, [5,2,2])$	$97-108_{l1}$	$43-54_{l1}$	$16-18_{b1}$	9_{u1}	3_{b0}	2_{dd}
$(9, d, [4,4,1])$	$116-126_{b1}$	$42-54_{b1}$	18_{j1}	9_{j1}	3_{bb}	2_{aw}
$(9, d, [4,3,2])$	$174-252_{l1}$	$71-108_{b1}$	$22-36_{b1}$	15_{u1}	5_{bb}	3_{xw}
$(9, d, [3,3,3])$	$227-336_{b1}$	$120-168_{l1}$	$29-54_{l1}$	24_{u1}	6_{bw}	3_{hw}
$(10, d, [8,1,1])$	10_{dw}	5_{dd}	–	–	–	–
$(10, d, [7,2,1])$	40_{c2}	20_{c1}	5_{d1}	3_{dd}	–	–
$(10, d, [6,3,1])$	$96-120_{l1}$	$45-60_{l1}$	13_{l1}	10_{u1}	3_{d1}	2_{dd}
$(10, d, [6,2,2])$	$147-180_{l1}$	$64-90_{l1}$	$19-20_{l1}$	15_{u1}	5_{d1}	2_{dd}
$(10, d, [5,4,1])$	$146-180_{l1}$	$66-90_{l1}$	$20-30_{l1}$	$10-15_{l1}$	$5-6_{lw}$	2_{aw}
$(10, d, [5,3,2])$	$284-360_{l1}$	$116-180_{l1}$	$35-60_{l1}$	30_{u1}	$7-9_{lw}$	4_{tw}
$(10, d, [4,4,2])$	$399-630_{l1}$	$146-270_{l1}$	$42-90_{l1}$	$19-37_{l1}$	$10-11_{lw}$	5_{xw}
$(10, d, [4,3,3])$	$492-840_{l1}$	$187-360_{l1}$	$53-120_{l1}$	50_{u1}	$10-15_{l0}$	5_{xw}

Table 3: Bounds on $A_3(n, d, w)$

(n, d, w)	$d = 3$	$d = 4$	$d = 5$	$d = 6$	$d = 7$	$d = 8$
$(4, d, 2)$	4_{dd}	2_{dd}	–	–	–	–
$(4, d, 3)$	8_{u1}	2_{wt}	–	–	–	–
$(5, d, 2)$	5_{dd}	2_{dd}	–	–	–	–
$(5, d, 3)$	12_{bx}	5_{w3}	2_{dd}	–	–	–
$(5, d, 4)$	10_{w0}	5_{w1}	2_{wt}	–	–	–
$(6, d, 2)$	6_{dd}	3_{dd}	–	–	–	–
$(6, d, 3)$	$18 - 20_{b1}$	8_{c1}	4_{dd}	2_{dd}	–	–
$(6, d, 4)$	30_{w0}	15_{w1}	4_{lw}	3_{ww}	–	–
$(6, d, 5)$	24_{s1}	12_{j1}	3_{wx}	2_{wt}	–	–
$(7, d, 2)$	7_{dd}	3_{dd}	–	–	–	–
$(7, d, 3)$	28_{c1}	14_{c1}	4_{dd}	2_{dd}	–	–
$(7, d, 4)$	$56 - 70_{c1}$	$23 - 28_{b1}$	$7 - 9_{w0}$	3_{ww}	2_{dd}	–
$(7, d, 5)$	$70 - 84_{c1}$	$32 - 42_{b1}$	$9 - 10_{l0}$	3_{ww}	2_{wt}	–
$(7, d, 6)$	56_{s1}	$14 - 20_{wl}$	7_{w1}	2_{wt}	2_{wt}	–
$(8, d, 2)$	8_{dd}	4_{dd}	–	–	–	–
$(8, d, 3)$	$34 - 37_{b1}$	16_{b1}	5_{dd}	2_{dd}	–	–
$(8, d, 4)$	112_{c1}	$42 - 56_{c1}$	$12 - 16_{l1}$	5_{u4}	2_{dd}	2_{dd}
$(8, d, 5)$	$152 - 224_{c1}$	$72 - 89_{c1}$	$17 - 26_{l0}$	8_{w0}	3_{ww}	2_{w3}
$(8, d, 6)$	$163 - 224_{c1}$	$56 - 80_{w0}$	$16 - 22_{wl}$	8_{w1}	3_{ww}	2_{wt}
$(8, d, 7)$	128_{p1}	$25 - 37_{cl}$	16_{j1}	4_{u1}	2_{wt}	2_{wt}
$(9, d, 2)$	9_{dd}	4_{dd}	–	–	–	–
$(9, d, 3)$	48_{c1}	24_{b1}	6_{dd}	3_{dd}	–	–
$(9, d, 4)$	$134 - 166_{b1}$	$56 - 72_{b1}$	$18 - 22_{l1}$	9_{u1}	3_{dd}	2_{dd}
$(9, d, 5)$	$284 - 403_{b1}$	$100 - 200_{l0}$	$29 - 57_{l1}$	18_{u1}	$5 - 6_{w0}$	3_{ww}
$(9, d, 6)$	$385 - 672_{l1}$	$110 - 240_{l0}$	$35 - 66_{l0}$	24_{w1}	6_{ww}	3_{ww}
$(9, d, 7)$	$357 - 576_{l1}$	$86 - 166_{l0}$	$30 - 56_{l1}$	18_{u0}	$5 - 6_{w3}$	3_{ww}
$(9, d, 8)$	$148 - 180_{l0}$	$44 - 83_{l1}$	$18 - 36_{w1}$	9_{w1}	$3 - 4_{w3}$	2_{wt}
$(10, d, 2)$	10_{dd}	5_{dd}	–	–	–	–
$(10, d, 3)$	$56 - 60_{b1}$	26_{b1}	6_{dd}	3_{dd}	–	–
$(10, d, 4)$	$193 - 240_{b1}$	$84 - 120_{b1}$	30_{u1}	15_{u1}	5_{dd}	2_{dd}
$(10, d, 5)$	$481 - 664_{b1}$	$116 - 288_{w1}$	$46 - 88_{l1}$	36_{u1}	$8 - 9_{l3}$	4_{w3}
$(10, d, 6)$	$640 - 1343_{g1}$	$187 - 600_{w0}$	$67 - 165_{l0}$	60_{u1}	$11 - 15_{l0}$	5_{w3}
$(10, d, 7)$	$732 - 1920_{g1}$	$187 - 553_{w0}$	$71 - 186_{l0}$	60_{u0}	$12 - 17_{l1}$	5_{w3}
$(10, d, 8)$	$549 - 900_{g0}$	$146 - 415_{w1}$	$55 - 140_{l1}$	45_{u1}	$10 - 11_{w3}$	5_{w3}
$(10, d, 9)$	$320 - 400_{s1}$	$66 - 184_{w1}$	$28 - 60_{l0}$	20_{j1}	$5 - 6_{w3}$	2_{wt}

Lower bounds	
a	From shorter code by trivial extension
b	Bogdanova and Ocetarova, [3]
c	Cyclic or permutation code
d	Code with $d = 2w$ or $d = 2w - 1$
h	From code with higher minimum distance
j	Code from Jacobsthal matrix
l	Lexicographic code
t	Trivial or explicitly stated code
u	Code from unrestricted code
x	Juxtaposition

Upper bounds	
0	Theorem 1, $i = 0$
1	Theorem 1, $i = 1$
2	Theorem 1, $i = 2$
d	Code with $d = 2w_1 + 2w_2$ [10, Theorem 3.3]
w	$A_3(n, d, [w_0, w_1, w_2]) \leq A_3(n, d, n - w_i)$, $i = 0, 1, 2$.
x	Exhaustive search [3]

Table 4: Key to Table 2

Lower bounds	
b	Found by Bogdanova [2]
c	Cyclic code
d	Code with $d = 2w$ or $d = 2w - 1$
g	Partitioning bound for $d = 3$ [9]
j	Code from Jacobsthal matrix
l	Lexicographic code
p	Perfect codes constructed in [8]
s	Shortening of longer code
u	Code from unrestricted code
w	Constant-composition code

Upper bounds	
0	Johnson bound [10, Theorem 3.11]
1	Johnson bound [10, Theorem 3.13]
3	Johnson bound, [10, Theorem 3.15]
4	Johnson bound, [10, Theorem 3.16]
d	Code with $d = 2w$ [10, Theorem 3.3] or $d = 2w - 1$ [11]
l	Linear programming bound [12]
t	Code with $A_3(n, d, w) = 2$ [10, Theorem 3.6]
w	$A_3(n, d, w) \leq A_3(n, d)$
x	Explicit proof $A_3(5, 3, 3) \leq 12$ [10, Theorem 3.17] and $A_3(6, 5, 5) = 3$

Table 5: Key to Table 3

On the Quaternary [18, 9, 8] Code

Jonas Olsson *
Department of Electrical Engineering
Linköpings universitet, S-581 83 Linköping, Sweden
jonoh@isy.liu.se

Abstract

It is proved that the extremal quaternary Hermitian self-dual code of length 18 is up to equivalence the only [18, 9, 8] quaternary code. As a by-product we give a characterization of all quaternary [14, 5, 8], [15, 6, 8], [16, 7, 8], and [17, 8, 8] codes.

1 Introduction

Let F_q^n denote the n-dimensional vector space over the Galois field $F_q = GF(q)$. An $[n, k, d; q]$ code $\mathcal{C}$ is a k-dimensional subspace of F_q^n having minimum distance d. The weight distribution of $\mathcal{C}$ will be denoted by the set $\{A_i\}_{0 \le i \le n}$. In the sequel all codes will be quaternary unless otherwise stated; an $[n, k, d; 4]$ code will be denoted $[n, k, d]$. The elements of the field $F = F_4$ will be denoted $0, 1, \omega, \omega^2$, where $\omega^2 = \omega + 1$.

MacWilliams *et.al.* [1] constructed an $[18, 9, 8]$ code as an extended cyclic code. The same code was also constructed by Pless [3]. This code is of particular interest since it is an *extremal Hermitian* self-dual code; it is a self-dual code under the *Hermitian* inner product with minimum distance $d = 2\lfloor n/6 \rfloor + 2$ (cf. [1] and [2]). In [1] this code is denoted S_{18} and in [4] and [5] it was proved that S_{18} is the only extremal Hermitian self-dual code which has a nontrivial automorphism of odd order. The code S_{18} has also been thoroughly investigated in [6]. In [7] Huffman proved that S_{18} is the only extremal Hermitian self-dual code of length 18. It remained undecided whether there are more $[18, 9, 8]$ codes. If so they are certainly not Hermitian self-dual.

Definition 1 *Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be two linear codes over F_q. Then $\mathcal{C}_1$ and $\mathcal{C}_2$ are said to be equivalent if $\mathcal{C}_1$ can be obtained from $\mathcal{C}_2$ via a sequence of transformations of the following types:*

(i) permutation of the coordinate positions;

(ii) multiplication of the elements in a given position by a non-zero element of F_q;

*This work was supported by the Swedish Research Counsil for Engineering Sciences under grant 271-97-554

(iii) application of a field automorphism to the elements in all coordinate positions.

We note that the transformations in Definition 1 are *distance preserving* (with respect to Hamming distance) and are such that linear codes are carried to linear codes.

The *support of a vector* is the set of non-zero coordinates. The *support of a code* is the union of supports of the codewords in the code. Let C be an $[n,k,d;q]$ code over F_q. The i-th *generalized Hamming weight* d_i of C is the minimum size of the support of an i-dimensional subcode of C (cf. [8]). In [9] a linear $[n,k,d;q]$ code over F_q is said to be *near*-MDS if $d_1 = d = n-k$ and $d_i = n-k+i,\ i = 2,3,...,k$. A linear $[n,k,d;q]$ code over F_q is said to be *near-near*-MDS if $d_1 = d = n-k-1, d_2 = n-k+1$ and $d_i = n-k+i,\ i = 3,4,...,k$ (cf. [10]).

Theorem 1 ([10]) *If $k > q > 3$ and $n > 2q-1+k$ then every $[n,k,n-k-1;q]$ code over F_q is near-near-MDS.*

We note immediately that quaternary codes with parameters $[14,5,8]$, $[15,6,8]$, $[16,7,8]$, $[17,8,8]$ and $[18,9,8]$ all are near-near-MDS codes. Furthermore, we have the following

Proposition 1 *All quaternary codes with parameters $[14,5,8]$, $[15,6,8]$, $[16,7,8]$, $[17,8,8]$ and $[18,9,8]$ contain a 2-dimensional subcode which is equivalent to the two times repeated Simplex code.*

Proof: Let C be a $[13+i, 4+i, 8], i = 1,2,3,4,5$ code. By Theorem 1 we have $d_2 = 10$ and C contains a 2-dimensional subcode with support of size 10. By deleting the coordinates not in the support we obtain a $[10, 2, \geq 8]$ code. Since, by the Griesmer bound, no $[10,2,9]$ MDS code exists the code has minimum distance 8. Thus, C has parameters $[10,2,8]$ and is - by a result of Bonisoli [11] - the two times repeated Simplex code. ◇

2 Classification result

The (unique) $[10,2,8]$ two times repeated two-dimensional Simplex code has one generator matrix of the form:

$$(I_2|P) = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & \omega & \omega^2 & 1 & \omega & \omega^2 & 0 & 1 \end{pmatrix}, \tag{1}$$

where I_2 is the 2×2 identity matrix and P is a 2×8 matrix.

Proposition 2 *Let C be a $[11+i, 2+i, 8], i = 1,2,...,7$ code which contains a 2-dimensional subcode which is equivalent to the two times repeated two-dimensional Simplex code. Then one can find a generator matrix G of a code equivalent to C of the form*

$$G_{11+i} = \left(I_{2+i} \mid y^T \mid \begin{matrix} P \\ X \end{matrix} \right),$$

where T denote transposition, y is the vector $(0,0,1,1,...,1) \in F^{2+i}$, P is the matrix in (1) and X is a $i \times 8$ matrix.

Proof: By assumption, $\mathcal{C}$ contains a 2-dimensional subcode which is equivalent to the two times repeated two-dimensional Simplex code. This proves the first two rows of G_{11+i}. Hence, we can assume that $y = (0, 0, y_1, y_2, ..., y_i)$. We will prove that $y_j \neq 0$ for all $j = 1, 2, ..., i$. Suppose, on the contrary, that $y_j = 0$ for some $j \in \{1, 2, ..., i\}$. Then the matrix obtained by deleting the zero-columns in the $3 \times (11 + i)$ matrix consisting of the first, second and the j-th row of G_{11+i} generates a $[11, 3, \geq 8]$ code. But, according to Brouwer [12] such a code does not exist. Hence, $y_j \neq 0$ for all $j \in \{1, 2, ..., i\}$ and by Definition 1 we can assume $y_j = 1$ for all $j \in \{1, 2, ..., i\}$. $\diamond$

By Proposition 1 and Proposition 2 a characterization of $[18, 9, 8]$ codes can be accomplished by characterizing all $[11+i, 2+i, 8], i = 1, 2, ..., 8$ codes, starting from $i = 1$, having a generator matrix of the form G_{11+i} in Proposition 2. By Proposition 1 for $i = 3, 4, 5, 6, 7$ this also provides a complete classification.

Any $[12, 3, 8]$ code that contains (a 2-dimensional subcode which is equivalent to) the two times repeated two-dimensional Simplex code is equivalent to a code with generator matrix

$$G_{12} = \left(\begin{array}{c|c|c} I_3 & \begin{matrix} 0 \\ 0 \\ 1 \end{matrix} & \begin{matrix} P \\ x \end{matrix} \end{array} \right),$$

where P is the matrix in (1) and $x = (x_1, x_2, ..., x_8) \in F^8$. We will search for all solutions for x such that G_{12} generates a $[12, 3, 8]$ code and then determine the equivalence classes.

Remark 1 *All searches and determinations of equivalence classes have been accomplished using the computer package MAGMA V2.10-2.*

The search for the possible vectors x can be reduced considerably using the following observation: by Definition 1 we may assume that the first non-zero entry in x to be a 1. Furthermore, since there can be at most two zeros in x it suffices to consider only three cases, namely:

1. $x = (0, 0, 1, x_4, x_5, x_6, x_7, x_8)$;
2. $x = (0, 1, x_3, x_4, x_5, x_6, x_7, x_8)$;
3. $x = (1, x_2, x_3, x_4, x_5, x_6, x_7, x_8)$.

The search complexity can be further reduced using the following observation: suppose that values for $x_1, x_2, ..., x_i$ have been determined. Let $\mathcal{C}(i)$ be the code with generator matrix having as columns the first $i+4$ columns of G_{12}. If there exists a possible solution for $x_{i+1}, x_{i+2}, ..., x_8$ then $n - (i + 4) \geq 8 - w$ must be satisfied, where w denotes the minimum distance of $\mathcal{C}(i)$. This idea will also be used later, when we deal with other matrices than G_{12}, in order to reduce the search complexity.

These observations give 3513 possibilities for x. It turns out that there are only 8 different weight distributions, but, in fact, 9 equivalence classes. In terms of x one representative for each one of the 9 equivalence classes is listed in the table below. In the last five columns we list also the corresponding weight enumerators A_8, A_9, A_{10}, A_{11} and A_{12}.

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	A_8	A_9	A_{10}	A_{11}	A_{12}
(I)	1	ω^2	ω	1	ω^2	ω	0	0	45	0	0	0	18
(II)	1	ω^2	ω	ω	ω	1	1	1	30	0	30	0	3
(III)	0	1	ω^2	ω	1	0	1	ω^2	33	0	24	0	6
(IV)	1	0	ω	ω	ω^2	ω	0	1	36	0	18	0	9
(V)	1	ω^2	1	ω	0	ω	ω^2	1	21	24	12	0	6
(VI)	1	0	1	1	ω^2	0	ω^2	1	30	12	6	12	3
(VII)	1	ω^2	0	0	ω	1	1	ω^2	27	12	12	12	0
$(VIII)$	1	0	ω^2	0	ω^2	ω	ω^2	ω^2	24	18	12	6	3
(IX)	0	1	ω	ω^2	ω	1	ω	1	24	18	12	6	3

Similarly, we state that any $[13,4,8]$ code that contains the two times repeated two-dimensional Simplex code is equivalent to a code with generator matrix

$$G_{13} = \left(\begin{array}{c|c|c} & 0 & P \\ & 0 & x \\ I_4 & 1 & \\ & 1 & y \end{array} \right),$$

where x is one of the nine possibilities $(I)-(IX)$ and $y = (y_1, y_2, ..., y_8) \in F^8$. For each one of nine possibilities we find the possibilities for y. All in all we find 5570 solutions for G_{13}. These are divided into 41 equivalence classes. Hence, there are exactly 41 $[13,4,8]$ codes that contain the two times repeated two-dimensional Simplex code. We find by computer that only 11 of them can be embedded in a $[14,5,8]$ code. Those are given in the table below in terms of y. In the second column the corresponding entry ((I)-(IX)) for x is given.

	x	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	A_8	A_9	A_{10}	A_{11}	A_{12}	A_{13}
(i)	(I)	1	1	ω^2	0	0	ω	ω^2	1	69	48	48	48	42	0
(ii)	(I)	1	ω^2	ω^2	ω	1	1	ω^2	1	81	0	120	0	54	0
(iii)	(III)	1	ω^2	0	ω	ω	ω	0	ω^2	81	0	120	0	54	0
(iv)	(III)	ω^2	ω^2	ω	ω^2	ω	1	ω	ω	81	0	120	0	54	0
(v)	(III)	1	ω^2	1	ω	ω	ω	1	1	81	0	120	0	54	0
(vi)	(III)	ω	0	1	1	1	ω^2	0	ω^2	81	0	120	0	54	0
(vii)	(III)	0	ω	ω^2	ω^2	0	ω	ω^2	ω	63	54	60	36	36	6
$(viii)$	(IV)	0	ω^2	ω	1	1	0	1	ω	66	51	54	42	39	3
(ix)	(IV)	1	1	0	ω	0	1	ω^2	ω	81	0	120	0	54	0
(x)	(IV)	ω^2	ω	ω^2	1	ω^2	ω	ω	ω^2	81	0	120	0	54	0
(xi)	$(VIII)$	ω	1	ω^2	ω	1	ω^2	0	0	69	48	48	48	42	0

By Proposition 1 any $[14,5,8]$ code contains the two times repeated two-dimensional Simplex code. Hence, any $[14,5,8]$ code is equivalent to a code with generator matrix

$$G_{14} = \left(\begin{array}{c|c|c} & 0 & P \\ & 0 & x \\ I_5 & 1 & \\ & 1 & y \\ & 1 & z \end{array} \right),$$

where x, y is one of the 11 possibilities $(i)-(xi)$ and $z = (z_1, z_2, ..., z_8) \in F^8$. We search for the possibilities for z and find 99 solutions for G_{14}. Among those

solutions there are exactly 9 generator matrices $G_{14}(1) - G_{14}(9)$ yielding non-equivalent $[14, 5, 8]$ codes. Those are given in the in table in the Appendix. The weight distributions of the corresponding codes are given by the table below

	A_8	A_9	A_{10}	A_{11}	A_{12}	A_{13}	A_{14}
$G_{14}(1)$	153	144	192	192	270	48	24
$G_{14}(2)$	153	144	192	192	270	48	24
$G_{14}(3)$	147	156	198	198	276	60	18
$G_{14}(4)$	189	0	420	0	378	0	36
$G_{14}(5)$	189	0	420	0	378	0	36
$G_{14}(6)$	189	0	420	0	378	0	36
$G_{14}(7)$	189	0	420	0	378	0	36
$G_{14}(8)$	189	0	420	0	378	0	36
$G_{14}(9)$	189	0	420	0	378	0	36

We can state the following.

Theorem 2 *There exist 9 (up to equivalence) quaternary* $[14, 5, 8]$ *codes.*

We have checked that only the codes with the generator matrices $G_{14}(4) - G_{14}(9)$ can be embedded in an $[15, 6, 8]$ code.

Next, we search for generator matrices of $[15, 6, 8]$ codes of the form

$$G_{15} = \begin{pmatrix} I_6 & G_0^{(1)} \\ & v \end{pmatrix},$$

where $(I_5 \mid G_0^{(1)})$ has to be one of $G_{14}(4) - G_{14}(9)$ and $v = (1, w), w = (w_1, w_2, ..., w) \in F^8$. We find that there are 25 solutions for G_{15} which are divided into only three equivalence classes. Thus, we have

Theorem 3 *There exist three (up to equivalence) quaternary* $[15, 6, 8]$ *codes; all three with non-zero weights:*

$$A_0 = 1, A_8 = 405, A_{10} = 1260, A_{12} = 1890, A_{14} = 540.$$

One generator matrix for each one these three codes in Theorem 3 is given by the matrices $G_{15}(1) - G_{15}(3)$ below.

$$G_{15}(1) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & \omega & \omega^2 & 1 & \omega & \omega^2 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & \omega^2 & \omega & 1 & 0 & 1 & \omega^2 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & \omega^2 & 0 & \omega & \omega & \omega & 0 & \omega^2 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & \omega & 1 & \omega^2 & 0 & \omega^2 & 1 & \omega & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & \omega^2 & \omega^2 & 1 & \omega & 0 & 0 & \omega^2 & 1 \end{pmatrix};$$

$$G_{15}(2) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & \omega & \omega^2 & 1 & \omega & \omega^2 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & \omega & \omega & \omega^2 & \omega & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & \omega^2 & \omega & \omega^2 & 1 & \omega^2 & \omega & \omega & \omega^2 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & \omega^2 & 0 & 0 & 1 & \omega & 1 & \omega^2 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & \omega & 1 & \omega^2 & \omega^2 & \omega^2 & 1 & \omega \end{pmatrix};$$

$$G_{15}(3) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & \omega & \omega^2 & 1 & \omega & \omega^2 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & \omega^2 & \omega & 1 & 0 & 1 & \omega^2 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & \omega & 0 & 1 & 1 & 1 & \omega^2 & 0 & \omega^2 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & \omega & 0 & 0 & \omega^2 & \omega & 1 & \omega & \omega \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & \omega^2 & \omega^2 & \omega & \omega & 1 & \omega^2 & \omega & \omega \end{pmatrix}.$$

Although the three $[15,6,8]$ codes given by the matrices $G_{15}(1) - G_{15}(3)$ have equal weight distribution only the the codes given by the matrices $G_{15}(2)$ and $G_{15}(3)$ can be embedded in an $[16,7,8]$ code. So, we search for generator matrices of $[16,7,8]$ codes of the form

$$G_{16} = \begin{pmatrix} I_7 & G_0^{(2)} \\ & v \end{pmatrix},$$

where $(I_6|G_0^{(2)})$ has to be either $G_{15}(2)$ or $G_{15}(3)$ and $v = (1,w), w = (w_1, w_2, ..., w_8) \in F_8$. The search gave 6 solutions for G_{16} which are divided into two equivalence classes. This results in the following theorem.

Theorem 4 *There exist two (up to equivalence) quaternary* $[16,7,8]$ *codes; both with non-zero weights*

$$A_0 = 1, A_8 = 810, A_{10} = 3360, A_{12} = 7560, A_{14} = 4320, A_{16} = 333.$$

One generator matrix for these two codes in Theorem 4 is given by the matrices $G_{16}(1)$ and $G_{16}(2)$ below.

$$G_{16}(1) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \omega & \omega^2 & 1 & \omega & \omega^2 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & \omega^2 & \omega & 1 & 0 & 1 & \omega^2 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & \omega^2 & 0 & \omega & \omega & \omega & 0 & \omega^2 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & \omega & 1 & \omega^2 & 0 & \omega^2 & 1 & \omega & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & \omega^2 & \omega^2 & 1 & \omega & 0 & 0 & \omega^2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & \omega & 1 & \omega^2 & 1 \end{pmatrix};$$

$$G_{16}(2) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \omega & \omega^2 & 1 & \omega & \omega^2 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & \omega & \omega & \omega^2 & \omega & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & \omega^2 & \omega & \omega^2 & 1 & \omega^2 & \omega & \omega & \omega^2 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & \omega^2 & 0 & 0 & 1 & \omega & 1 & \omega^2 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & \omega & 1 & \omega^2 & \omega^2 & \omega^2 & 1 & \omega \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & \omega & 0 & 1 & 1 & \omega^2 & 1 \end{pmatrix}.$$

We continue and search for generator matrices of $[17,8,8]$ codes of the form

$$G_{17} = \begin{pmatrix} I_8 & G_0^{(3)} \\ & v \end{pmatrix},$$

where $(I_7|G_0^{(3)})$ has to be either $G_{16}(1)$ or $G_{16}(2)$ and $v = (1,w), w = (w_1, w_2, ..., w_8) \in F^8$. The search gave 4 solutions for G_{17} divided into two pairs of equivalent codes.

Theorem 5 *There exist two (up to equivalence) quaternary* $[17, 8, 8]$ *codes; both with non-zero weights*

$$A_0 = 1, A_8 = 1530, A_{10} = 8160, A_{12} = 25704, A_{14} = 24480, A_{16} = 5661.$$

One generator matrix for these two codes in Theorem 5 is given by the matrices $G_{17}(1)$ and $G_{17}(2)$ below.

$$G_{17}(1) = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \omega & \omega^2 & 1 & \omega & \omega^2 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & \omega^2 & \omega & 1 & 0 & 1 & \omega^2 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & \omega^2 & 0 & \omega & \omega & \omega & 0 & \omega^2 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & \omega & 1 & \omega^2 & 0 & \omega^2 & 1 & \omega & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & \omega^2 & \omega^2 & 1 & \omega & 0 & 0 & \omega^2 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & \omega & 1 & \omega^2 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & \omega^2 & \omega^2 & \omega & \omega^2 & \omega & 1 & \omega & \omega
\end{pmatrix};$$

$$G_{17}(2) = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \omega & \omega^2 & 1 & \omega & \omega^2 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & \omega^2 & \omega & 1 & 0 & 1 & \omega^2 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & \omega^2 & 0 & \omega & \omega & \omega & 0 & \omega^2 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & \omega & 1 & \omega^2 & 0 & \omega^2 & 1 & \omega & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & \omega^2 & \omega^2 & 1 & \omega & 0 & 0 & \omega^2 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & \omega & 1 & \omega^2 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & \omega^2 & \omega & \omega^2 & 0 & 0 & \omega^2 & 1 & \omega
\end{pmatrix}.$$

Finally, we search for generator matrices of $[18, 9, 8]$ codes of the form

$$G_{18} = \begin{pmatrix} I_9 & \begin{matrix} G_0^{(4)} \\ v \end{matrix} \end{pmatrix},$$

where $(I_8|G_0^{(4)})$ has to be either $G_{17}(1)$ or $G_{17}(2)$ and $v = (1, w), w = (w_1, w_2, \ldots, w_8) \in F^8$. The search gave two generator matrices which, however, yield equivalent codes. Hence, we have

Theorem 6 *There exists one (up to equivalence) quaternary* $[18, 9, 8]$ *code with non-zero weights*

$A_0 = 1, A_8 = 2754, A_{10} = 18360, A_{12} = 77112, A_{14} = 110160, A_{16} = 50949, A_{18} = 2808.$

One generator matrix of the unique $[18, 9, 8]$ code is given by

$$G_{18} = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \omega & \omega^2 & 1 & \omega & \omega^2 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & \omega^2 & \omega & 1 & 0 & 1 & \omega^2 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & \omega^2 & 0 & \omega & \omega & \omega & 0 & \omega^2 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & \omega & 1 & \omega^2 & 0 & \omega^2 & 1 & \omega & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & \omega^2 & \omega^2 & 1 & \omega & 0 & 0 & \omega^2 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & \omega & 1 & \omega^2 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & \omega^2 & \omega & \omega^2 & 0 & 0 & \omega^2 & 1 & \omega \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & \omega^2 & \omega^2 & \omega & \omega^2 & \omega & 1 & \omega & \omega
\end{pmatrix}.$$

In view of the above theorem we can give the following strengthening of a result by Huffman (see Theorem 2.1 [7]).

Corollary 1 *Let* C *be an* $[18, 9, 8]$ *quaternary code, then* C *is equivalent to* S_{18}.

3 Appendix: Table of representatives of $[14, 5, 8]$ codes.

i	$G_{14}(i)$
1	$1\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0$ $0\ 1\ 0\ 0\ 0\ 0\ 1\ \omega\ \omega^2\ 1\ \omega\ \omega^2\ 0\ 1$ $0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ \omega^2\ 0\ \omega^2\ \omega\ \omega^2\ \omega^2$ $0\ 0\ 0\ 1\ 0\ 1\ \omega\ 1\ \omega^2\ \omega\ 1\ \omega^2\ 0\ 0$ $0\ 0\ 0\ 0\ 1\ 1\ \omega\ \omega\ 0\ 1\ \omega^2\ \omega^2\ 1\ \omega^2$
2	$1\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0$ $0\ 1\ 0\ 0\ 0\ 0\ 1\ \omega\ \omega^2\ 1\ \omega\ \omega^2\ 0\ 1$ $0\ 0\ 1\ 0\ 0\ 1\ 1\ \omega^2\ \omega\ 1\ \omega^2\ \omega\ 0\ 0$ $0\ 0\ 0\ 1\ 0\ 1\ 1\ 1\ \omega^2\ 0\ 0\ \omega\ \omega^2\ 1$ $0\ 0\ 0\ 0\ 1\ 1\ 0\ \omega^2\ 1\ \omega^2\ \omega\ \omega^2\ \omega\ \omega$
3	$1\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0$ $0\ 1\ 0\ 0\ 0\ 0\ 1\ \omega\ \omega^2\ 1\ \omega\ \omega^2\ 0\ 1$ $0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ \omega\ \omega\ \omega^2\ \omega\ 0\ 1$ $0\ 0\ 0\ 1\ 0\ 1\ 0\ \omega^2\ \omega\ 1\ 1\ 0\ 1\ \omega$ $0\ 0\ 0\ 0\ 1\ 1\ \omega\ \omega^2\ 0\ \omega^2\ 1\ 1\ \omega\ \omega^2$
4	$1\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0$ $0\ 1\ 0\ 0\ 0\ 0\ 1\ \omega\ \omega^2\ 1\ \omega\ \omega^2\ 0\ 1$ $0\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ \omega^2\ \omega\ 1\ 0\ 1\ \omega^2$ $0\ 0\ 0\ 1\ 0\ 1\ 1\ \omega^2\ 0\ \omega\ \omega\ \omega\ 0\ \omega^2$ $0\ 0\ 0\ 0\ 1\ 1\ \omega\ 1\ \omega^2\ 0\ \omega^2\ 1\ \omega\ 0$
5	$1\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0$ $0\ 1\ 0\ 0\ 0\ 0\ 1\ \omega\ \omega^2\ 1\ \omega\ \omega^2\ 0\ 1$ $0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ \omega\ \omega\ \omega^2\ \omega\ 0\ 1$ $0\ 0\ 0\ 1\ 0\ 1\ \omega^2\ \omega\ \omega^2\ 1\ \omega^2\ \omega\ \omega\ \omega^2$ $0\ 0\ 0\ 0\ 1\ 1\ 1\ \omega^2\ 0\ 0\ 1\ \omega\ 1\ \omega^2$
6	$1\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0$ $0\ 1\ 0\ 0\ 0\ 0\ 1\ \omega\ \omega^2\ 1\ \omega\ \omega^2\ 0\ 1$ $0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ \omega\ \omega\ \omega^2\ \omega\ 0\ 1$ $0\ 0\ 0\ 1\ 0\ 1\ \omega^2\ \omega\ \omega^2\ 1\ \omega^2\ \omega\ \omega\ \omega^2$ $0\ 0\ 0\ 0\ 1\ 1\ 1\ \omega\ 1\ \omega^2\ \omega^2\ \omega^2\ 1\ \omega$
7	$1\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0$ $0\ 1\ 0\ 0\ 0\ 0\ 1\ \omega\ \omega^2\ 1\ \omega\ \omega^2\ 0\ 1$ $0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ \omega\ \omega\ \omega^2\ \omega\ 0\ 1$ $0\ 0\ 0\ 1\ 0\ 1\ \omega^2\ \omega\ \omega^2\ 1\ \omega^2\ \omega\ \omega\ \omega^2$ $0\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ \omega\ 1\ \omega\ \omega^2\ \omega\ \omega$
8	$1\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0$ $0\ 1\ 0\ 0\ 0\ 0\ 1\ \omega\ \omega^2\ 1\ \omega\ \omega^2\ 0\ 1$ $0\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ \omega^2\ \omega\ 1\ 0\ 1\ \omega^2$ $0\ 0\ 0\ 1\ 0\ 1\ \omega\ 0\ 1\ 1\ 1\ \omega^2\ 0\ \omega^2$ $0\ 0\ 0\ 0\ 1\ 1\ 1\ 0\ \omega\ 0\ \omega^2\ \omega^2\ \omega^2\ 1$
9	$1\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0$ $0\ 1\ 0\ 0\ 0\ 0\ 1\ \omega\ \omega^2\ 1\ \omega\ \omega^2\ 0\ 1$ $0\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ \omega^2\ \omega\ 1\ 0\ 1\ \omega^2$ $0\ 0\ 0\ 1\ 0\ 1\ \omega\ 0\ 1\ 1\ 1\ \omega^2\ 0\ \omega^2$ $0\ 0\ 0\ 0\ 1\ 1\ \omega\ 0\ 0\ \omega^2\ \omega\ 1\ \omega\ \omega$

References

[1] F.J. MacWilliams, A.M. Odlyzko, N.J.A. Sloane and H.N. Ward, *Self-dual codes over GF(4)*, J. Combin. Theory Ser. A 25 (1978) 288-318.

[2] J.H. Conway and N.J.A. Sloane, *Sphere Packings, Lattices and Groups* (Springer-Verlag, New York, 1988).

[3] V. Pless, *Q-codes*, J. Combin. Theory Ser. A 43 (1986) 258-276.

[4] W.C. Huffman, *On extremal self-dual quaternary codes of length 18 to 28, I*, IEEE Trans. Inform. Theory, vol. 36, pp. 651-660, 1990.

[5] W.C. Huffman, *On extremal self-dual quaternary codes of length 18 to 28, II*, IEEE Trans. Inform. Theory, vol. 37, pp. 1206-1216, 1991.

[6] Y. Cheng and N.J.A. Sloane, *The automorphism group of an* $[18, 9, 8]$ *quaternary code*, Discr. Math., vol. 83, pp. 205-212, 1990.

[7] W.C. Huffman, *Characterization of Quaternary Extremal Codes of Lengths 18 and 20*, IEEE Trans. Inform. Theory, vol. 43, pp. 1613-1616, 1997.

[8] V.K. Wei, *Generalized Hamming weights for linear codes*, IEEE Trans. Inform. Theory, vol. 37, pp. 1412-1418, 1991.

[9] S.M. Dodunekov and I.N. Landgev, *On Near-MDS codes*, Journal of Geometry, vol. 54, pp. 30-43, 1995.

[10] J. Olsson, *On Near-Near-MDS Codes*, Fifth International Workshop on Algebraic and Combinatorial Coding Theory, pp. 231-236, June 1 - June 7, 1996, Sozopol, Bulgaria.

[11] A. Bonisoli, *Every equidistant linear code is a sequence of dual Hamming codes*, Ars Combinatoria, vol. 18, pp. 181-186, 1983.

[12] A.E. Brouwer, *Bounds on the minimum distance of linear codes*, [Online], Eindhoven University of Technology, Eindhoven, The Netherlands, Available: http://www.win.tue.nl/math/dw/voorlincod.html.

Spherical codes generated by weighted unions

Thomas Ericson
University of Linköping, Sweden

Victor Zinoviev
Institute for Problems of Information Transmission,
Moscow, Russia

November 27, 1998

Abstract

Using the principle of weighted unions two new spherical codes are constructed. From these constructions we obtain new lower bounds on the kissing number in dimensions 13 and 14.

keywords: spherical codes, contact number, kissing number.

1 Introduction

A spherical code is a finite pointset on the unit sphere in n-dimensional Euclidean space. We characterize such a code by a triple (n, ρ, M) of parameters, where n is called *dimension*, ρ is called *squared minimum distance*, and M is called *cardinality*. The dimension n equals the dimension of the smallest Euclidean space containing the code. The squared minimum distance ρ is the square of the smallest Euclidean distance bewteen two distinct points in the code. The cardinality, finally, is simply the number of points in the code. The points are usually referred to as *codewords*. Given dimension n and squared distance ρ we are interested in finding spherical codes with largest possible cardinality M. Clearly this number depends on the squared distance ρ. The

case $\rho = 1$ is of special interest. In that case the maximal cardinality is referred to as the *kissing number* (or *contact number*), and is usually denoted as τ_n. The following cases are known: $\tau_1 = 2, \tau_2 = 6, \tau_3 = 12, \tau_8 = 240$, and $\tau_{24} = 196560$. In all other cases only upper and lower bounds are known. Leech and Sloane [5] in 1971 derived lower and upper bounds for contact number τ_n in the region $n \leq 24$. In their approach they used error-correcting codes. In 1988 Conway and Sloane ([2], p. 23) republished this table with only one change for lower bounds. In the present paper we improve the lower bounds in two cases, providing the new bounds: $\tau_{13} \geq 1154$ and $\tau_{14} \geq 1606$. Except for the single improvement offered in [2] these results are the first improvements since the paper [5] of Leech and Sloane in 1971.

2 The Y_k-constructions

In [3], [4] we described a class of spherical codes generated from binary codes. The constructions are denoted Y_k, and there is one such construction for each integer $k \geq 2$. Although simple these codes include several cases of best known spherical codes. We also demonstrated that good spherical codes can often be obtained by weighted unions of codes obtained from the Y_k-constructions. We now use that approach for constructing the codes generating the new lower bounds for contact number in dimensions $n = 13, 14$.

The case Y_2 is simply the well known binary antipodal representation of a binary code. We use the symbolic notation $[(\pm\alpha)^n]$ to indicate that each codeword is a real vector with n components, each taking the value α or $-\alpha$. Clearly we must have $n\alpha^2 = 1$. It is easily established that for a binary code A of length n and with minimum Hamming distance d the resulting spherical code has minimum squared Euclidean distance $\rho = 4d/n$. The cardinality of the spherical code equals - of course - the cardinality of the binary code.

The construction Y_3 employs two binary codes: a constant weight code C and an unrestricted code A. The length of the unrestricted code A equals the weight of the constant weight code C. If the constant weight code C has length n and weight w we use the symbolic notation $[0^{n-w}, (\pm\alpha)^w]$ for the resulting spherical code. This notation indicates that each codeword is a real vector of length n, containing 0 in $n-w$ positions and $\pm\alpha$ in the w remaining ones. For each pair $(c, a) \in C \times A$ there is a codeword x in the spherical code such that x has a 0 in each one of the positions where c has a 0 and $\pm\alpha$ in

each of the positions where c contains a 1. The signs are determined by the components in the binary codeword $a \in A$. It is easily established that the squared minimum Euclidean distance ρ of the spherical code X satisfies

$$\rho \geq \min \{\frac{d_C}{w}, \frac{4d_A}{w}\},$$

where d_C and d_A are the minimum Hamming distances in the binary codes C and A respectively.

We use the symbolic notation $[0^i, (\pm\alpha)^j | \pm\beta]$ to denote a spherical code obtained from the Y_3-construction by a weighted union. This construction uses two spherical codes obtained from the Y_3-construction in such a way that each codeword contains a codeword from one of the basic Y_3-codes appended with a tail, which is $+\beta$ or $-\beta$ depending on which one of the Y_3-codes is used. Here α and β are real numbers selected such that the constraint $j\alpha^2 + \beta^2 = 1$ is satisfied. As a special case one and the same Y_3-code could be used for both tails $\pm\beta$.

A corresponding construction is of course applicable to the Y_2-construction, with a similar symbolic notation for the resulting spherical code. It is also obvious that the idea of weighted union can be extended so as to include more than two constituent spherical codes and more complicated tails. However, for our present purposes the simple special cases indicated above will be sufficient.

3 Two special cases

We are now ready to describe the special constructions leading to the improved bounds on contact number. Consider first the case $n = 13$. We construct a spherical code W as a regular union of four different spherical codes with the following forms:

$$\begin{array}{lcl} W_1: & [0^8, (\pm 1/2)^4 | 0] & (n, \rho, M) = (13, 1, 816) \\ W_2: & [(\pm 1/4)^{12} | \pm 1/2] & (n, \rho, M) = (13, 1, 288) \\ W_3: & [0^{11}, \pm\sqrt{3}/2 | \pm 1/2] & (n, \rho, M) = (13, 1, 48) \\ W_4: & [0^{12} | \pm 1] & (n, \rho, M) = (13, 4, 2). \end{array}$$

The first code W_1 is obtained from the Y_3-construction using the constant weight code C_1 with parameters $(n, w, d, M) = (12, 4, 4, 51)$ (see [1], Table I-

A) and the trivial unrestricted code A_1 with parameters $(n, d, M) = (4, 1, 16)$. To this spherical code a trivial tail consisting of a 0 is added. The second code W_2 is obtained as a weighted union of two identical copies of codes obtained from the Y_2-construction, using the unrestricted binary code A_2 with parameters $(n, d, M) = (12, 4, 144)$ (see [1], Table II). The code W_3 is obtained as a union of two copies of the biorhtogoinal code, generated by the Y_3-construction. The code W_4, finally, is nothing else than the antipodal code, consisting of two points, where the first twelve positions are zero.

Consider the matrix $\mathbf{S} = [s_{ij}]$, with entities s_{ij} defined as

$$s_{ij} = \begin{cases} \max\{(x, y) : \ x, y \in W_i, \ x \neq y\} & \text{if} \quad i = j \\ \max\{(x, y) : \ x \in W_i, \ y \in W_j\} & \text{if} \quad i \neq j, \end{cases}$$

where $i, j = 1, 2, 3, 4$ and (x, y) denotes the inner product of x and y.

Without much effort we compute:

$$\mathbf{S} = \begin{bmatrix} 1/2 & 1/2 & \sqrt{3}/4 & 0 \\ 1/2 & 1/2 & (\sqrt{3}+2)/8 & 1/2 \\ \sqrt{3}/4 & (\sqrt{3}+2)/8 & 1/2 & 1/2 \\ 0 & 1/2 & 1/2 & -1 \end{bmatrix}$$

We clearly get $\max(s_{ij}) = 1/2$, and so we have $\rho(W) = 1$. It follows that the spherical code

$$W = W_1 \bigcup W_2 \bigcup W_3 \bigcup W_4$$

has parameters $(n, \rho, M) = (13, 1, 1154)$.

For the case $n = 14$ we choose four codes as follows:

$$\begin{array}{lcl} W_1: & [0^9, (\pm 1/2)^4 | 0] & (n, \rho, M) = (14, 1, 1040) \\ W_2: & [(\pm 1/4)^{13} | \pm \sqrt{3}/4] & (n, \rho, M) = (14, 1, 512) \\ W_3: & [0^{12}, \pm\sqrt{3}/2 | \pm 1/2] & (n, \rho, M) = (14, 2, 52) \\ W_4: & [0^{13} | \pm 1] & (n, \rho, M) = (14, 4, 2). \end{array}$$

The code W_1 is obtained from the Y_3-construction using the constant weight code C_1 with parameters $(n, w, d, M) = (14, 4, 4, 65)$ ([1], Table I-A) and the trivial unrestricted code A_1 with parameters $(n, d, M) = (4, 1, 16)$. The code W_2 is obtained from the Y_2-construction. In this case we use two disjoint unrestricted codes A_2 and A_2', both with parameters $(n, d, M) =$

$(13, 4, 256)$. They can be obtained by (1)- and (0)-reductions of the code A with parameters $(14, 4, 512)$ ([1], Table II). The two last codes are again obtained from the biorthogonal code and the antipodal code.

Exactly as in the previous case we define a correlation matrix $\mathbf{S}$. In the present case it takes the form

$$\mathbf{S} = \begin{bmatrix} 1/2 & 1/2 & \sqrt{3}/4 & 0 \\ 1/2 & 1/2 & \sqrt{3}/4 & 0 \\ \sqrt{3}/4 & \sqrt{3}/4 & 1/2 & 1/2 \\ 0 & 0 & 1/2 & -1 \end{bmatrix}$$

Again we have $\max(s_{ij}) = 1/2$, and so $\rho(W) = 1$. It follows that the spherical code

$$W = W_1 \bigcup W_2 \bigcup W_3 \bigcup W_4$$

has parameters $(n, \rho, M) = (14, 1, 1606)$.

4 Conclusions

The constructions described above allow us to formulate the following result.

Theorem 1 *For $n = 13, 14$ the contact number τ_n is bounded below as follows*

$$\tau_{13} \geq 1154 \quad \textit{and} \quad \tau_{14} \geq 1606.$$

We notice that these new bounds exceed the bounds given in [5] by 24 in both cases.

References

[1] A.E. Brouwer, J.B. Shearer, N.J.A. Sloane & W.D. Smith, *A new table of constant weight codes*, "IEEE Trans. Inform. Theory", Vol. IT-36, No 6, pp. 1334-1380, November 1990.

[2] J.H. Conway & N.J.A. Sloane, *Sphere Packings, Lattices and Groups*, Sec. Ed., Springer-Verlag, 1993.

[3] TH. ERICSON & V.A. ZINOVIEV, *New packings on Euclidean sphere for finite dimensions*,"Problems Information Transmission", Vol. 28, No. 2, pp. 47-53, 1992.

[4] TH. ERICSON & V.A. ZINOVIEV, *Spherical codes generated by binary partitions of symmetric pointsets*", "IEEE Trans. Inform. Theory", Vol. IT-41, No. 1, pp. 107-129, January 1995.

[5] J. LEECH & N.J.A. SLOANE, *Sphere packings and error-correcting codes*, "Canad. J. Math.", Vol. 23, No. 4, pp. 718-745, 1971.

A New Practical Algorithm for the Construction of a Perfect Hash Function

M. Atici
International Computer Institute
University of Ege, Izmir-Turkey
atici@ube.ege.edu.tr

D. R. Stinson
Department of Computer Science and Engineering
University of Nebraska, Lincoln, NE 68588
stinson@bibd.unl.edu

R. Wei
Department of Mathematics and Statistics
University of Nebraska, Lincoln, NE 68588
rwei@cse.unl.edu

Abstract

A perfect hash function for a subset X of $\{0, 1, \cdots, n-1\}$ is an injection h from X into the set $\{0, 1, \cdots, m-1\}$. Perfect hash functions are useful for the compact storage and fast retrieval of frequently used objects. In this paper, we discuss some new practical algorithms for efficient construction of perfect hash functions, and we analyze their complexity and program size.

Keywords: perfect hash family, perfect hash function, program size, complexity.

1 Introduction

A *hash function* is a function $h : \{1, 2, \cdots, n\} \rightarrow \{1, 2, \cdots, m\}$. A hash function is said to be *perfect* on a subset X of $\{1, 2, \cdots, n\}$ if h is injective on X, i.e., if $h|_X$ is one-to-one. Perfect hash functions are useful for the compact storage and fast retrieval of frequently used data, such as reserved words in programming languages, command names in interactive systems, etc. Let $w = |X|$; then $w \leq m$. When $w = m$, the function h is called a *minimal perfect hash function*. Minimal perfect hash functions have applications in compilers, operating systems, language translation systems, hypertext, hypermedia, file managers, and information retrieval systems. For more information about perfect hash functions and minimal perfect hash functions, readers can consult the recent survey paper [3] and its references.

The purpose of this paper is to present some new practical algorithms for construction of a perfect hash function. The efficiency of the algorithm is measured in three ways. First is the amount of time required to find a hash function h which is perfect on a given subset X. Second is the time required to evaluate a given function h for a given $x \in X$. Third is the amount of memory required to store a description of the function h, i.e., the *program size*. The memory required will be the logarithm of the number of the possible perfect hash functions in the associated *perfect hash family*, as defined below.

Definition 1.1 *An* (n, m, w)-perfect hash family *is a finite set of hash functions* $\mathcal{F}$ *such that*

$$h : A \to B$$

for each $h \in \mathcal{F}$, *where* $|A| = n$ *and* $|B| = m$, *with the property that for any* $X \subseteq A$ *such that* $|X| = w$, *there exists at least one* $h \in \mathcal{F}$ *such that* $h|_X$ *is one-to-one.*

We use the notation PHF$(N; n, m, w)$ to denote an (n, m, w)-perfect hash family with $|\mathcal{F}| = N$. We can think of a PHF$(N; n, m, w)$ as an $N \times n$ array of m symbols, where each row of the array corresponds to one of the functions in the family. This array has the property that, for any subset of w columns, there exists at least one row such that the entries in the w given columns of that row are distinct. We will use this representation in some small examples in the sequel.

Let $N(n, m, w)$ denote the smallest value of N for which a PHF$(N; n, m, w)$ exists. In [4], $N(n, m, w)$ is proved to be $\Theta(\log n)$. However, the proof of [4] is not constructive, and it seems difficult to give explicit constructions that are good asymptotically. Hence, it is interesting to find explicit constructions for PHFs. We use some constructions where N is a polynomial function of $\log n$ (for fixed m and w). Moreover, our constructions have the advantage that they are simple and easy to program.

Our goal is to obtain an algorithm for construction and evaluation of a (minimal) perfect hash function in which the complexity and program size are low, and which also works well in practice.

The rest of this paper is arranged as follows. In Section 2, we describe our constructions of PHFs, both direct and recursive. Then we give algorithms to realize these constructions in Section 3. We will analyze the efficiency of these algorithms in Section 4.

All logarithms in this paper are to the base 2.

2 Constructions

2.1 Direct Constructions

In this section, we give two direct constructions of perfect hash families. These are simple "base" PHFs which will be used as initial families in our main recursive construction.

The first construction is based on a finite affine plane (see [1, Corollary 3.2]). Let q be a prime power such that $q + 1 > \binom{w}{2}$. Consider the array having columns indexed by pairs $(x, y) \in F_q \times F_q$, and rows indexed by $F_q \cup \{-1\}$, where F_q is a finite field with q elements and $-1 \notin F_q$. The entry in row r and column (x, y) is $x \times r + y$ if $r \in F_q$, and x if $r = -1$. It is easy to see that any two different columns have precisely one row of conflict. Since $q + 1 > \binom{w}{2}$, it follows that for any w different columns there will be a row that has w different entries in these w columns. Hence we have the following result.

Theorem 2.1 *Suppose* q *is a prime power and* $q + 1 > \binom{w}{2}$. *Then there exists a PHF*$(q + 1; q^2, q, w)$.

In the first construction, m is $\Omega(w^2)$. We will use another base PHF when $m \leq \binom{w}{2}$. This construction is far from optimal, however, the hash families are very easy to compute, and they are considerably smaller than the trivial construction in which $N = \binom{n}{w}$.

For a subset $X = \{x_1, x_2, \cdots, x_w\}$, where $x_1 < x_2 < \cdots < x_w$, we define a perfect hash function h_X as follows:

$$h_X(x) = \begin{cases} 1 & \text{if } x < x_2 \\ 2i-1 & \text{if } x_{2(i-1)} < x < x_{2i}, \text{ for some } i = 2, 3, \cdots, \lfloor \frac{w}{2} \rfloor \\ 2i & \text{if } x = x_{2i} \text{ for some } i = 1, 2, \cdots, \lfloor \frac{w}{2} \rfloor \\ w & \text{if } x \geq x_w. \end{cases}$$

Since $h_X(x_i) = i$ for $i = 1, 2, \cdots, w$, h_X is perfect on X. Thus the family

$$\{h_X : X \subseteq \{1, 2, \cdots, n\}, |X| = w\}$$

is a PHF. (In fact, it is a minimal PHF.)

Let us determine the number of functions, N, in the family. Observe that h_X depends only on the values $x_2, x_4, \cdots$. First, consider the case where w is even. Denote $t = w/2$. We have that $\{x_2, x_4, \cdots, x_{2t}\} \subseteq \{2, \cdots, n\}$ and $x_{2i} \geq x_{2(i-1)} + 2$. If we define $d_i = x_{2i} - i$ for $1 \leq i \leq t$, then we construct a list of t distinct elements $d_1, \ldots, d_t$ where $\{d_1, \ldots, d_t\} \subseteq \{1, \ldots, n-t\}$. Thus we see that

$$N = \binom{n - \frac{w}{2}}{\frac{w}{2}}$$

in the case where w is even. A similar argument when w is odd establishes the following theorem.

Theorem 2.2 *For any $n > w$, the PHF$(N; n, w, w)$ constructed above has*

$$N = \binom{n - \lceil \frac{w}{2} \rceil}{\lfloor \frac{w}{2} \rfloor}.$$

We mentioned that this construction represents a considerable improvement over the trivial family. For example, if $n = 11$ and $w = 5$, then we have $N = \binom{8}{2} = 28$ as compared to $N = \binom{11}{5} = 462$ in the trivial PHF.

In the case $n = w + 1$, we obtain a PHF$(\lfloor \frac{w}{2} \rfloor + 1; n, w, w)$, which can be shown to be optimal.

2.2 A Recursive Construction

In this section, we describe the recursive construction given in [1]. We begin with a specific type of difference matrix. Suppose that the integers n, w have the property that $\gcd(n, \binom{w}{2}!) = 1$. Let $D = (d_{ij})$, where $d_{ij} = ij \bmod n$ for $0 \leq i \leq \binom{w}{2}$ and $0 \leq j \leq n-1$. This is called an $(n, \binom{w}{2} + 1)$-*difference matrix*, since for all h, i such that $0 \leq h < i \leq \binom{w}{2}$, we have $\{d_{h,j} - d_{i,j} \bmod n : 0 \leq j \leq n-1\} = N_n$. (See [2] for more information about difference matrices.) The following lemma ([1, Theorem 4.1]) gives a recursive construction for PHF that uses difference matrices.

Lemma 2.1 *Suppose there is an $(n_0, \binom{w}{2} + 1)$-difference matrix and a PHF$(N_0; n_0, m, w)$. Then there is a PHF$((\binom{w}{2} + 1)N_0; n_0^2, m, w)$.*

For completeness, we outline the construction. Let A be a PHF$(N_0; n_0, m, w)$ and let $D = (d_{i,j})$ be an $(n_0, \binom{w}{2} + 1)$-difference matrix. For $0 \leq j \leq n_0 - 1$, let A^j denote the array obtained from A by letting the permutation σ^j act on the columns of A, where $\sigma(i) = (i-1) \bmod n_0$. Now let

Algorithm 3.1: Construction of a (q^2, q, w) hash function

Input:
q, w and $\{x_1, x_2, \ldots, x_w\} \subseteq \{0, 1, \cdots, q^2 - 1\}$,
where q is prime, $q + 1 > \binom{w}{2}$

(1) **for** $k := 1$ **to** w **do**
$i_k := \lfloor \frac{x_k}{q} \rfloor$,
$j_k := (x_k - i_k \times q) \bmod q$

(2) **If** all i_k's are different **then** $r := -1$
else find r, where $0 \leq r \leq q - 1$
such that $(i_k \times r + j_k) \bmod q$ are different for $k = 1, ..., w$

(3) the constructed hash function is h_r

$$B = \begin{array}{|c|c|c|c|} \hline B_{0,0} & B_{0,1} & \cdots & B_{0,n_0-1} \\ \hline B_{1,0} & B_{1,1} & \cdots & B_{1,n_0-1} \\ \hline \vdots & \vdots & \vdots & \vdots \\ \hline B_{\binom{w}{2},0} & B_{\binom{w}{2},1} & \cdots & B_{\binom{w}{2},n_0-1} \\ \hline \end{array}$$

where $B_{i,j} = A^{d_{i,j}}, 0 \leq i \leq \binom{w}{2}, 0 \leq j \leq n_0 - 1$. Then B is the desired PHF.

Using the difference matrices constructed above, and iterating Lemma 2.1, we have the following theorem.

Theorem 2.3 *[1] Suppose there exists a PHF$(N_0; n_0, m, w)$, where $gcd(n_0, \binom{w}{2}!) = 1$. Then there is a PHF$((\binom{w}{2} + 1)^j N_0; {n_0}^{2^j}, m, w)$ for any integer $j \geq 1$.*

3 Algorithms for Construction of a Perfect Hash Function

In this section, we describe algorithms which realize the constructions of the previous section. The first two algorithms concern Theorem 2.1. Suppose q is a prime, $q + 1 > \binom{w}{2}$, and $n \leq q^2$. For a given w-subset X of $\{0, 1, \cdots, n - 1\}$, Algorithm 3.1 finds a hash function which is perfect on X and takes on values in $\{0, 1, \cdots, q - 1\}$, and outputs the description of that function. Algorithm 3.2 will use the description of the hash function to evaluate the function for any $x \in \{0, 1, \cdots, n - 1\}$.

Similarly, Algorithms 3.3 and 3.4 realize the construction of Theorem 2.2.

Finally, Algorithms 3.5 and 3.6 realize the recursive construction of Theorem 2.3. Suppose we have a "base" PHF$(N; n_0, m, w)$ and a w-subset $X \subseteq \{0, 1, \cdots, n_0^{2^j} - 1\}$, where $\gcd(n_0, \binom{w}{2}!) = 1$. Algorithm 3.5 finds the hash function, say h, which is perfect on X. Algorithm 3.6 evaluates the hash function h which is found by Algorithm 3.5 at any input $x \in \{0, 1, ..., n - 1\}$, where $n = n_0^{2^j}$. Notice that in Algorithm 3.5, we do not need to store the whole constructed PHF; we only need to store the base PHF.

It is straightforward to combine the previous algorithms to give a general construction. Suppose integers n, m, w, and a w-subset $X \subseteq \{0, 1, \cdots, n - 1\}$ are given. We want to find a hash function which is perfect on X. We can proceed as follows. First we find the

Algorithm 3.2: Evaluate a (q^2, q, w) hash function

Input:
q, w, $x \in \{0, 1, \cdots, q^2 - 1\}$ and $r \in \{-1, 0, \cdots, q - 1\}$,
where q is prime, $q + 1 > \binom{w}{2}$

(1) $z := x$

(2) $i := \lfloor \frac{z}{q} \rfloor$
$j := (z - q \times i) \bmod q$

(3) **If** $r = -1$ **then** $h_r(x) := i$
else $h_C(x) := (i \times r + j) \bmod q$

Algorithm 3.3: Construction of an (n, w, w) hash function

Input: n, w and $\{x_1, x_2, \cdots, x_w\} \subseteq \{0, 1, \cdots, n - 1\}$

(1) Sort the elements $x_1, x_2, \cdots, x_w$ such that $x_1 < x_2 < \cdots < x_w$

(2) Let $t := \lfloor \frac{w}{2} \rfloor$

(3) Define $c_i := x_{2i}$, for $1 \leq i \leq t$

(4) The constructed hash function is denoted as h_C, where
$C := (c_1, c_2, \cdots, c_t)$

Algorithm 3.4: Evaluate an (n, w, w) hash function

Input:
n, w, $x \in \{0, 1, \cdots, n - 1\}$ and $C = (c_1, c_2, \cdots, c_t)$ $(t = \lfloor \frac{w}{2} \rfloor)$

(1) Define $c_0 := 0$ and $c_{t+1} := n + 1$

(2) **For** $i := 1$ to $t + 1$ **do**

(a) **If** $x = c_i$ **then** define $y := 2i$ and exit loop

(b) **If** $c_{i-1} < x < c_i$ **then** define $y := 2i - 1$ and exit loop

(3) Define $h_C(x) := y$

Algorithm 3.5: Construct an (n, m, w) Hash Function, $n = n_0^{2^j}$

Input:
A base PHF$(N; n_0, m, w)$, where $\gcd(n_0, \binom{w}{2}!) = 1$
$X = \{y_1, ..., y_w\} \subseteq \{0, 1, ..., n_0^{2^j} - 1\}$
(1) While $j > 0$ **do**
(a)**for** $k := 1$ **to** w **do**

$$i_k{}^{(j)} := \left\lfloor \frac{y_k}{n_0^{2^{(j-1)}}} \right\rfloor$$

$$x_k{}^{(j)} := [y_k - n_0^{2^{(j-1)}} \times i_k{}^{(j)}] \bmod n_0^{2^{(j-1)}}$$

(b) find d_j, $0 \le d_j \le \binom{w}{2}$, such that $(x_k{}^{(j)} + d_j \times i_k{}^{(j)}) \bmod n_0^{2^{(j-1)}}$ are all different for $k = 1, 2, \cdots, w$.
(c) **for** $k := 1, 2, \cdots, w$ **do**

$$y_k := (x_k{}^{(j)} + d_j \times i_k{}^{(j)}) \bmod n_0^{2^{(j-1)}}$$

(d) $j := j - 1$
(2) Find the hash function h_C in the base PHF$(N; n_0, m, w)$ which is perfect on $\{y_1, ..., y_w\}$
(3) The constructed hash function is denoted h_D where $D = (d_j, d_{j-1}, ..., d_1; C)$

Algorithm 3.6: Evaluate an (n, m, w) Hash Function, $n = n_0^{2^j}$

Input:
$0 \le x \le n_0^{2^j} - 1$
$D = (d_j, ..., d_1; C)$
(1) /* Reduce x to $y \in [0, n_0 - 1]$ by using d_j's */
While $j > 0$ **do**

$$i^{(j)} := \left\lfloor \frac{x}{n_0^{2^{(j-1)}}} \right\rfloor$$

$$l^{(j)} := (x - n_0^{2^{(j-1)}} \times i^{(j)}) \bmod n_0^{2^{(j-1)}}$$

$$x := (l^{(j)} + d_j \times i^{(j)}) \bmod n_0^{2^{(j-1)}}$$

$j := j - 1$
(2) $y := x$
(3) $h_D(x) := h_C(y)$ where $h_C(y)$ is in the base PHF$(N; n_0, m, w)$

smallest prime $q > \binom{w}{2}$, so it follows $\gcd(q, \binom{w}{2}!) = 1$. (Notice that this step is actually a preprocessing step, since it does not require knowing the particular set X that is to be hashed.) Then we use Algorithm 3.5, letting $j = \lceil \log(\log n) - \log(\log n_0) \rceil$, to find the suitable hash function. If $q \leq m$, then step (2) will use a PHF$(q+1; q^2, q, w)$ from Algorithm 3.1 as the base PHF, otherwise it will use a PHF$(\binom{q-\lceil \frac{w}{2} \rceil}{\lfloor \frac{w}{2} \rfloor}; q, w, w)$ from Algorithm 3.3 as the base PHF. (In this second case, it is not necessary that q be prime.) In either case, Algorithm 3.5 will output the description of the hash function.

4 Efficiency of the Algorithms

In this section, we look at the efficiency of our algorithms. We are interested both in the complexity of the algorithms as well as the running time of an actual implementation. We consider the case of minimal perfect hashing, i.e., $m = w$. In this case, we can take q to be the smallest integer such that $\gcd(q, \binom{w}{2}!) = 1$. The value of q is less than w^2 and it can be computed in time polynomial in $\log w$, using the Euclidean algorithm to compute greatest common divisors.

Let's first consider program size. The size of the hash family in Theorem 2.3 is $(\binom{w}{2} + 1)^j N_0$. Here we have that $j \leq \log \log n$ and

$$N_0 < \binom{w^2}{\frac{w}{2}} < (w^2)^{\frac{w}{2}} = w^w.$$

Thus the program size is bounded above by

$$j \log w^2 + \log N_0 < 2 \log w \log \log n + w \log w.$$

It is known that any minimal PHF has program size that is $\Omega(\log \log n + w)$ (see, e.g., [4, p. 129] or [3, p. 9]). So our construction is not much larger than an optimal one.

In the encoding of PHF that we use, we have a sequence at most $\log \log n$ d's (produced by Algorithm 3.5), where $0 \leq d_i \leq q - 1$ for each i. Thus each d requires at most $\log q < 2 \log w$ bits to encode it. The remaining $w \log w$ bits correspond to the list of $\lfloor \frac{w}{2} \rfloor$ c's produced by Algorithm 3.3, where $0 \leq c_i \leq w^2 - 1$ for each i.

The above analysis is a provable, worst-case bound. In practice, however, we usually do not require so much space. This is because the d's are frequently very small and may not require $2 \log w$ bits to encode them. It appears that the program size is more likely to be $O(w \log w + \log \log n)$, since $d_i = 0$ or 1 "most of the time". We will now try to justify this assumption with an informal heuristic argument and some experiments.

We have done some experiments for fixed w values to compute the average number of values that need to be tested to find an acceptable d in step 1(b) of Algorithm 3.5. In these computations, we take $n = 2^{30}$ and randomly create 100 subsets X ($|X| = w$) of $\{0, 1, ..., 2^{30} - 1\}$ for each w. We run the program for each subset X, computing the average number of tested d values. In Table 4.1, columns 1, 2, and 3 record the values of w, q, and j, respectively. The fourth column gives average number of d values checked (over all j iterations). The sixth column gives the computing time in seconds. (We have done these computations with Sun Ultra 1 Model 140 in the International Computer Institute at the University of Ege, Turkey.)

We now derive a heuristic estimate for the number of d values tested (see column 5 of Table 4.1). If we choose k random elements from a set of n elements (repetition allowed), the probability that all k elements will be distinct is

Table 4.1

w	q	j	Aver. num. of d values tested	$j-1+e$	time (in seconds)
3	5	4	4.89	5.72	0.00013
10	47	3	4.72	4.72	0.00033
20	191	2	3.72	3.72	0.00067
50	1229	2	4.13	3.72	0.00248
100	4951	2	3.25	3.72	0.00668
150	11177	2	3.60	3.72	0.01373
200	19913	2	3.92	3.72	0.02298
300	44851	1	2.60	2.72	0.04432
400	79801	1	2.44	2.72	0.07488
500	124753	1	2.81	2.72	0.11518
600	179717	1	2.76	2.72	0.16355
700	244667	1	2.82	2.72	0.21934
800	319601	1	2.68	2.72	0.28208
900	404557	1	2.28	2.72	0.35176
1000	499507	1	2.91	2.72	0.43724

$$p(k,n) = \left(1-\frac{1}{n}\right)\left(1-\frac{2}{n}\right)\cdots\left(1-\frac{k-1}{n}\right) = \prod_{i=1}^{k-1}\left(1-\frac{i}{n}\right). \tag{1}$$

In step 1(b) of iteration l of Algorithm 3.5, we find a value d such that the w values $(x+d\times i)$ mod $q^{2^{l-1}}$ are distinct elements in $Z_{q^{2^{l-1}}}$. If we assume that these w values are randomly distributed, we can estimate the probability that they are distinct using Eq. (1) with $k=w$ and $n=q^{2^{l-1}}$.

For $l>1$, $p(w,q^{2^{l-1}}) \approx 1$ while for $l=1$, we have

$$\begin{aligned} p(w,q) &\approx p(w,w^2/2) = \left(1-\frac{1}{w^2/2}\right)\cdots\left(1-\frac{w-1}{w^2/2}\right) \\ &\approx \prod_{i=1}^{w-1} e^{-(\frac{i}{w^2/2})} \approx e^{-1}. \end{aligned}$$

Here, we are using the approximation $1-x \approx e^{-x}$, which is true when x is a small positive real number. Thus we estimate that the total number of d values tested should be approximately $j-1+e$. These estimates, recorded in column 5 of Table 4.1, are quite close to the actual experimental values obtained in column 4.

Now we briefly analyze the complexity of the algorithms to construct and evaluate a hash function. For simplicity, we use the "uniform cost" model which assumes that any arithmetic operation can be done in $O(1)$ time (see, e.g., [3, p.10]). As above, we are considering the case $w=m$ and we assume that $q<w^2$ has been determined in a preprocessing step.

Step (1) of Algorithm 3.5 requires $O(\log\log n)$ iterations. Each iteration takes time $O(w \log w)$ to test each particular d value (the w numbers in step 1(b) can be sorted to determine if they are distinct). The number of d values that need be considered is $O(w^2)$. Thus step (1) can be accomplished in time $O(w^3 \log w \log\log n)$. Algorithm 3.3 takes time $O(w \log w)$, so the total time is $O(w^3 \log w \log\log n)$. Notice also that, if the heuristic argument presented above is valid, then the time required to construct the hash function is reduced to $O(w \log w \log\log n)$.

The evaluation time is analyzed is a similar fashion. Algorithm 3.6 requires time $O(\log\log n)$, and Algorithm 3.4 can be modified to run in time $O(\log w)$ if a binary search is used. Therefore the total time is $O(\log w + \log\log n)$.

Finally, we should emphasize again that the algorithms are very suitable for practical use, as is shown by the timings in Table 4.1.

Acknowledgment

The research of the second and third authors is supported by NSF Grant CCR-9610138.

References

[1] M. Atici, S. S. Magliveras, D. R. Stinson and W.-D. Wei. Some Recursive Constructions for Perfect Hash Families. *Journal of Combinatorial Designs* 4 (1996), 353–363.

[2] C. J. Colbourn and J. H. Dinitz. *CRC Handbook of Combinatorial Designs*, CRC Press, 1996.

[3] Z. J. Czech, G. Havas and B. S. Majewski. Perfect Hashing, *Theoretical Computer Science* **182** (1997), 1–143.

[4] K. Mehlhorn, *Data Structures and Algorithms 1: Sorting and Searching*, Springer-Verlag, Berlin, 1984.

Iterative Multistage Maximum Likelihood Decoding of Multi-Level Codes*

Diana Stojanović, Marc P.C. Fossorier and Shu Lin [1]
[1]Department of Electrical Engineering, University of Hawaii, Honolulu, Hawaii 96822, USA
E-mail: {dixi,marc,slin}@spectra.eng.hawaii.edu

Abstract

In this paper, an efficient soft-decision iterative multistage decoding algorithm for decoding decomposable and multilevel concatenated codes is presented. This algorithm achieves maximum likelihood decoding (MLD) through iterations with optimality tests at each decoding stage. It achieves MLD performance with a significant reduction in decoding complexity. Decoding results for some long codes are included.

1 Introduction

Multistage decoding (MSD) [1] is devised for decoding decomposable codes and multilevel codes [2] to achieve an efficient trade-off between error performance and decoding complexity. In a conventional soft-decision MSD scheme, each stage uses soft-decision decoding, but hard-decision decoded output is used to modify the soft-decision received sequence for the next stage decoding. Decoded information is passed down from one stage to the next until the last decoding stage. With this decoding, incorrect decoded information at one stage may result in error propagation through the subsequent decoding stages. Therefore, the decoding is not MLD even if every decoding stage is MLD, it is a suboptimum decoding scheme. This decoding works well for short to moderately long codes, say lengths up to 64. It provides excellent trade-off between error performance and decoding complexity. The performance degradation compared with that of MLD is very small, no more than 0.4 dB for decomposable codes of lengths 64 [3]. However, for long codes of lengths 128 and up, the performance degradation becomes severe. For example, the performance degradation of the (128,64) Reed-Muller (RM) code with three-stage decoding (each stage uses the Viterbi decoding) is 1.5dB at bit-error rate (BER) of 10^{-5} [4].

Error performance of MSD can be improved by passing a list of L best decoded estimates from one stage to the next. At the last stage, the codeword with the best metric (largest

*This research was supported by the National Science Foundation under Grants No. NCR-94-15374 and NCR-97-32959, and NASA under Grant NAG 5-931.

correlation, or smallest Euclidean distance) is chosen as the decoded codeword [5]. Of course, the decoding complexity of this multistage list decoding increases multifoldly. For long decomposable codes, in order to maintain the trellis complexity of each component code in a practically implementable range, the number of decoding stages must go up. As a result, the size of the list must increase to maintain a small performance degradation compared to MLD. This results in a large increase of computational complexity. To overcome this problem, a list of variable size depending on the signal-to-noise ratio (SNR) can be used to reduce the average computational complexity.

In this paper, we present an iterative multistage maximum likelihood decoding (IMS-MLD) algorithm which achieves MLD performance through decoding iterations. Each new decoding iteration begins with the generation of a new estimate at a certain decoding stage. This new estimate is then passed down to the subsequent stages of decoding. During a decoding iteration, two simple optimality conditions are tested. If one of the conditions is satisfied, the decoding process is terminated and the best codeword found at the time is the most likely (ML) codeword. Decoding iteration continues until the ML codeword is found. In this IMS-MLD algorithm, a new estimate is generated only when it is needed. Optimality tests prevent error propagation.

In the following, we first present the new decoding algorithm and some decoding results for several long codes. Then we discuss several possible variations of this algorithm.

2 MSD of Multilevel Codes

An m-level concatenated code [3] is formed from a set of m inner codes and a set of m outer codes as shown in Figure 1. The m inner codes are coset codes formed from a binary linear block code A of length n and a sequence of m linear subcodes of A_1, denoted by $A_2, A_3, \ldots, A_{m+1} = \{\mathbf{0}\}$, with $A_{i+1} \subset A_i$ for $1 \leq i \leq m$. The i-th inner code $[A_i/A_{i+1}]$ is simply the set of **representatives** of cosets of A_i modulo A_{i+1}, denoted by A_i/A_{i+1}. Each codeword in A_1 is a unique sum of m coset representatives from $[A_1/A_2], [A_2/A_3], \ldots, [A_m/\{\mathbf{0}\}]$, respectively.

At the i-th level encoding, $\mathbf{b}^{(i)}$ is encoded into the following sequence,

$$\mathbf{c}^{(i)} = (\mathbf{c}_1^{(i)}, \mathbf{c}_2^{(i)}, \ldots, \mathbf{c}_N^{(i)}) = (f_i(b_1^{(i)}), f_i(b_2^{(i)}), \ldots, f_i(b_N^{(i)})), \tag{1}$$

where $\mathbf{c}_j^{(i)} = f_i(b_j^{(i)})$ is a coset representative in $[A_i/A_{i+1}]$. Therefore, $\mathbf{c}^{(i)}$ is a sequence of coset representatives from $[A_i/A_{i+1}]$ and is a codeword in the i-th level concatenated code, denoted by $C_i \triangleq B_i \circ [A_i/A_{i+1}]$. The direct sum $C \triangleq B_1 \circ [A_1/A_2] \oplus B_2 \circ [A_2/A_3] \oplus \ldots \oplus B_m \circ [A_m/A_{m+1}]$ forms an m-level concatenated code. For simplicity, we denote this m-level code by $C \triangleq \{B_1, B_2, \ldots, B_m\} \circ \{A_1, A_2, \ldots, A_m\}$. Every codeword $\mathbf{c} \in C$ is a sum of m codewords $\mathbf{c}^{(1)}, \mathbf{c}^{(2)}, \ldots, \mathbf{c}^{(m)}$ from the m component concatenated codes $C_1, C_2, \ldots, C_m$, respectively, i.e., $\mathbf{c} = \mathbf{c}^{(1)} + \mathbf{c}^{(2)} + \ldots + \mathbf{c}^{(m)}$.

Suppose a codeword $\mathbf{c}$ corresponding to m outer codewords $\mathbf{b}^{(1)}, \mathbf{b}^{(2)}, \ldots, \mathbf{b}^{(m)}$ is transmitted. Let $\mathbf{r} = (\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_N)$ be the received sequence at the output of the matched filter in the receiver, where j-th section $\mathbf{r}_j = (r_{j1}, r_{j2}, \ldots, r_{jn})$ consists of n real numbers. At the first decoding stage, $\mathbf{r}^{(1)} \triangleq \mathbf{r}$ is decoded into a codeword $\mathbf{b}^{(1)}$ in the first outer code B_1.

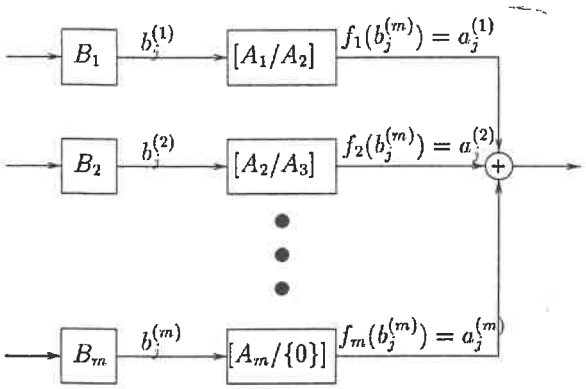


Figure 1: An encoder for an m-level concatenated code

For $1 \leq i \leq m$, let $q_i \triangleq |[A_i/A_{i+1}]|$ and $[A_i/A_{i+1}] = \{\mathbf{a}_j^{(i)} : 1 \leq j \leq q_i\}$. Then a coset in A_i/A_{i+1} is given by

$$\{\mathbf{a}_j^{(i)} + \mathbf{a} : \mathbf{a} \in A_{i+1}\}.$$

For $1 \leq i \leq m$, the i-th outer code, denoted by B_i, is an (N, K_i) linear block code over $GF(q_i)$. Let $f_i(\cdot)$ be a one-to-one mapping from $GF(q_i)$ onto $[A_i/A_{i+1}]$. This mapping is simply the inner code encoding at the i-th level. Let $\mathbf{b}^{(i)} = (b_1^{(i)}, b_2^{(i)}, \ldots, b_N^{(i)})$ be a codeword in the i-th outer code B_i.

Then the effect of the decoded estimate $\mathbf{b}^{(1)}$ (or $\mathbf{c}^{(1)}$) is removed from $\mathbf{r}^{(1)}$. This results in a modified received sequence $\mathbf{r}^{(2)} = (\mathbf{r}_1^{(2)}, \mathbf{r}_2^{(2)}, \ldots, \mathbf{r}_N^{(2)})$ with $\mathbf{r}_j^{(2)} = (r_{j1}^{(2)}, r_{j2}^{(2)}, \ldots, r_{jn}^{(2)})$. Then $\mathbf{r}^{(2)}$ is used for decoding at the second stage and is decoded into a codeword $\mathbf{b}^{(2)} \in B_2$. Again, the effect of $\mathbf{b}^{(2)}$ is removed from $\mathbf{r}^{(2)}$ to obtain a modified received sequence $\mathbf{r}^{(3)}$ for the third stage decoding. This continues until the last estimate $\mathbf{b}^{(m)}$ is obtained.

Suppose the i-th stage of decoding has been completed and $\mathbf{b}^{(i)}$ is the decoded estimate. From $\mathbf{b}^{(i)}$ and (1), we obtain the estimate $\mathbf{c}^{(i)} = (\mathbf{c}_1^{(i)}, \mathbf{c}_2^{(i)}, \ldots, \mathbf{c}_N^{(i)})$. Then the received sequence $\mathbf{r}^{(i+1)} = (\mathbf{r}_1^{(i+1)}, \mathbf{r}_2^{(i+1)}, \ldots, \mathbf{r}_N^{(i+1)})$ for the $(i+1)$-th stage of decoding is obtained from $\mathbf{c}^{(i)}$ and $\mathbf{r}^{(i)} = (\mathbf{r}_1^{(i)}, \mathbf{r}_2^{(i)}, \ldots, \mathbf{r}_N^{(i)})$ as follows: For $1 \leq l \leq n$ and $1 \leq j \leq N$, the l-th symbol of the j-th section $\mathbf{r}_j^{(i+1)}$ of $\mathbf{r}^{(i+1)}$ is given by

$$r_{jl}^{(i+1)} = r_{jl}^{(i)} \cdot (1 - 2c_{jl}^{(i)}) \tag{2}$$

(assuming BPSK signaling and $c_{jl} \rightarrow (-1)^{c_{jl}}$ mapping).

Each stage of decoding consists of two steps, the inner and outer decoding. At the i-th decoding stage, the inner decoder processes the N sections of the received sequence $\mathbf{r}^{(i)} = (\mathbf{r}_1^{(i)}, \mathbf{r}_2^{(i)}, \ldots, \mathbf{r}_N^{(i)})$ independently and forms N metric tables. Let $\{\mathbf{a}^{(i)}\} \triangleq \{\mathbf{a}^{(i)} + \mathbf{a} : \mathbf{a} \in A_{i+1}\}$ be a coset in A_i/A_{i+1} with $\mathbf{a}^{(i)}$ as the coset representative. The metric of $\{\mathbf{a}^{(i)}\}$ with respect to $\mathbf{r}_j^{(i)}$ is defined as the best metric between $\mathbf{r}_j^{(i)}$ and the vectors in $\{\mathbf{a}^{(i)}\}$, denoted by $M(\{\mathbf{a}^{(i)}\})$. Let $\mathbf{a}^{(i)} + \mathbf{a}^*$ be the vector in $\{\mathbf{a}^{(i)}\}$ with the best metric. This vector is called the **label** of $\{\mathbf{a}^{(i)}\}$ with respect to $\mathbf{r}_j^{(i)}$. Form a metric table, denoted by $MT_j^{(i)}$, which for each coset $\{\mathbf{a}^{(i)}\} \in A_i/A_{i+1}$, stores its metric $M(\{\mathbf{a}^{(i)}\})$ and its label $\mathbf{a}^{(i)} + \mathbf{a}^*$. This table is called the **metric table** for $\mathbf{r}_j^{(i)}$. The inner decoding at the i-th stage is to form N metric tables, one for each section of the received sequence $\mathbf{r}_j^{(i)}$. These N metric tables are then passed to the outer decoder. Let $\mathbf{b}^{(i)} = (b_1^{(i)}, b_2^{(i)}, \ldots, b_N^{(i)})$ be a codeword in B_i. The metric of $\mathbf{b}^{(i)}$ is defined as the following sum:

$$M(\mathbf{b}^{(i)}) = \sum_{j=1}^{N} M(\{f_i(b_j^{(i)})\}),$$

where $M(\{f_i(b_j^{(i)})\})$ is the metric of the coset $\{f_i(b_j^{(i)})\}$ with $f_i(b_j^{(i)})$ as the coset leader. The outer decoding is to find the codeword $\tilde{\mathbf{b}}^{(i)} \in B_i$ which has the best metric among all the codewords in B_i. This can be achieved by using a trellis-based decoding algorithm, such as the Viterbi algorithm.

3 The Iterative Multistage MLD Algorithm

For simplicity, we first describe the proposed IMS-MLD algorithm using a two-level concatenated code, $\{B_1, B_2\} \circ \{A_1, A_2\}$. Once this is understood, we generalize it to m-level codes.

Let $\mathbf{b}^{(1)} = (b_1^{(1)}, b_2^{(2)}, \ldots, b_N^{(1)})$ be the decoded codeword at the first stage. Then $\mathbf{b}^{(1)}$ has the best metric with respect to the received sequence $\mathbf{r}^{(1)} = \mathbf{r}$. The metric of $\mathbf{b}^{(1)}$ is given by

$$M(\mathbf{b}^{(1)}) = \sum_{j=1}^{N} M(\{f_1(b_j^{(1)})\}),$$

where $M(\{f_1(b_j^{(1)})\})$ is the metric of the coset $\{f_1(b_j^{(1)})\} \in A_1/A_2$. Let $f_1(b_j^{(1)}) + \mathbf{a}_j^*$ be the label of coset $\{f_1(b_j^{(1)})\}$ with respect to the j-th section $\mathbf{r}_j^{(1)}$ of $\mathbf{r}^{(1)}$. Then

$$M(\{f_1(b_j^{(1)})\}) = M(f_1(b_j^{(1)}) + \mathbf{a}_j^*), \text{ and } M(\mathbf{b}^{(1)}) = \sum_{j=1}^{N} M(f_1(b_j^{(1)}) + \mathbf{a}_j^*). \tag{3}$$

If we replace each component $b_j^{(1)}$ of the decoded codeword $\mathbf{b}^{(1)}$ by its corresponding coset label $f_1(b_j^{(1)}) + \mathbf{a}^*$, we obtain the following **coset label sequence,**

$$L(\mathbf{b}^{(1)}) \triangleq (f_1(b_1^{(1)}) + \mathbf{a}_1^*, f_1(b_2^{(1)}) + \mathbf{a}_2^*, \ldots, f_1(b_N^{(1)}) + \mathbf{a}_N^*). \tag{4}$$

From (3) and (4), we see that the metric of $\mathbf{b}^{(1)}$ is the metric of its corresponding coset label sequence.

If the coset label sequence $L(\mathbf{b}^{(1)})$ is a codeword in the overall concatenated code $C = \{B_1, B_2\} \circ \{A_1, A_2\}$, then it is the codeword in C that has the best metric and hence the most likely codeword. In this case, the decoding is done and the sequence $\mathbf{a}^* = (\mathbf{a}_1^*, \mathbf{a}_2^*, \ldots, \mathbf{a}_N^*)$ is a codeword in the second-level concatenated code $C_2 = B_2 \circ A_2$. The second outer codeword $\mathbf{b}^{(2)} = (b_1^{(2)}, b_2^{(2)}, \ldots, b_N^{(2)})$ can be recovered from $\mathbf{a}^*$ as follows: $\mathbf{b}^{(2)} = (f_2^{-1}(\mathbf{a}_1^*), f_2^{-1}(\mathbf{a}_2^*), \ldots, f_2^{-1}(\mathbf{a}_N^*))$, where $f_2^{-1}(\cdot)$ is the inverse mapping of $f_2(\cdot)$. Alternatively, for systematic encoding, the information bits are obtained directly from the coset label sequence. If the coset label sequence $L(\mathbf{b}^{(1)})$ is not a codeword in C, then it is a sum of a codeword $(f_1(b_1^{(1)}), f_1(b_2^{(1)}), \ldots, f_1(b_N^{(1)}))$ in $C_1 = B_1 \circ [A_1/A_2]$ and a sequence of codewords in A_2 which is **not** a codeword in $C_2 = B_2 \circ A_2$. In this case, decoding continues. Therefore, the condition that $L(\mathbf{b}^{(1)})$ is a codeword in C is an optimality condition, which will be used in the proposed IMS-MLD algorithm.

At the second stage of decoding, the received sequence $\mathbf{r}^{(2)} = (\mathbf{r}_1^{(2)}, \mathbf{r}_2^{(2)}, \ldots, \mathbf{r}_N^{(2)})$ for decoding is obtained by removing the effect of the decoded codeword $\mathbf{b}^{(1)}$ at the first stage

from $\mathbf{r}^{(1)} = \mathbf{r}$, as given by (2). The inner decoder forms N metric tables, one for each section of $\mathbf{r}^{(2)}$. These tables are then passed to the outer decoder. The outer decoder finds the codeword $\mathbf{b}^{(2)} = (b_1^{(2)}, b_2^{(2)}, \ldots, b_N^{(2)})$ in B_2 that has the best metric with respect to $\mathbf{r}^{(2)}$. The metric of $\mathbf{b}^{(2)}$ is given by

$$M(\mathbf{b}^{(2)}) = \sum_{j=1}^{N} M(f_2(b_j^{(2)})). \tag{5}$$

For $1 \leq j \leq N$, let $f_2(b_j^{(2)}) = \mathbf{a}_j^{(2)}$ which is a codeword in A_2. Then $\mathbf{a}^{(2)} \triangleq (\mathbf{a}_1^{(2)}, \ldots, \mathbf{a}_N^{(2)})$ is a codeword in $C_2 = B_2 \circ A_2$, and

$$M(\mathbf{b}^{(2)}) = M(\mathbf{a}^{(2)}) = \sum_{j=1}^{N} M(\mathbf{a}_j^{(2)}) \tag{6}$$

where $M(\mathbf{a}_j^{(2)})$ is the metric of $\mathbf{a}_j^{(2)}$ with respect to $\mathbf{r}_j^{(2)}$.

Now we compare metric $M(\mathbf{b}^{(1)})$ and metric $M(\mathbf{b}^{(2)})$. Recall that

$$M(f_1(\mathbf{b}_j^{(1)}) + \mathbf{a}_j^*) = \max_{\mathbf{a}_j \in A_2} M(f_1(\mathbf{b}_j^{(1)}) + \mathbf{a}_j)$$

Then for any $\mathbf{a}_j \in A_2$,

$$M(f_1(\mathbf{b}_j^{(1)}) + \mathbf{a}_j^*) \geq M(f_1(\mathbf{b}_j^{(1)}) + \mathbf{a}_j). \tag{7}$$

Since $\mathbf{a}_j^{(2)} \in A_2$, it follows from (7) that

$$M(f_1(\mathbf{b}_j^{(1)}) + \mathbf{a}_j^*) \geq M(f_1(\mathbf{b}_j^{(1)}) + \mathbf{a}_j^{(2)}). \tag{8}$$

Let $\mathrm{corr}(\cdot, \cdot)$ denote the correlation function (any other metric can be used as well). It is easy to show that

$$\mathrm{corr}(f_1(\mathbf{b}_j^{(1)}) + \mathbf{a}_j, \mathbf{r}_j^{(1)}) \;=\; \mathrm{corr}(\mathbf{a}_j, \mathbf{r}_j^{(2)}) \tag{9}$$

for any $\mathbf{a}_j \in A_2$. Then it follows from (3), (6), (8) and (9) that

$$M(\mathbf{b}^{(1)}) \geq M(\mathbf{b}^{(2)}), \tag{10}$$

where equality holds if and only if the sequence $\mathbf{a}^* = (\mathbf{a}_1^*, \mathbf{a}_2^*, \ldots, \mathbf{a}_N^*)$ is a codeword in C_2.

For the iterative two-stage MLD algorithm, decoding iterations are based on the generation of a sequence of estimates at the first stage. The estimates are generated in decreasing order of metrics, one at a time. This can be achieved by using the list Viterbi algorithm [6]. At the i-th iteration, the first-stage decoder generates the i-th best estimate, denoted by $\mathbf{b}^{(1),i} = (b_1^{(1),i}, b_2^{(1),i}, \ldots, b_N^{(1),i})$. Let $\mathbf{b}^{(2),i}$ denote the decoded codeword at the second stage based on $\mathbf{b}^{(1),i}$ and $\mathbf{r}^{(2)}$. The proposed algorithm is based on the following two theorems. Theorem 1 is a direct consequence of (10).

Theorem 1: For $i > 0$, $M(\mathbf{b}^{(1),i}) \geq M(\mathbf{b}^{(2),i})$, where equality holds if and only if the coset label sequence for $\mathbf{b}^{(1),i}$, $L(\mathbf{b}^{(1),i}) \triangleq (f_1(b_1^{(1),i}) + \mathbf{a}_1^*, f_1(b_2^{(1),i}) + \mathbf{a}_2^*, \ldots, f_1(b_N^{(1),i}) + \mathbf{a}_N^*)$, is a codeword in C, i.e., $(\mathbf{a}_1^*, \mathbf{a}_2^*, \ldots, \mathbf{a}_N^*)$ is a codeword in $C_2 = B_2 \circ A_2$.

Let i_0 be the integer such that $1 \leq i_0 < i$ and

$$M(\mathbf{b}^{(2),i_0}) = \max_{1 \leq j < i} M(\mathbf{b}^{(2),j}). \tag{11}$$

Then $\mathbf{b}^{(2),i_0}$ is the best decoded codeword at the second decoding stage during the first $i-1$ iterations. Theorem 2 follows from Theorem 1 and the fact that for $i < j$, $M(\mathbf{b}^{(1),i}) \geq M(\mathbf{b}^{(1),j})$.

Theorem 2: For $i > i_0$, if $M(\mathbf{b}^{(2),i_0}) \geq M(\mathbf{b}^{(1),i})$, then the codeword in C that corresponds to $\mathbf{b}^{(1),i_0}$ and $\mathbf{b}^{(2),i_0}$ is the most likely codeword with respect to the received sequence $\mathbf{r}$. If $M(\mathbf{b}^{(2),i_0}) < M(\mathbf{b}^{(1),i})$ and the coset label sequence $L(\mathbf{b}^{(1),i})$ is a codeword in C, then $L(\mathbf{b}^{(1),i})$ is the most likely codeword in C with respect to $\mathbf{r}$.

In fact, the optimality conditions of Theorem 2 are also necessary conditions for the proposed iterative decoding algorithm.

3.1 Decoding Algorithm

- **Step 1**: Compute the first (best) estimate $\mathbf{b}^{(1),1}$ of the first decoding stage and its metric $M(\mathbf{b}^{(1),1})$. Check whether the coset label sequence $L(\mathbf{b}^{(1),1})$ is a codeword in C. If it is, $L(\mathbf{b}^{(1),1})$ is the most likely codeword and the decoding stops. Otherwise, go to Step 2.

- **Step 2**: Perform second stage decoding and obtain the estimate $L(\mathbf{b}^{(2),1})$ with metric $M(\mathbf{b}^{(2),1})$. Set $i_0 = 1$, and store $\mathbf{b}^{(1),1}$ and $\mathbf{b}^{(2),1}$. Go to Step 3.

- **Step 3**: For $i > i_0$, $\mathbf{b}^{(1),i_0}$ and $\mathbf{b}^{(2),i_0}$ are currently stored in a buffer register together with the metric $M(\mathbf{b}^{(2),i_0})$. Determine the i-th best estimate $\mathbf{b}^{(1),i}$ of the outer code B_1, and its metric $M(\mathbf{b}^{(1),i})$. If $M(\mathbf{b}^{(2),i_0}) \geq M(\mathbf{b}^{(1),i})$, then $\mathbf{b}^{(1),i_0}$and $\mathbf{b}^{(2),i_0}$ together give the most likely code in C, and decoding is finished. Otherwise, go to Step 4.

- **Step 4**: Check if the coset label sequence $L(\mathbf{b}^{(1),i})$ is a codeword in C. If it is, $L(\mathbf{b}^{(1),i})$ is the most likely codeword in C and decoding is finished. Otherwise go to Step 5.

- **Step 5**: Generate $\mathbf{b}^{(2),i}$. Update i_0, $\mathbf{b}^{(1),i_0}$ and $\mathbf{b}^{(2),i_0}$ and $M(\mathbf{b}^{(2),i_0})$. Go to Step 3.

The decoding process iterates until the most likely codeword is found. Therefore, the maximum number of iterations is $q_1^{K_1}$. This is the extreme case. In general, the number of iterations required to obtain the most likely codeword is very small compared to $q_1^{K_1}$. We may limit the number of iterations to keep decoding computational complexity down and decoding delay small. In this case, the decoding algorithm achieves near optimal error performance.

Theorems 1 and 2, and the two-stage iterative MLD decoding can be generalized to m stages. In m-stage decoding, new decoding iteration can be initiated at any stage above the final stage. Decoding iteration begins with the generation of a new estimate at the starting stage, say stage-l. If all the $q_l^{K_l}$ estimates at the stage l (resulting from a particular sequence of codewords from stages above the stage l) have already been generated and tested, the decoder moves up to the $(l-1)$-th stage and starts a new iteration with a

new estimate. Decoding iterations continue until the ML codeword is found. Just like the two-stage decoding, the final decoding decision is made at the first stage.

Suppose the decoding process is at the i-th decoding stage of j-th iteration. Let $\mathbf{b}^{(i),j}$ denote the decoded codeword in the outer code B_i. Let $L(\mathbf{b}^{(i),j})$ denote the coset label sequence corresponding to $\mathbf{b}^{(i),j}$. The metric of $L(\mathbf{b}^{(i),j})$ or $\mathbf{b}^{(i),j}$ is denoted by $M(\mathbf{b}^{(i),j})$. Let $\mathbf{b}^{(i_0),j_0}$ denote the codeword whose metric $M(\mathbf{b}^{(i_0),j_0})$ is the **best (largest)** among the codewords that have been generated before the i-th decoding stage of j-th iteration, and whose coset label sequence $L(\mathbf{b}^{(i_0),j_0})$ is a codeword in the overall m-level code C. Then, the buffer contains $M(\mathbf{b}^{(i_0),j_0})$, $L(\mathbf{b}^{(i_0),j_0})$, the estimates of the stages $1, 2, \ldots, (i_0 - 1)$ from which the estimate $\mathbf{b}^{(i_0),j_0}$ resulted, and the estimates of the stages $1, 2, \ldots, (i-1)$ from which the estimate $\mathbf{b}^{(i),j}$ resulted.

At the completion of the i-th decoding stage of the j-th iteration, the decoder makes one of the following three moves:

1. If $M(\mathbf{b}^{(i),j}) \leq M(\mathbf{b}^{(i_0),j_0})$ the decoder moves up to stage-$(i-1)$ and starts a new iteration with a new estimate.

2. Otherwise, if $M(\mathbf{b}^{(i),j}) > M(\mathbf{b}^{(i_0),j_0})$, the coset label sequence $L(\mathbf{b}^{(i),j})$ is tested. If it is a codeword in C, then the buffer is updated ($M(\mathbf{b}^{(i_0),j_0}) = M(\mathbf{b}^{(i),j})$, etc.). The decoder moves up to stage-$(i-1)$ and starts a new iteration with a new estimate.

3. If $M(\mathbf{b}^{(i),j}) > M(\mathbf{b}^{(i_0),j_0})$, and the coset label sequence $L(\mathbf{b}^{(i),j})$ is not a codeword in C, then decoder moves down to the $(i+1)$-th stage of j-th iteration.

When the decoder reaches the last (m-th) stage, it must move up to $(m-1)$-th stage and start a new iteration (since the label sequence of the estimate $\mathbf{b}^{(m),j}$ is always a codeword in C).

Whenever the decoder reaches the first stage at the beginning of an iteration, a decision is made at the completion of the first-stage decoding whether the decoding is terminated or continues. Suppose the decoder has reached and completed the first stage decoding at the j-th iteration. The decoder makes one of the following moves:

1. If $M(\mathbf{b}^{(1),j}) \leq M(\mathbf{b}^{(i_0),j_0})$, then the decoding is finished. The ML codeword is formed from $\mathbf{b}^{(i_0),j_0}$ and the codewords above i_0-th stage which resulted in the generation of $\mathbf{b}^{(i_0),j_0}$.

2. Otherwise, if $M(\mathbf{b}^{(1),j}) > M(\mathbf{b}^{(i_0),j_0})$, and the coset label sequence $L(\mathbf{b}^{(i),j})$ is a codeword in C, then $L(\mathbf{b}^{(i),j})$ is the ML codeword. Decoding stops.

3. If $M(\mathbf{b}^{(1),j}) > M(\mathbf{b}^{(i_0),j_0})$ and $L(\mathbf{b}^{(i),j})$ is not a codeword in C, then the decoder moves down to the second stage and continues the decoding for the j-th iteration.

Thus, the tests performed at the first decoding stage are actually the optimality conditions.

In m-stage decoding, new decoding iteration can be initiated at any stage above the final stage. Decoding iteration begins with the generation of a new estimate at the starting stage, say stage-l. From there, the decoder can either move down to the next stage, or, if the coset label sequence is a codeword in C_l, it moves up and starts a new iteration at stage-$(l-1)$. If all the $q_l^{K_l}$ estimates at the stage l (resulting from a particular sequence of codewords from stages above the stage l) have already been generated and tested, the decoder moves up

to the $(l-1)$-th stage and starts a new iteration with a new estimate. Decoding iterations continue until the ML codeword is found. Just like the two-stage decoding, the final decoding decision is made at the first stage.

3.2 Simulation Results & Decoding Complexity

In this section, we give two examples in which IMS-MLD is applied to two RM codes of length 128. We assume BPSK transmission over additive white Gaussian noise (AWGN) channel. We also use encoding in reduced echelon form in all simulated cases to minimize the BER [7].

Example 1 Consider the third order RM code of length 128, which is a (128, 64, 16) code. This code can be decomposed as multilevel concatenated code [3], and a possible decomposition is given by:

$$(128,64,16) = \{(16,1)(16,5)^2,(16,5)(16,11),(16,11)^2(16,15)\} \circ \{(8,8),(8,5),(8,3)\}.$$

The outer codes are interleaved RM codes of length 16, while the inner codes are formed by a partition chain, $(8,8)/(8,5)/(8,3)$. The universal code is partitioned by the two subcodes of RM codes, namely $(8,5) \subset (8,7)$ and $(8,3) \subset (8,4)$.

The 16-section full code trellis has 2^{26} states [8]. However, after decomposition, the 8-section trellises of the component outer codes have 2^9, 2^8, and 2^9 states, respectively. Therefore, for MSD, the total trellis state complexity is equal to $2^9 + 2^8 + 2^9$, which is much less than 2^{26}.

The bit error performances for different algorithms are given in Figure 2. It can be seen that the performance curve of the IMS-MLD algorithm agrees with the union bound for the (128,64) RM code. The new algorithm outperforms the conventional MSD by 1.35dB at BER=2×10^{-6}.

The computational complexity is expressed in terms of number of real operations (addition and comparisons) required for decoding one block of data. The computational complexity of Viterbi algorithm based on the full code trellis is 9.3×10^9. For conventional MSD, the computational complexity is 4.26×10^4. Due to the iterative nature of the IMS-MLD algorithm, the average complexity varies with the SNR. The values of the average number of real operations per block, and average numbers of estimates generated at each decoding level (ave_i, for $i = 1,2,3$) are given in Table 1. All the values decrease as the SNR increases. At a certain point, the complexity of the IMS-MLD becomes even smaller than that of conventional MSD. Another way to compare is to include a test on the coset label sequence for the conventional MSD. If this sequence is a codeword at any level, the decoding of the following stages is not necessary. The complexity of this modified conventional MSD becomes variable with SNR, and ranges from 2.6×10^4 for SNR=2.0 dB, to 0.55×10^4 for SNR=5.0 dB. The complexities of various algorithms are given in Table 3 for comparison.

The problem of large average number of estimates generated at each stage, as well as the large worst case complexity, can be overcome by setting a limit on the number of estimates generated by the list Viterbi Algorithm of the outer codes. Setting these limits to 5 and 2, for the first and second outer decoder respectively, results in an iterative algorithm with the same performance as that of List MSD [4] with parameters 5 and 2. However, the

computational complexity of the suboptimum iterative algorithm is much lower, as it can be seen from Table 2 (complexity of the list decoding is equal to the upper bound on the complexity of suboptimum IMS algorithm). This suboptimum version of IMS not only has large reduction in the average number of estimates generated at each stage for small SNR, but it has bounded worst case complexity, as well. Complexities of different algorithms are given in Table 3 for comparison.

From Figure 2 and Table 3, it can be seen that, compared to conventional MSD, IMS-MLD achieves significantly better performance with relatively small increase in average computational complexity. In Figure 2 and Table 3, the list decoding of [4] is also included. It can be seen that IMS-MLD outperforms the List MSD by 0.4dB. When compared to Viterbi decoding algorithm based on the full code trellis, IMS-MLD achieves the same performance with enormous reduction in both computational and trellis complexity.

Example 2 Consider the 4-th order (128,99) RM code with the following decomposition

$$(128, 99, 8) = \{(16, 5)(16, 11), (16, 11)^2, (16, 15)^3(16, 16)\} \circ \{(8, 8), (8, 6), (8, 4)\}.$$

Again, the outer codes are obtained by interleaving the RM codes of length 16. The trellis state complexities of the three outher codes B_1, B_2, and B_3 are 2^8, 2^8, and 2^4, respectively. These complexities are very small compared to 2^{19} states of the full code trellis.

The IMS-MLD outperforms the conventional MSD by 0.75dB. If suboptimum IMS algorithm is used with the list sizes of the list Viterbi algorithm limited to 5 for the first stage outer code and 2 for the second stage outer code, it achieves almost optimum performance with a very small average complexity and with bounded worst case complexity. Table 4 contains complexities of different decoding algorithms.

4 Conclusion

In this paper, an efficient iterative multistage MLD algorithm has been presented. As shown with two examples, it reduces decoding complexity enormously, compared to Viterbi Algorithm. This algorithm can be tailored in many ways to trade off error performance for reduction in decoding complexity. One version with a significant reduction in worst case complexity is to limit the maximum number of iterations. In this way, the worst case complexity per block and maximum decoding delay are bounded.

Only the trellis based decoding algorithms (such as Viterbi Algorithm) have been considered because of their standardized implementation. The other reason was that Viterbi algorithm results in optimum decoding of component codes, which is necessary for the optimality of the overall iterative multi-stage decoding algorithm. For the suboptimum versions, computationally efficient probabilistic algorithms, like ordered statistics [9], can be applied for decoding component codes.

Although examples are given for RM codes, the same algorithm can be applied to any multilevel coded modulation code, and decomposable BCH and EG codes. Recently, many such codes have been found [10]. Some of these codes are the best known codes for given lengths and dimensions.

References

[1] H. Imai and S. Hirakawa, "A new multilevel coding method using error correcting codes." *IEEE Transactions in Information Theory*, vol. IT-23, no. 3, pp. 371-376, May 1977.

[2] E. L. Blokh and V. V. Zyablov, "Generalized Concatenated Codes," Moscow, 1976.

[3] J. Wu, S. Lin, T. Kasami, T. Takata and T. Fujiwara, "An Upper Bound on the Effective Error Coefficient of Two-Stage Decoding and Good Two-Level Decompositions of Some Reed-Muller Codes," *IEEE Transactions on Communications*, vol. COM-42, pp. 813-818, February/March/April 1994.

[4] D. Stojanović, "Multistage Decoding of Block and Convolutional Codes," MS Thesis, University of Hawaii at Manoa, Honolulu, HI, December 1996.

[5] U. Dettmar, J. Portugheis and H. Hentsch "New Multistage Decoding Algorithm" *Electronics Letters*, vol. 28 No. 7 pp. 635-636, 1992.

[6] N. Seshadri and C. W. Sundberg "List Viterbi Decoding Algorithms with Applications" *IEEE Transactions on Communications*, vol. COM-42, pp. 313-322, February/March/April 1994.

[7] M. Fossorier, S. Lin and D. Rhee "Bit Error Probability for Maximum Likelihood Decoding of Linear Block Codes and Related Soft Decision Decoding Methods," *IEEE Transactions on Information Theory*, vol. IT-44, November 1998.

[8] S. Lin, T. Kasami, T. Fujiwara and M. Fossorier, *Trellises and Trellis-based Decoding Algorithms for Linear Block Codes*, Kluwer Academic Publishers, 1998.

[9] M. Fossorier and S. Lin, "Soft-Decision Decoding of Linear Block Codes based on Ordered Statistics," *IEEE Trans. on Inform. Theory*, Vol. 41, pp. 1379-1396, September 1995.

[10] R. Morelos-Zaragoza, T. Fujiwara, T. Kasami and S.Lin, "Construction of Generalized Concatenated Codes and Their Trellis-Based Decoding Complexity" to appear in *IEEE Transactions on Inform. Theory*

Table 1: Decoding Complexity for IMS-MLD of RM(3,7)=(128,64,16)

SNR	2.0	2.5	3.0	3.5	4.0
No of opr [10^4]	45.0	12.6	10.0	4.0	2.6
ave_1	31	9.12	3.2	1.84	1.47
ave_2	40	11	3	1.1	0.69
ave_3	9.5	2.8	0.78	0.22	0.081

Table 2: Decoding Complexity for Suboptimum IMS of RM(3,7)=(128,64,16)

SNR	2.0	2.5	3.0	3.5	4.0
No of opr $[10^4]$	6.25	3.33	1.82	1.15	0.83
ave_1	3.24	2.55	2.04	1.71	1.46
ave_2	3.5	2.20	1.36	0.92	0.66
ave_3	1.41	0.67	0.30	0.14	0.07

Table 3: Computational complexity of different algorithms for (128,64,16) code

Decoding Algorithm		Average number of real operations at SNR:							Upper bound on complexity
		2.0	2.5	3.0	3.5	4.0	4.5	5.0	
conventional MSD	$[10^4]$	2.57	1.88	1.42	1.03	0.80	0.65	0.55	4.26
suboptimum IMS	$[10^4]$	6.25	3.33	1.82	1.15	0.83	-	-	36.6
optimum IMS-MLD	$[10^4]$	45.0	12.6	10.0	4.0	2.6	-	-	-
Viterbi	$[10^9]$	9.3				9.29	9.28	9.23	9.3

Table 4: Computational complexity of different algorithms for (128,99,8) code

Decoding Algorithm		Average number of real operations at SNR:							Upper bound on complexity
		2.5	3.0	3.5	4.0	4.5	5.0	5.5	
conventional MSD	$[10^4]$	1.2	1.1	0.9	0.7	0.56	0.4	0.3	1.5
suboptimum IMS	$[10^4]$	3.3	2.1	1.3	0.8	0.56	0.4	-	7.8
optimum IMS-MLD	$[10^4]$	22.0	7.9	6.3	1.0	0.56	0.4	-	-
Viterbi	$[10^9]$	4.62	4.61	4.56	4.44	4.20	3.82	3.29	4.63

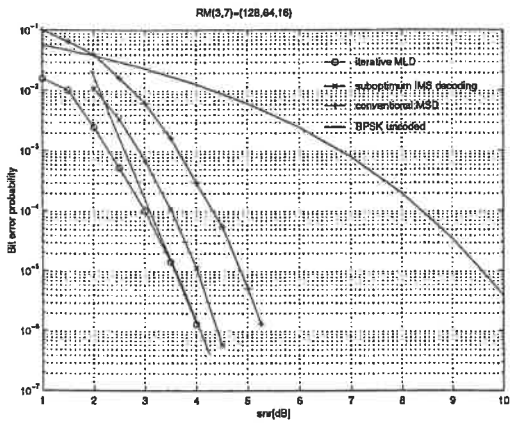


Figure 2: Performance of different algorithms for (128,64,16) code

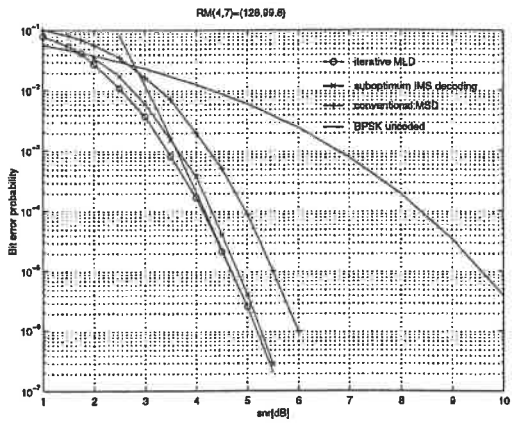


Figure 3: Performance of different algorithms for (128,99,8) code

Permutation Soft Decision Decoding of Some Expanded Reed-Solomon Codes

Emmanuelle Delpeyroux[+] and Jérôme Lacan[++]

[+] IRIT/AAECC, University Paul Sabatier, Toulouse, France,

[++] LIB, IUT Belfort-Montbeliard, France *.

Email: Emmanuelle.Delpeyroux@utbm.fr, Jerome.Lacan@utbm.fr

1 Introduction

The practical importance of Reed-solomon (RS) codes is well established (see [1]-[3]). For these codes, the use of soft decision decoding can really improve the performance (see [4]-[5]). In fact, in most cases, these codes are over $I\!F_{2^m}$ but they are used under their expanded binary form and the avalaible soft information is relative to the binary symbols. However, the soft decoding algorithms as the Generalized Minimum Distance (GMD) of Forney [6] or the Chase's algorithm [7] make use of soft decision information only on the byte level. In [1], it is explained that "the major drawback with RS codes (for satellite use) is that the present generation of decoders do not make full use of bit-based soft decision information". Some works were presented in order to improve this point [8]. This is also the purpose of this paper.

For this, we use for the decoding a permutation group acting on the q-ary image of some q^m-ary cyclic codes recently introduced in [9]. The idea of using some permutations for the hard decision decoding of some binary codes was already proposed (see for example [10]). But here, we use the permutations for the soft decision decoding of expanded RS codes.

The principle is the following. We first consider the received binary word and we determine its least reliable bits with the bit-level soft information. By using a permutation which lets invariant the code, we permute the received word in order to "group" the non reliable bits on the same bytes. These bytes are then considered as the least reliable for the soft decision decoding.

*The authors are now with the LaRIS, UT Belfort-Montbéliard, Rue du château - Sévenans F-90010 BELFORT Cedex, France

After the decoding, the binary form of the corrected word is permuted by the inverse permutation to obtain the initial emitted word.

This extended abstract is organized as follows. In section 2, we first recall the definition of the permutation group and its action over the codewords [9]. In [9], it is proved that the q-ary image of some q^m-ary RS codes are invariant under the action of some of these particular groups. We also recall these results. In the third section, we propose three decoding algorithms using the permutations. We present simulation results to evaluate their bit error rate (BER) performance for additive white gaussian noise (AWGN) channels.

2 Permutation groups and invariant codes under the action of some of these groups

In order to recall some results on codes invariant under the action of some particular permutation groups, let us fix some notations and explain some preliminary points.

In this paper we denote by $[N = q^m - 1, NZ]_{q^r}$ a cyclic code of length N over $\mathbb{F}_{q^r}$, where NZ is its set of nonzeros. Let $\underline{\alpha} = \{\alpha_0, \alpha_1, \ldots, \alpha_{m-1}\}$ be a basis of $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$ and let $a(z)$ be an element of $\mathbb{F}_{q^m}[z]/(z^N - 1)$. We define the q-ary image of $a(z)$ with respect to the basis $\underline{\alpha}$ by the bijective module homomorphism :

$$\mathcal{D}_{\underline{\alpha}} \; : \; \left\{ \begin{array}{ccc} \mathbb{F}_{q^m}[z]/(z^N-1) & \longrightarrow & (\mathbb{F}_q[x]/(x^N-1))^m \\ a(z) = \sum_{i=0}^{m-1}(\sum_{j=0}^{N-1} a_{i,j} z^j)\alpha_i & \longrightarrow & (a_0(x), a_1(x), \ldots, a_{m-1}(x)) \end{array} \right.$$

where $a_i(x) = \sum_{j=0}^{N-1} a_{i,j} x^j$.

Now, we consider the RS code C_1 equal to $[N, \{\beta\}]_{q^m}$, where β is a primitive element of $\mathbb{F}_{q^m}$. Let $g_1^{\underline{\alpha}}(z)$ be a generator of C_1. When we write the coefficients of $\mathbb{F}_{q^m}$ over the basis $\underline{\alpha}$ then $g_1^{\underline{\alpha}}(z)$ may be expressed as $\theta(z)(z^{u_0}\alpha_0 + \ldots + z^{u_{m-1}}\alpha_{m-1})$, where $\theta(z)$ is the primitive idempotent of $[N, \{\beta, \beta^q, \ldots, \beta^{q^m-1}\}]_q$ (see [9] section 4). This writing defines a particular m-tuple $(u_0, u_1, \ldots u_{m-1})$. The q-ary image of C_1 with respect to the basis $\underline{\alpha}$ is equal to $\{\mathcal{D}_{\underline{\alpha}}(c(z))/c(z) \in C_1\}$ and we denote it by $\mathcal{D}_{\underline{\alpha}}(C_1)$. Then a generator of $\mathcal{D}_{\underline{\alpha}}(C_1)$ is equal to $(\theta(x)x^{u_0}, \ldots, \theta(x)x^{u_{m-1}})$ (see [9] section 4). Let us denote it by $G_1^{\underline{\alpha}}(x)$.

The expression of $G_1^{\underline{\alpha}}(x)$ leads us to define a permutation set such that $\mathcal{D}_{\underline{\alpha}}(C_1)$ is invariant under the action of the permutations of this set. In [9] it is proved that this set is a group. We only recall here its definition.

Definition 2.1 *Let C_1 be equal to $[N, \{\beta\}]_{q^m}$, where β is a primitive element of $\mathbb{F}_{q^m}$. Suppose that $G_1^{\underline{\alpha}}(x) = (\theta(x)x^{u_0}, \theta(x)x^{u_1}, \ldots, \theta(x)x^{u_{m-1}})$.*

Let (l, σ, a) be the permutation of $\{0, 1, \ldots, m-1\} \times \mathbb{Z}/(N)$ which send (i, j) onto $(\sigma(i), (j + s_i^{(l,\sigma)})q^l + a)$, where

1- $\sigma \in S_m$ (the group of permutations of $\{0, 1, \ldots, m-1\}$)

2- $a \in Z/(N)$ and $l \in \{0, 1, \ldots, m-1\}$

3- $s_i^{(l,\sigma)} = q^{-l}u_{\sigma(i)} - u_i$, for $i = 0, 1, \ldots, m-1$.

We define by $\mathcal{P}_1^{\underline{\alpha}}$ the permutation group equal to $\{(l, \sigma, a), l = 0, 1, \ldots, m-1, \sigma \in S_m$ and $a \in \mathbb{Z}/(N)\}$.

Let C be equal to $[N, NZ]_{q^m}$. Each codeword of $\mathcal{D}_{\underline{\alpha}}(C)$ may be expressed as a $m \times N$-array, where a row corresponds to a polynomial of $\mathbb{F}_q[x]/(x^N - 1)$ expressed in the monomial basis $\{1, x, x^2, \ldots, x^{N-1}\}$ and a column corresponds to an element of $\mathbb{F}_{q^m}$ expressed over $\mathbb{F}_q$ in the basis $\underline{\alpha}$. By a position we means a pair (i, j) of $\{0, 1, \ldots, m-1\} \times \mathbb{Z}/(N)$. Let us now provide the precise definition of the action of $\mathcal{P}_1^{\underline{\alpha}}$ over an element of $\mathcal{D}_{\underline{\alpha}}(C)$.

Definition 2.2 *Let C be equal to $[N, NZ]_{q^m}$. Let $c(x)$ be an element of $\mathcal{D}_{\underline{\alpha}}(C)$. Let us suppose that $c(x) = (c_0(x), c_1(x), \ldots, c_{m-1}(x))$, where $c_i(x) = \sum_{j=0}^{N-1} c_{i,j}x^j$. Let (l, σ, a) be a permutation of $\mathcal{P}_1^{\underline{\alpha}}$.*

Then (l, σ, a) acts over $c(x)$ such that the coefficient $c_{i,j}$ (on position (i, j)) is sent onto the position $(l, \sigma, a)(i, j)$, for $i = 0, 1, \ldots, m-1$ and $j \in \mathbb{Z}/(N)$.

The new obtained word is called the permuted of $c(x)$.

In the following proposition, we prove that for some particular code C the permuted of a codeword of $\mathcal{D}_{\underline{\alpha}}(C)$ is also a codeword of $\mathcal{D}_{\underline{\alpha}}(C)$.

Proposition 2.3 *Let C_0 be equal to $[N, V]_{q^m}$, where V is some union of full sets of conjugates with respect to $\mathbb{F}_q$. Let C_1 be equal to $[N, \{\beta\}]_{q^m}$, where β is a primitive element of $\mathbb{F}_{q^m}$. And let us suppose that $G_1^{\underline{\alpha}}(x) = (\theta(x)x^{u_0}$, $\theta(x)x^{u_1}$, $\ldots$, $\theta(x)x^{u_{m-1}})$.*

Then $\mathcal{D}_{\underline{\alpha}}(C_0)$, $\mathcal{D}_{\underline{\alpha}}(C_1)$ and $\mathcal{D}_{\underline{\alpha}}([N, \{\beta\} \cup V]_{q^m})$ are invariant under the action of $\mathcal{P}_1^{\underline{\alpha}}$.

PROOF See [9] prop. 4.5. □

Let $\underline{\alpha}^\perp$ be the trace dual basis of $\underline{\alpha}$ and $C^\perp$ the dual code of C.

Proposition 2.4 *Let C be a code in $\mathbb{F}_{q^m}[z]/(z^N - 1)$ such that $\mathcal{D}_{\underline{\alpha}}(C)$ is invariant under the action of $\mathcal{P}_1^{\underline{\alpha}}$.*

Then $\mathcal{D}_{\underline{\alpha}^\perp}(C^\perp)$ is also invariant under the action of $\mathcal{P}_1^{\underline{\alpha}}$.

PROOF See [9] prop. 4.6. □

3 Decoding and simulations results

3.1 Decoding algorithms

We propose here three different algorithms based on permutations. The used permutations are the permutations $(l, \sigma, 0)$. We consider that q is equal to 2 but these algorithms may be applied in the non-binary case. The principle of the use of these permutations is the following :

We consider the received binary word and we permute it using one of these permutations. We transform the obtained word into a 2^m-ary word. The soft coefficient of each 2^m-ary symbol is determined as the worst (least reliable) soft coefficients of the corresponding bits. Using these new coefficients, we use the usual RS soft decision decoding algorithm to produce some proposed codewords. These codewords are expanded into their binary form and permuted with the inverse permutation. The distance between each obtained binary codewords and the initial binary received word is computed (see [6]). The nearest codeword (the more likely) is retained.

Clearly, if the used permutation is the identity, we obtain the classical decoding algorithm.

This operation is made for several permutations (one permutation corresponds to one step). The final corrected word is the best of the codewords proposed by the differents steps of the decoding. The difference between the three algorithms is in the choice of the permutations to use.

The first one (PERM1) simply uses all the possible permutations. Therefore, it processes $m \times m!$ soft decision decoding by word.

The second algorithm (PERM2) considers the differents rows of the received binary word (considered as a $m \times N$-array) and determines the $d-1$ worst positions of each row. The used permutations are all that are able to group onto one column only some of the non-reliable positions of the differents rows. The average number of the used permutations can be evaluated. The decoder tries $(d-1)^m$ times to group m positions onto one column.

It is possible to verify that the probability to group m random positions onto one column is less or equal to $(m! \times m)/N^{m-1}$. Thus to decode each word, the decoder makes on average less than $(d-1)^m \times (m! \times m)/N^{m-1}$ soft decision decoding by word.

The third algorithm (PERM3) determines the least reliable coefficient of each row. It then considers all the possibilities to choose $m-1$ rows in the m rows. For each possibility, it considers all the permutations which are able to group onto one column the worst position of each of the $m-1$ rows. If there are several permutations which verify this property, it chooses the one such that the last coefficient of the column is the worst possible. The decoder

therefore uses at most m soft decision decoding for each word.

3.2 Simulation results

In order to use the permutations in the decoding, let us consider the code $C = [7, \{\beta\} \cup \{\beta^5, \beta^3, \beta^6\}]_8$, where β is such that $\beta^3 + \beta^2 + 1 = 0$. The parameters of this code are $[7, 4, 4]$, so it is able to correct t errors and s erasures provided that $2t + s < 4$. Thus, it can correct one byte error and one byte erasure or 3 byte erasures. Let us consider $\mathcal{D}_{\underline{\alpha}}(C)$, the binary image of C with respect to the basis $\underline{\alpha}$ equal to $\{1, \alpha, \alpha^2\}$, where α verifies $\alpha^3 + \alpha + 1 = 0$. Then $G_1^{\underline{\alpha}}(x) = (\theta(x), \theta(x)x^5, \theta(x)x^2)$, where $\theta(x)$ is the primitive idempotent of $[7, \{\beta, \beta^2, \beta^4\}]_2$. Proposition 2.3 proves that $\mathcal{D}_{\underline{\alpha}}(C)$ is invariant under the action of $\mathcal{P}_1^{\underline{\alpha}}$.

The simulations first consist in adding a additive white gaussian noise (AWGN) to the binary form of the sent codewords. For each bit, a soft information is computed. Theoretically, this information is represented by real numbers, but in practical, it is quantified. In these simulations, it is quantified over 5 bits (32 possibles values).

In order to evaluate the bit error rate (BER) performances of these algorithms, we implemented other known algorithms. The first one is the maximum likelihood decoding (MLD). Its principle is simply to consider all the codewords and to choose the nearest one of the received binary word. It is the optimum decoding. The second one is the usual soft decision decoding algorithm of G.D. Forney: the generalized minimum distance (GMD). Note that the implemented version is the one presented by D.J. Taipale and M.J. Seo in [11]. It is equivalent to the GMD algorithm of G.D. Forney [6], but it needs less operations. The last one is the usual hard decision (HD) decoding.

The results of these simulations are presented on the figure 1.

The coding gain of the different algorithms at BER=10^{-5} is :

decoding	*MLD*	*PERM1*	*PERM2*	*PERM3*	*GMD*	*HD*
gain	5.4	5.15	5	4.75	4	2.6

The performances of the 3 decoders PERM1, PERM2 and PERM3 are very interesting. They are not very far of those of the MLD (resp. 0.25, 0.4 and 0.65) and the gain in comparison with the GMD is significant (near 1.15, 1 and 0.75 dB).

If we consider the argument of complexity, PERM2 and PERM3 are the most interesting. Actually, in all cases, PERM1 processes a soft decision decoding for all the permutations. Thus, the number of needed operations is multiplied by $m \times m! = 18$. For PERM2, it is explained in the paragraph

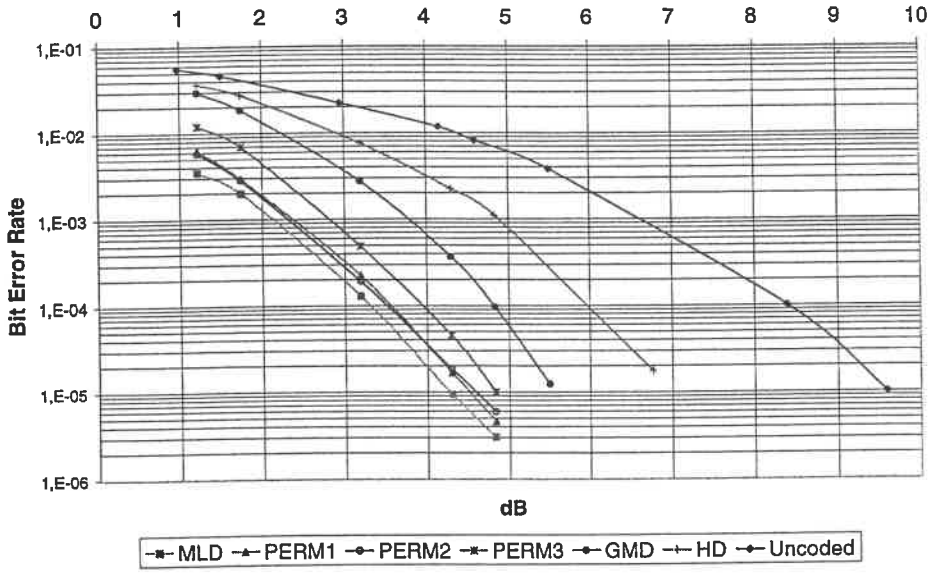


Figure 1: BER performances of the different decodings

3.1 that the average number of possible permutations is $(d-1)^m \times (m! \times m)/N^{m-1} = 9.91$. But a permutation can appear several times in the possible permutations. In practical, the soft decision decoding associated to this permutation is made only once. This fact is also true for PERM3. For example, the $m = 3$ permutations found by PERM3 are identical in near of third of cases. In this case, the decoder processes only one soft decision decoding.

Moreover, for each of these three algorithms, as the different decodings are independant, it is possible to process them in a parallel way. Furthermore, the electronic complexity of a RS decoder including the permutations seems to be not very important than one of an usual RS decoder.

Acknowledgments

The authors wish to acknowledge Professor Alain Poli for permit us to use the software SECC made by M.C. Gennero, A. Poli and C. Poli.

References

[1] E.R. Berlekamp, R.E. Peile, and S.P. Pope, "The application of error control to communications", *IEEE Commun.Mag.,* vol 25, 1987.

[2] W.W. Wu, D. Haccoun, R.E. Peile, and Y. Hirata, "Coding for satellite communications", *IEEE J.Select. Areas Commun.*, vol. SAC-5 , 1987

[3] Consultative Committee for Space Data System, "Recommendations for Space Data System Standarts: Telemetry Channel Coding", *Blue Book*, 1984.

[4] J.G. Proakis, *"Digital Communications"*, New York: McGraw-Hill, 1983.

[5] G.C. Clark and J.B. Cain *"Error Control Coding for Digital Communications"*, New York: Plenum, 1981

[6] G.D. Forney, Jr. "Generalized minimum distance decoding", *IEEE Trans. Inform. Theory*, vol. IT-12, 1966.

[7] D. Chase, "A Class of Algorithms for Decoding Block Codes with Channel Measurement Information", *IEEE Trans. Inform. Theory*, vol. IT-1, no. 6, Jan. 1972.

[8] A. Vardy and Y. Be'ery, "Bit-level Soft Decision Decoding of Reed-Solomon Codes", *IEEE Trans. Inform. Theory*, vol. 39, no. 3, March 1991.

[9] J. Lacan and E. Delpeyroux, "Permutation Group of the q-ary Image of Some q^m-ary Cyclic Codes", Finite Field: Theory, Applications and algorithms, *Contemporary Mathematics, 225,* American Mathematical Society, 1998.

[10] O. Papini, J. Wolfmann, "Algèbre Discrète et Codes Correcteurs", Editions Springer-Verlag, *Collections "Mathématiques et Applications"*, vol. 20, 1995.

[11] D.J. Taipale and M.J. Seo, "An Efficient Soft-Decision Reed-Solomon Decoding Algorithm", *IEEE Trans. Inform. Theory*, vol. 43, no. 2, July 1994.

Bit-level soft-decision sequential decoding for Reed Solomon codes

Min-seok Oh
CCSR
University of Surrey
GUILDFORD, GU2 5XH, U.K.
m.oh@ee.surrey.ac.uk

Peter Sweeney
CCSR
University of Surrey,
GUILDFORD, GU2 5XH, U.K.
p.sweeney@ee.surrey.ac.uk

Abstract

We present a novel sequential decoding based on bit-level soft-decision for RS codes. In the scheme, each information bit stream is encoded by a binary generator matrix in systematic form and a binary-branch trellis is constructed by using a binary paritycheck matrix. For sequential decoding, the standard Fano decoding algorithm is modified to get near maximum likelihood performance. In order to minimise the required computations, we employ a technique of sequence shifting so that a more reliable consecutive bit sequence is chosen as the information sequence.

I. Introduction

Reed Solomon codes are known to be very effective for practical applications requiring burst error correction or concatenated coding. In spite of these strengths, soft-decision decoding is not generally available, being much more complex than hard-decision decoding. Although Forney and Chase have presented algebraic decoding methods using soft decisions [14][15], they used the soft-decision information at the symbol level and their methods had limitations in the aspect of complexity and performance. More recently, there was a bit-level soft-decision decoding approach [22] using a binary matrix for RS codes. It was a useful approach toward the implementation of maximum likelihood performance for RS codes using bit-level-soft-decision decoding.

Subsequently, the use of trellis decoding methods has also been reported [4] using a sequential decoding method on symbol level trellis structure. In that work, although the results obtained were good, the approach did not use the soft-decision information at the bit level and the non-binary-branch trellis structure needed a large number of computations due to alternative path searching. This potential problem is expected to be more serious in the application to long RS codes.

From the view-point of the sequential decoder, a trellis structure with fewer branches will be more efficient even if the search depth increases. The reason is that the complexity of sequential decoding does not increase exponentially with constraint length, in particular with the number of parity check symbols in block codes.

In this paper we present a decoding method that can completely use the bit level soft-decision information. It is carried out on a systematic trellis structure with binary-branches, which is constructed by a decomposition process in which the multilevel linear cyclic block codes are mapped onto an equivalent binary code. In the structure, the binary parity check matrix is used to construct a systematic trellis structure[1]. This trellis structure makes the sequential decoder reflect bit-level soft decision information.

For decoding, the Fano algorithm is adapted to achieve near maximum likelihood performance. In order to get more computational gain, a technique of sequence shifting is introduced, in which a more reliable consecutive bit sequence is chosen as the

information sequence by cyclic-shifting of the received sequence. Since a shifted sequence with the most reliable information sequence is also a code word, the decoder can simply use the same trellis structure without any additional process.

Simulation has been carried out assuming BPSK modulation and 16 level demodulator quantization on an AWGN channel. The procedure for the binary-branch trellis is described in Section II. In section III, the Fano decoding algorithm and its modifications are explained and there is a discussion of the computational problem and the different types decoding failure event. Finally, simulation results are shown and analysed in section IV in terms of performance and complexity.

II. Binary branch trellis for RS codes

Consider t-error correcting (n, k) RS codes over $GF(2^m)$ with minimum distance d. Since RS codes are linear cyclic codes in the symbol basis, the set of all possible codes, C, can be generated by the matrix equation such that $C = X \cdot G$ where X is the k information sequence $x = \{x_0, x_1, \cdots x_{k-1}\}$ and G is a $k \times n$ generator matrix with the cyclic form of the generator polynomial $g(x)$ [17][18][22].

In order to construct the binary generator matrix, let $\gamma_0, \gamma_1 \cdots \gamma_{m-1}$ be the basis of $GF(2^m)$ for binary representation so that any element β over $GF(2^m)$ can be expressed as

$$\beta = \sum_{i=0}^{m-1} \gamma^i \cdot b^i \qquad \text{for } b^i \in GF(2). \tag{1.1}$$

Then the code word sequence $c = (c_0, c_1, \cdots, c_{n-1})$ is represented by

$$c = (b_0^{m-1}, b_0^{m-2}, \cdots, b_0^0, b_1^{m-1}, b_1^{m-2}, \cdots, b_1^0, \cdots\cdots, b_{n-1}^{m-1}, b_{n-1}^{m-2}, \cdots, b_{n-1}^0), \tag{1.2}$$

where $b_i^j \in GF(2)$ and $c_i = \sum_{l=0}^{m-1} b_i^j \cdot \gamma^l$ for $i = 0,1 \cdots n-1$. Thus the i-th symbol element of the code word, c_i, can be written associated with the generator matrix G for an information symbol sequence $x = \{x_0, x_1, \cdots x_{k-1}\}$ as

$$\begin{aligned} c_i &= \sum_{j=0}^{k-1} x_j \cdot g_{ji} = \sum_{j=0}^{k-1} \left(\sum_{l=0}^{m-1} b_j^l \cdot \gamma^l \right) \cdot g_{ji} \\ &= \sum_{j=0}^{k-1} \left[\sum_{l=0}^{m-1} b_j^l \cdot \left[\gamma^l \cdot g_{ji} \right] \right] \end{aligned} \tag{1.3}$$

where g_{ji} is the symbol element in the j-th row and i-th column of the matrix G with an element over $GF(2^m)$. Since the term in the inner bracket becomes another linear combination, symbol elements of the matrix G can be viewed ate the bit level as

$$\begin{bmatrix} \alpha^{m-1} \\ \vdots \\ \alpha^0 \end{bmatrix} \cdot g_{ij} = \begin{bmatrix} \alpha^{m-1} \times g_{ij} \\ \vdots \\ \alpha^0 \times g_{ij} \end{bmatrix}, \tag{1.4}$$

where α is a primitive element of $GF(2^m)$. As a result we have the $k \cdot m \times n \cdot m$ binary generator matrix G_b. For the systematic codes, if the matrix G is changed before the decomposition, the obtained G_b is also in systematic form. With the binary generator matrix G_b, each bit of a code word is represented by

$$c_j^b = \sum_{i=0}^{km-1} X_i \cdot G_b(i, j) \qquad \text{for } j = 0,1, \cdots, nm-1, \tag{1.5}$$

where X_i is the i-th bit of information sequence, $G_b(i, j)$ is the i-th row and j-th column of generator matrix G_b. The code obtained is linear and quasi-cyclic.

Once the systematic binary generator matrix G_b is determined, the parity check matrix H_b is easily obtained by transposing the sub-matrix of the first $k \cdot m$ identical matrix. Thus we have the binary parity check matrix H_b from G_b as

$$H_b = [h_0, h_1, \cdots h_{nm-1}], \tag{1.6}$$

where h_j is the j-th column of H_b, which is a with $(n-k)\cdot m$-tuple.

From the binary parity check matrix H_b, we construct the trellis structure defined by Wolf [1]. The state for a node of depth $(j+1)$ is obtained as

$$S_{j+1} = S_j + c_j^b \cdot h_j \qquad \text{for } j = 0,1,\cdots,nm-1 \tag{1.7}$$

where S_0 is $(n-k)\cdot m$-tuple zero state and c_j^b is the j-th bit of a code word. The state for arbitrary depth k is expressed by

$$S_k = \sum_{j=1}^{k} c_{j-1}^b \cdot h_{j-1}. \tag{1.8}$$

This trellis structure has the following properties [1]:

- Number of possible states is $\min(2^k, 2^{n-k})$.
- Trellis sequence starts with $(n-k)\cdot m$-tuple zero state and ends with the zero state.
- Trellis sequence is periodic and the reverse trellis sequence also exists.

On the trellis structure, there are 2^{mk} possible paths which correspond to a unique code word.

III. Sequential decoder

In this section, the application of Fano sequential decoding is explained on the systematic binary-branch trellis structure which has been described in the previous section. Each searching operation is performed with the metric based on the bit-level soft-decision. For (n,k) RS codes over $GF(2^m)$ the length is $m \cdot n$ but the decoder needs the path extensions only up to depth $m \cdot k$ because the state at the end of an information sequence implicates the unique correct path.

For near maximum likelihood performance, the Fano algorithm is modified, and a sequence shifting method is presented to ameliorate the computational problem.

Decoding failure events of the sequential decoder

We consider three kinds of decoding failure sources in sequential decoding. The first happens in the case that the whole path metric Λ_n of a searched path is greater than the path metric Λ_t for the transmitted code. In this situation, the searched path is regarded as correct and thus these errors can not be corrected even in any maximum-likelihood decoder. We define this type of decoding failure as an *uncorrectable error event*, having probability p_v.

The second type of error event happens in the case that the sequential decoder has chosen a sub-optimum path as being correct. If maximum likelihood decoding, e.g. the Viterbi algorithm, were implemented, such errors would be able to be corrected because better paths still exist among those that have not been tried. This type of error occurrence is designated an *error event by a wrong path* and its probability is represented by p_w.

The third error source arises from the limitation of computations to avoid decoder buffer overflow. In a practical system, this problem can be handled in two ways. One is to deliver just the information symbols in a systematic code or to take linear combinations in a non-systematic code. The other method is to erase the affected sequence and call for

retransmission. Actually this type of errors is the most serious in practical applications[11]. We also define this decoding error probability as p_c.

From the consideration of above three types of the decoding failures, the overall decoding error probability P_e is upper bounded as

$$p_e \le p_v + p_w + p_c \tag{2.1}$$

Performance improvement

The decoding performance can be improved by minimising the p_w and p_c at the cost of additional computations. To limit the complexity, we introduce the *decision rule function* and the *path updating function*.

The *decision rule function* is used to reduce the p_w. This function is carried at the end of the depth to decide whether a searched path is accepted by a decision rule. If a searched path does not satisfy the *decision rule*, the path is ignored and the decoder initiates the back tracking operation to look for more likely path.

Fano proposed a threshold condition[5] for a correct path in sequential decoding such that

$$\sum_{i=1}^{N}\left[\log\frac{p(j|i)}{p(j)}\right] \ge N\cdot R, \quad \text{for } i=0,1 \tag{2.2}$$

$$\sum_{i=1}^{N}\left[\log\frac{p(j|i)}{p(j)} - R\right] \ge 0, \tag{2.3}$$

here $p(j)$ is the total probability of observing an output in the j-th quantization level, $p(j|i)$ is the probability of observing in the j-th quantization interval given that the symbol i is transmitted, N is the number of the output sequence, and R is the code rate. Let N be the code length, then the left term of equation (2.3) represents the whole path metric, and thus we can get a useful interpretation that the path metric of the correct path is lower bounded by zero. By this condition, we find decision rule I which is to compare the Λ_n for the searched path to zero.

Decision rule II has a further step that includes the additional comparison between the Λ_n and the Λ_k for the information sequence. Since the whole path metric Λ_n is greater than Λ_k in the correct path, only a path satisfying the condition $\Lambda_n > \Lambda_k$ is accepted as a correct path. Furthermore the second decision rule has the bias term B_1 to avoid decoding failure by a slight difference between Λ_k and Λ_n. The value of B_1 is empirically obtained from the inspection of the distribution of such error patterns. Figure 1.1 shows the flow chart of the decision rule.

Next we consider the decoding failure event with probability p_c. In general, a larger decoder buffer contributes to lower decoding error probability, but in the worst channel conditions it is likely that other error mechanisms will produce errors regardless of the buffering provided. Therefore the improvement of p_c within a given computational limit is desirable; this problem has been discussed by Anderson[6]. In this paper, we propose the use of a *path updating function* as follows.

The decoder calculates the whole path metric of a searched path at the end of the information sequence. Whenever a searched path has path metric greater than the best one recorded so far, the path is updated as the best path. In the meantime, when the decoder reaches the computational limit without any valid path search, it releases the best path as the decoded one.

In this scheme, since the updated path is always the most likely path among the searched paths, if the correct path has been tried at least once, the decoder can produce

the maximum likelihood decoding decision. Thus it is important that the searching operation is carried out in the region including the correct path. Furthermore if the computational limitation is set efficiently, the computational complexity will be improved considerably as well. Figure 1.2 shows the flow of the path updating function at the depth k which is the end of information sequence.

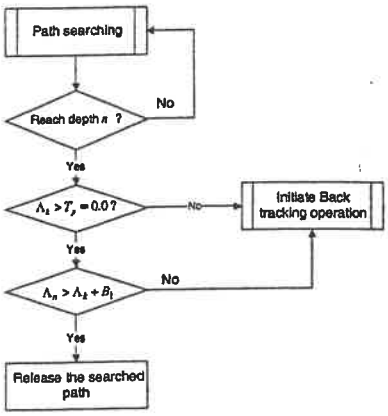


Figure 1.1 Flow chart of decision rule

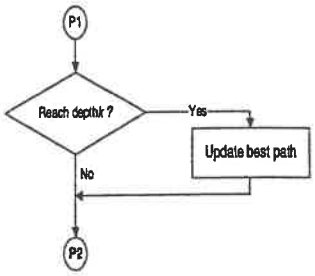


Figure 1.2 Path update process

Computational complexity of the sequential decoder

We consider two important aspects that affect the computational complexity in the sequential decoding; *trellis complexity* and *algorithm complexity*.

Trellis complexity is discussed in terms of state-complexity [21] or branch complexity [19]. In this paper, we present the method to reduce the branch-complexity.

The usual trellis structure for (n,k) RS codes has 2^m non-binary branches at each node. However, this trellis characteristic causes heavy alternative searching operations in the application of long RS codes. At this point, we realise the attractive property of sequential decoding that its complexity does not increase exponentially with the trellis depth. Thus the binary-branch trellis structure seems desirable to avoid the complex alternative searching.

The decomposition process explained in section II allows the simple binary-branch trellis. In the structure, the whole depth becomes longer from n to $n \cdot m$ but the branches at each level are reduced from 2^m to 2. On this trellis structure, sequential decoding is applied with the bit-level metric.

The other complexity problem exists in the sequential algorithm. There are many factors that affect the computations of a sequential algorithm such as the number of errors, the burst length, the error position, and the confidence level of the error bit. Among the factors, we are interested in the complexity dependent on the position of errors, because other factors are more difficult to manage in the decoder. Figure 2 is the computational distribution for the correction of a single bit error occurring in each bit position of a code word. It is seen that the computations required decrease as the error occurs in later position, and an error in the parity check part has an effect which is less dependent on the position than one in the information part.

By investigation of the computational distribution in Figure 2, we derive a scheme called *sequence shifting*, in which the decoder tries a cyclically shifted sequence with more reliable information bits. In this scheme, the decoder firstly deals with a sequence in which less reliable bits have been shifted to the rear of the sequence. The most reliable sequence is decided by summing the confidences for the k information symbol block from each symbol position, sorting the results into order.

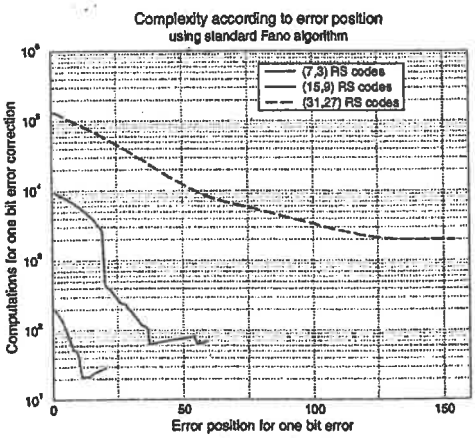


Figure 2 Computational distribution
for one-bit error correction in different positions

Thus the sum of confidences, I_i, for k consecutive symbol sequence starting from the i-th symbol position is represented by $I_i = \sum_{j=0}^{k-1} f_{(i+j)\%n}$ for $i = 0,1,\cdots,n-1$, where f_j is the confidence level of the j-th symbol of the received sequence. Figure 3.1 shows the process to select the best reliable information sequence at the received sequence.

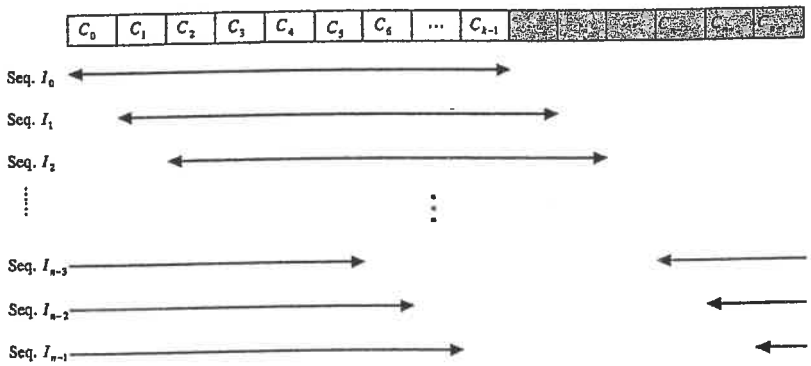


Figure 3.1 The most reliable sequence sorting

For the efficiency of the searching operation, the decoder can try more than one sequence, with a reduced computational limit, L_c, such that an overall computation limit L is observed. If the first sequence reaches the L_c, the next reliable sequence is tried. This procedure is continued within the trial number Y until the correct path has been found. Figure 3.2 shows this strategy for (15, 9) RS codes. Once the number of tried sequences Y is decided, the computational limit L_c for each sequence is

$$L_c = \frac{L}{Y} \tag{2.4}$$

At every sequence shift, the shifted number is saved and used to restore the original sequence order after a sequence has been released as the decoded one. By using the sequence shifting method with the sequence trial number Y, certain bad combinations of errors may be avoided. In particular, in channel conditions with low bit error rate, or where error tend to occur in bursts, the complexity can be improved significantly. Other approaches are of course possible, including completely re-ordering the sequence and the trellis according to reliability, but this might be impractical for hardware implementation.

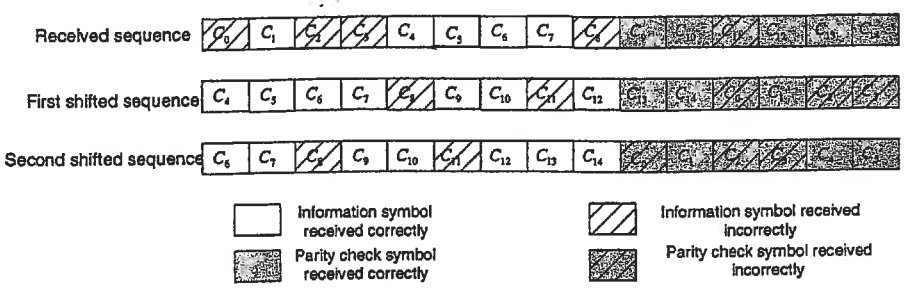
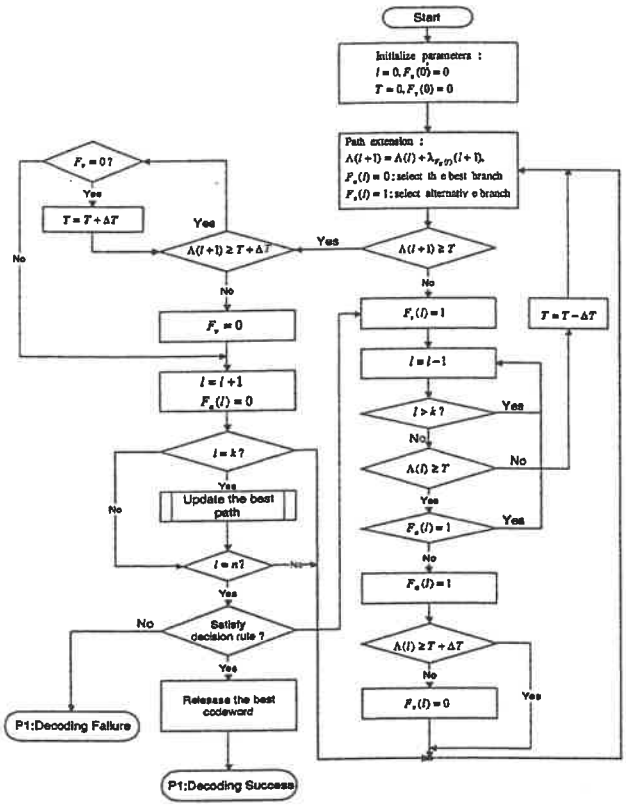


Figure 3.2 Sequence Shifting Process

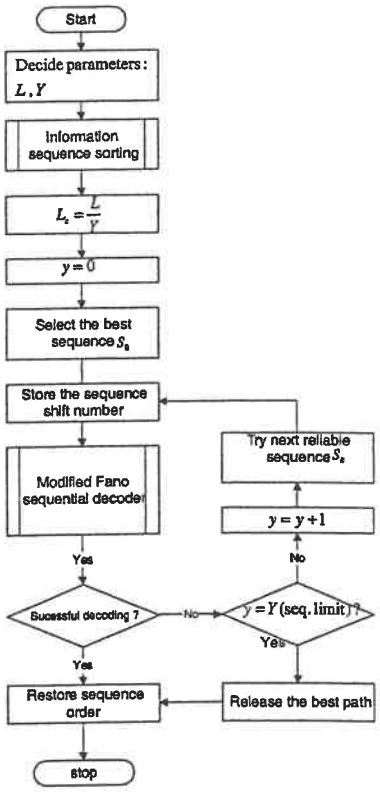
Cost-effective decoder

Finally we present a cost-effective decoder using the previous techniques for the improvement of the performance and complexity. Figure 4 shows the flow of the modified Fano algorithm and Figure 5 is the overall decoding procedure. In the decoding scheme, the computations increase due to the modification for the performance improvement than those using standard Fano algorithm, but the sequence shifting method makes the additional cost minimised with the performance improvement. Therefore if the optimised L and Y are applied, this decoding scheme will be very effective producing near maximum likelihoodperformance.



l = trellis depth, $F_v(l)$ = the first visit indication flag at depth l,
T = current threshold, $\Lambda(l)$ = path metric at depth l,
$F_a(l)$ = alternative searching indication flag at depth l,
ΔT = Threshold spacing

Figure 4 Modified Fano Algorithm



L = overall computational limit,
Y = maximum - trial - number

Figure 5 Overall decoding procedure

IV. Simulation Results

The simulation has been performed on digital communication model which consists of a BPSK system with 16 level quantized demodulator output on the AWGN channel. The performance is analysed in two aspects of the decoding error probability p_e and the average path extensions per bit, $\overline{C}$. Threshold spacing ΔT, the computational limit L and bias B_1 have been determined as the optimised values. Floating point numbers were used for the metric so that possible calculation error is minimised.

Effect of sequence shifting

Figure 6 and Figure 7 show the effect of the sequence shifting for (7,3) and (15,9) RS codes using the standard Fano algorithm. In the simulation, the values of the computational limit L were fixed at 2^{10} and 2^{18} for only one sequence, that is Y=1.

In the results, the decoding method using the sequence shifting produced slightly better performance for (7,3) RS codes and for (15,9) RS codes. On the other hand, in the complexity comparison the use of the shifting method showed much lower computations for both codes. In particular the reduction of complexity appeared more effectively in (15,9) RS codes. Moreover as Eb/No increases, the required computations have been reduced rapidly. From these results, it has been demonstrated that a shifted sequence with less complex error pattern produces lower complexity with at least equal performance. If such a scheme is used with an algorithm optimised for error correcting ability, it will be more effective in the decoding of long RS codes.

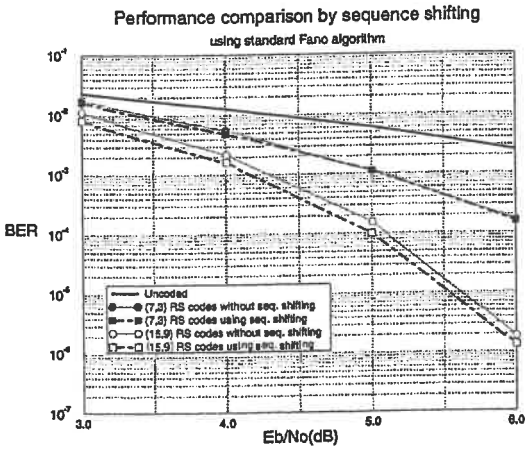


Figure 6 Comparison of performance for sequence shifting method

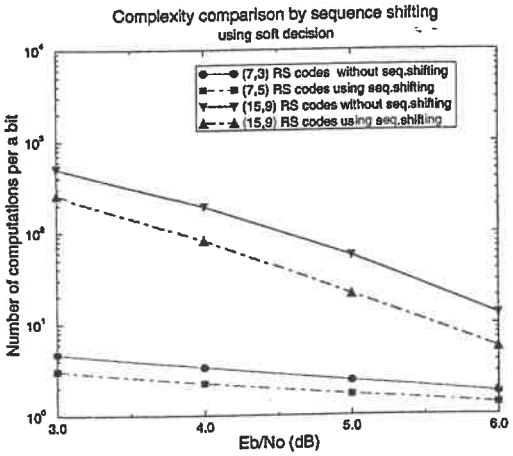


Figure 7 Complexity comparison for sequence shifting method

Cost effective performance

Figure 8 and Figure 9 are, from the simulations, considered to represent the most cost-effective configurations for RS codes with different code lengths and code rates. The simulations have been carried out under the optimised values of L, L_c and Y. In the results, because of the decision rule function, the decoder required more computations but the use of the sequence shifting with optimised Y, reduced the complexity and maximised the performance.

Figure 9 illustrates that the complexity decreases rapidly as SNR is high and the computational reductions were relatively effective in the decoding of low code rate with long codes.

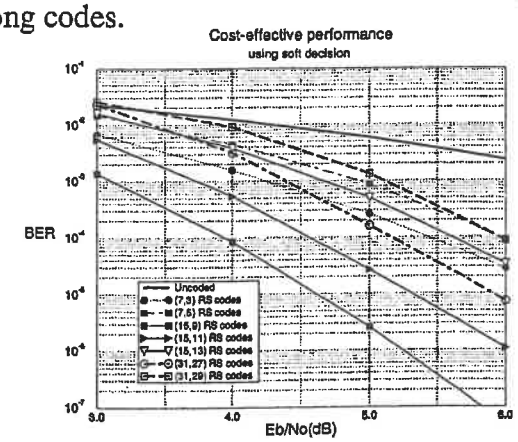


Figure 8 Cost-effective performance for RS codes

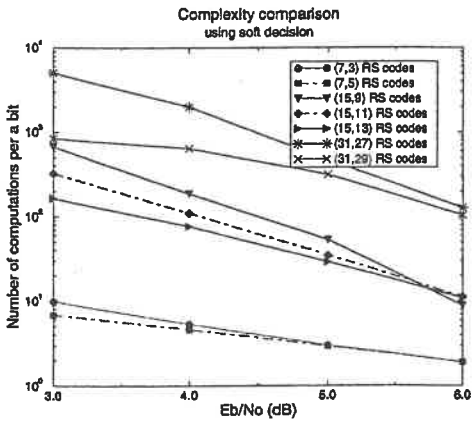


Figure 9 Complexity comparison for RS codes

V. Conclusion

We have presented a cost-effective decoder for RS codes. The binary codes produced by decomposition enabled the decoder to use the bit-level soft decision values on the binary-branch trellis structure. Although the overall searching depth increased, the binary-branch structure allowing only one alternative searching produced more efficient back-tracking-operation. Moreover the sequence shifting method demonstrated the great computational reduction and it maximised the roles of the decision rule and path updating function.

When the Viterbi algorithm is compared, this new decoding approach has very low complexity while the decoding error probability is close to the maximum likelihood performance. As the code length increases or SNR increases, the computational efficiency will be relatively improved. In the comparison with previous work [4], this approach showed slightly better coding gain but much lower complexity. In addition, since this approach has employed the soft decision values in bit level for the sequential decoding, it can reflect the channel information for each bit so that the decoder can search the correct path with more reliable bit-level channel information.

In order to achieve the most cost-effective decoder, the decoding parameters L, L_c, and Y should be adjusted according to a given channel condition, code rate, and code length. In general, the larger computational limit the better performance produced. However, the decision on these parameters must be chosen by considering the additional complexity required for any performance improvement.

References

[1] Wolf, J. K., "Efficient Maximum likelihood Decoding of Linear Block Codes Using a Trellis", *IEEE Trans. Inform. Theory,* vol. IT-20, No.1, Jan. 1978.

[2] Sweeney, P., "Cyclic Block Codes Definable Over Multiple Finite Fields", *Electronics Letters* 2nd March 1995,vol.31.

[3] Michelson, A. and Levesque, A., "Error-Control Techniques For Digital Communication", Wiley-Inter-science, 1985

On Bounding the Probability of Decoding Error with the Minimum Distance

(extended abstract)

Jean-Pierre Tillich[1] and Gilles Zémor[2]

[1]Université Paris-Sud

LRI, bât. 490, 91 405 Orsay

[2]ENST

Dépt. Informatique et réseaux

46 rue Barrault, 75 634 Paris

Jean-Pierre.Tillich@lri.fr, zemor@infres.enst.fr

1 Introduction

A classical problem of coding theory is to study the behaviour of the probability of a decoding error $f_C(p)$ when a block code C is used to transmit information over a binary symmetric channel with transition probability p. This quantity is defined as the average over all codewords x of C of the probability $f_C^x(p)$ that a maximum-likelihood decoding scheme does not recover the transmitted word x.

Shannon's fondamental theorem states that for any $R < 1 - H_2(p)$, where $H_2(p) = -p\log_2 p - (1-p)\log_2(1-p)$ is the binary entropy function, there exist families (C_n) of codes with increasing block length n and fixed rate R such that $f_{C_n}(p)$ vanishes when n tends to infinity. Ever since, a lot of effort was put into studying the asymptotic behaviour of $f_C(p)$ for the best codes. In particular it was proved that, for the best codes of length n, $f_C(p)$ is exponentially small in n as long as $R < 1 - H_2(p)$, and the best known

upper bounds of the form

$$f_C(p) \leq e^{-nE(R,p)+o(n)} \tag{1}$$

are due to Elias [3] and Gallager [4, 5]. Furthermore, results of type (1) are obtained by random coding arguments and are valid for *almost all* codes.

Let us rephrase the above facts by defining *the threshold decoding probability* of C, as the value θ such that

$$f_C(\theta) = 1/2.$$

For growing n and for codes of rate R with a distribution of weights approaching the binomial distribution, that is almost all codes, we have

$$\theta = H_2^{-1}(1-R) + o(1). \tag{2}$$

Furthermore, if we pick a random code C of large block length n, its probability $f_C(p)$ of decoding error will almost surely jump suddenly from almost zero to almost one around the threshold probability $\theta(C)$ and satisfy an upper bound of the type

$$f_C(p) \leq e^{-nK(\theta,p)+o(n)}$$

where $K(\theta,p)$ is some function such that $K(\theta,p) > 0$ for $p < \theta$. The best known lower bound on $K(\theta,p)$ gives :

$$\begin{array}{lll} K(\theta,p) & \geq & \theta \ln \frac{\theta}{p} + (1-\theta) \ln \frac{1-\theta}{1-p} \qquad \text{for } \frac{\theta^2}{\theta^2+(1-\theta)^2} \leq p \leq \theta \\ K(\theta,p) & \geq & H_2(\theta) \ln 2 - 2\ln(\sqrt{p} + \sqrt{1-p}) \qquad \text{for } 0 \leq p \leq \frac{\theta^2}{\theta^2+(1-\theta)^2} \end{array}$$

In this paper we prove a result of a similar nature for *all* codes. For reasons of clarity we shall state the results for linear codes but linearity is actually not essential. We obtain bounds on $f_C(p)$ which depends only on the code's minimum distance $d(C)$ and its threshold probability $\theta(C)$ and does not presuppose anything else about its weight distribution. The upperbound is of the form

$$f_C(p) \leq e^{-dL(\theta,p)}$$

where $L(\theta,p)$ is a function such that $L(\theta,p) > 0$ for $p < \theta$. In other words, as long as p stays smaller than a certain threshold value, $f_C(p)$ is an exponentially small function of the minimum distance.

More precisely, we have the following theorem.

Theorem 1 *Let C be a linear binary block code of any length, and minimum distance d. Over the binary symmetric channel with transition probability p, the probability of decoding error $f_C(p)$ of the code C satisfies :*

$$f_C(p) \leq \Phi\left[\sqrt{d}\left(\sqrt{-\ln(1-\theta)} - \sqrt{-\ln(1-p)}\right)\right]$$

where θ is the threshold decoding probability of C, i.e. is the transition probability for which $f_C(\theta) = 1/2$, and Φ stands for the Gaussian cumulative distribution, $\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-t^2/2} dt$.

Note that for growing d this upperbound is smaller than $e^{-dL(\theta,p)}$ with

$$L(\theta,p) = \frac{1}{2}\left(\sqrt{-\ln(1-\theta)} - \sqrt{-\ln(1-p)}\right)^2.$$

Moreover, if we denote by δ the quantity $d(C)/n$, then from (2) we know that for almost all codes θ is much bigger than $\delta/2$ and we obtain an upperbound on the probability of decoding error which vanishes exponentially in d for values of p between $\delta/2$ and θ. We don't know of any other bound which involves only d and θ which gives a result of this kind.

Theorem 1 is in fact a consequence of a more general result which we obtain by refining an isoperimetric method introduced by Margulis [6] and further developped by Talagrand [7] and Bobkov [1].

2 A sketch of the proof of Theorem 1

A general result about the threshold behaviour of monotone properties

Theorem 1 is actually a consequence of a much more general result which estimates the threshold behavior of monotone properties. By a monotone (increasing) property on $\mathbf{H}^n = \{0,1\}^n$ we mean a subset Ω of $\mathbf{H}^n$ which is increasing with respect to the partial order for which $x \preceq y$ iff for any $i = 1, 2, \ldots, n$ we have $x_i \leq y_i$. In other words whenever x belongs to Ω any $y \succeq x$ is also in Ω. A decreasing property is a subset $\Omega \subset \{0,1\}^n$ such that whenever $x \in \Omega$ any $y \preceq x$ is also in Ω.

To bring in this general result about the threshold behaviour of monotone properties we need some additional notation and a few definitions. Let μ_p denote the measure on $\mathbf{H}^n$ defined for any subset $\Omega \subset \mathbf{H}^n$ by

$$\mu_p(\Omega) = \sum_{x \in \Omega} p^{|x|}(1-p)^{n-|x|}.$$

Here $|x|$ denotes the weight $\sum_{i=1}^{n} x_i$ of a binary vector $x = (x_1, \ldots, x_n) \in \mathbf{H}^n$.

In order to study the behaviour of $f(p) = \mu_p(\Omega)$ as a function of p, Margulis introduced in 1974 [6], the function $h_\Omega(x)$ defined by

$$\begin{array}{rcll} h_\Omega(x) & = & 0 & \text{if } x \notin \Omega \\ h_\Omega(x) & = & card\{y \notin \Omega, d(x,y) = 1\} & \text{if } x \in \Omega \end{array}$$

where $d(x,y)$ denotes the Hamming distance between x and y. By estimating the quantity $\int h_\Omega d\mu_p$, Margulis obtains, for increasing sets Ω, a lower bound on $\frac{d}{dp}\mu_p(\Omega)$ that quantifies the "threshold behaviour" of $f(p) = \mu_p(\Omega)$. Talagrand obtained more precise information on $\mu_p(\Omega)$ by studying $\int \sqrt{h_\Omega} d\mu_p$ [7], and Bobkov and Goetze [1] improved his results by showing :

$$\int \sqrt{h_\Omega} d\mu_p \geq \frac{1}{12\sqrt{\ln \frac{1}{p(1-p)}}} J(\mu_p(\Omega)) \tag{3}$$

where $J(x) = x(1-x)\sqrt{\ln(\frac{1}{x(1-x)})}$. This may be thought of as an isoperimetric inequality, since $\int \sqrt{h_\Omega} d\mu_p$ is a measure of the boundary $\partial\Omega = \{x, h_\Omega(x) \neq 0\}$ of Ω.

We further improve on this by showing :

Theorem 2 *Let ϕ denote the normal density, i.e $\phi(t) = \frac{1}{\sqrt{2\pi}} e^{-t^2/2}$ and let Φ stand for the Gaussian cumulative distribution, that is $\Phi(x) = \int_{-\infty}^{x} \phi(t)dt$. Let Ψ be the function defined on $(0,1)$ by $\Psi(x) = \phi(\Phi^{-1}(x))$, and extended by continuity on $[0,1]$ with $\Psi(0) = \Psi(1) = 0$.*

For any increasing set Ω we have :

$$\int \sqrt{h_\Omega} d\mu_p \geq \frac{1}{\sqrt{2 \ln 1/p}} \Psi(\mu_p(\Omega)) \tag{4}$$

Remark. When $\mu_p(\Omega)$ tends to 0, it can be checked that the lower bound in the above theorem is equivalent to $\frac{1}{\sqrt{\ln 1/p}} J(\mu_p(\Omega))$.

A proof of theorem 2 will be given in the full paper. The isoperimetric inequality in theorem 2 can be integrated to yield the following :

Theorem 3 *Let $\Omega \subset \mathbf{H}^n$ be an increasing set, let $\partial\Omega = \{\omega, h_\Omega(\omega) \neq 0\}$ and let $\Delta = \inf_{\omega\in\partial\Omega} h_\Omega(\omega)$. Let $f(p) = \mu_p(\Omega)$ and let θ be defined by $f(\theta) = 1/2$. Then $f(p)$ satisfies :*

$$f(p) \leq \Phi\left(\sqrt{2\Delta}(\sqrt{-\ln\theta} - \sqrt{-\ln p})\right) \quad \text{for } 0 < p < \theta \tag{5}$$

$$f(p) \geq \Phi\left(\sqrt{2\Delta}(\sqrt{-\ln\theta} - \sqrt{-\ln p})\right) \quad \text{for } \theta < p < 1. \tag{6}$$

Before we give the proof of this theorem let us first gather a few properties on Φ and Ψ that we need. (For a proof of these statements see for instance [2] lemma 5.2 p.88).

lemma 1 *i. Ψ is a positive and concave function on $(0,1)$,and $\Psi(x) = \Psi(1-x)$ for every $x \in (0,1)$,*
ii. $\Psi' = -\Phi^{-1}$,
iii. $\Psi\Psi'' = -1$ on $(0,1)$.

Proof of theorem 3 : First note that the Cauchy-Schwartz inequality gives us

$$\int \sqrt{h_\Omega(x)}d\mu_p \leq \left(\mu_p(\partial\Omega)\int h_\Omega(x)d\mu_p\right)^{1/2}$$

and, since $\int h_\Omega(x)d\mu_p \geq \int_{\partial\Omega}\Delta d\mu_p = \Delta\mu_p(\partial\Omega)$ by definition of Δ, we get :

$$\int h_\Omega(x)d\mu_p \geq \sqrt{\Delta}\int \sqrt{h_\Omega(x)}d\mu_p \tag{7}$$

Next apply Margulis-Russo's formula [7]

$$\frac{d}{dp}\mu_p(\Omega) = \frac{1}{p}\int h_\Omega(x)d\mu_p,$$

which together with (7) and theorem 2 gives us :

$$f'(p) \geq \frac{\sqrt{\Delta}}{p\sqrt{2\ln 1/p}}\Psi(f(p)).$$

Apply property *iii.* of lemma 1, $-\frac{1}{\Psi(s)} = \Psi''(s)$, to obtain

$$-\Psi''(f(p))f'(p) \geq \frac{\sqrt{\Delta}}{p\sqrt{2\ln 1/p}}. \tag{8}$$

Next, multiply by -1 and integrate : we get, for $p < \theta$,

$$\int_p^\theta \Psi''(f(s))f'(s)ds \leq \int_p^\theta \frac{-\sqrt{\Delta}}{s\sqrt{-2\ln s}}ds,$$

i.e.

$$\Psi'(\theta) - \Psi'(p) \leq \left[\sqrt{-2\Delta \ln s}\right]_p^\theta.$$

Then we use the the fact that $\Psi'(f(\theta)) = \Psi'(1/2) = 0$ and $-\Psi'(f(p)) = \Phi^{-1}(f(p))$ by property *ii.* of lemma 1. The left-hand side of the last inequality is therefore simply $\Phi^{-1}(f(p))$. Since Φ is increasing, apply Φ to obtain (5). To obtain (6), integrate (8) between θ and p. ∎

Remark. Note that for large negative t we have $\log \Phi(t) \sim -t^2/2$, so that we have an upper bound for fixed $p \leq \theta$ and growing Δ which is exponential in Δ. It should be mentioned that applying the aforementioned isoperimetric inequalities of [7, 1] would also yield exponential bounds but with a much smaller exponent (by a factor of more than one hundred).

Maximum-likelihood decoding. It is rather straightforward to apply theorem 3 in order to estimate the probability of decoding error. In this extended abstract we will only consider the case of linear codes. Let $C \subset \mathbf{H}^n$ be such a code of dimension k and minimum distance d. Let $r = n - k$. Let H be a parity-check matrix for C and for any $x \in \mathbf{H}^n$ define its syndrome $\sigma(x) = H.{}^t x$. To every one of the 2^r possible syndromes s associate an $\omega \in \mathbf{H}^n$ of minimum weight such that $\sigma(\omega) = s$. Let Ω be the set of all those ω's, so that σ is a one-to-one correspondence between Ω and the set S of syndromes. The set Ω is a *decoding region* for the zero codeword, i.e. a set of *correctable error-patterns.* A maximum-likelihood decoding scheme consists of adding to the received vector v the vector $\omega \in \Omega$ such that $\sigma(\omega) = \sigma(v)$. A decoding error occurs if the codeword thus obtained is not the original codeword, i.e. if the error vector is not in Ω. This happens with probability

$$f_C(p) = 1 - \mu_p(\Omega).$$

Remark. The set Ω is *decreasing.*

We have :

proposition 1 *If Ω is a decoding region for the zero codeword of C, and if $\Delta = \inf_{\omega \in \partial\Omega} h_\Omega(\omega)$, then*

$$\Delta \geq d/2.$$

Proof : Let $\omega \in \partial\Omega$. This means that no codeword is nearer to ω than the zero codeword ($\omega \in \Omega$), and that there exists a codeword $c \neq 0$ such that changing one '0' coordinate of ω to '1' will change ω into a vector closer to c than to zero (ω is on the frontier). But then there must be at least $|c|/2$ '0' coordinates of ω that, when changed to '1' change ω into a vector closer to c than to zero. Otherwise $\omega + c$ would be a vector of weight strictly less than ω and with the same syndrome. This contradicts the definition of Ω. ∎

For any vector $x = (x_i)_{1 \leq i \leq n}$ of $\mathbf{H}^n$, let $\overline{x} = (1 - x_i)_{1 \leq i \leq n}$. Note that $\overline{\Omega}$ is an increasing set, that $h_\Omega(x) = h_{\overline{\Omega}}(\overline{x})$, and that

$$\mu_p(\Omega) = \mu_{1-p}(\overline{\Omega}).$$

Theorem 1 therefore follows from theorem 3 applied to $\overline{\Omega}$.

References

[1] S. Bobkov, F. Goetze, "Discrete isoperimetric and Poincaré-type Inequalities", Technical report SFB 343 University of Bielefeld 96-086, November 1996.

[2] I. Csiszàr, J. Körner, *Information Theory Coding theorems for discrete memoryless systems*, Academic press, 1981.

[3] P. Elias, "Coding for two noisy channels," in: *Information Theory*, Academic Press, 1956, pp. 61–74.

[4] R. G. Gallager, "A simple derivation of the coding theorem and some applications," *IEEE Trans. Inform. Theory*, vol. 11, 1965, pp. 3–18.

[5] R. G. Gallager, *Information theory and reliable communications*, Wiley, 1968.

[6] G. Margulis, "Probabilistic characteristics of graphs with large connectivity", Problemy Peredachi Informatsii, 10 (1974), pp. 101–108.

[7] M. Talagrand, "Isoperimetry, logarithmic Sobolev inequalities on the discrete cube, and Margulis' graph connectivity theorem", Geometric and Functional Analysis, 3 (1993), pp. 295–314.

On the relation of error correction and cryptography to an off line biometric based identification scheme

George I. Davida* Yair Frankel†
Brian J. Matt‡ René Peralta §

December 9, 1998

Abstract

An off-line biometric identification protocol based on error correcting codes was recently developed as an enabling technology for secure biometric based user authentication. The protocol was designed to bind a user's iris biometric template with authorization information via a magnetic strip in the off-line case while reducing the exposure of a user's biometric data. In this paper we give an in depth discussion of the role of error correcting codes in the cryptographically secure biometric authentication scheme.

An Iris scan is a biometric technology which uses the human iris to authenticate users [BAW96, HMW90, Dau92, Wil96]. This technology produces a 2048 bit user biometric template such that any future scan of the same user's iris will generate a "similar" template. By similar, we mean having an

*Center for Cryptography, Computer, and Network Security, University of Wisconsin-Milwaukee, USA. E-mail: davida@cs.uwm.edu.

†CertCo LLC, New York, NY, USA. E-mail: yfrankel@cs.columbia.edu.

‡Sandia National Laboratories. E-mail: bjmatt@sandia.gov. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

§Center for Cryptography, Computer, and Network Security, University of Wisconsin-Milwaukee, USA. E-mail: peralta@cs.uwm.edu

acceptable Hamming distance within a predefined range, usually around one to ten percent of the size of the code (e.g., Hamming distance between original reading and future reading is set to be in the range from 20 to 200). One can think of a biometric reading of a user as a faulty communication channel which may introduce a limited number of errors. Moreover, the Hamming distance for the biometric readings of two different users has been shown to be much higher, about 45 percent (or 920 bits).

Recently, a cryptographically secure mechanism for off-line biometric identification based on majority decoding and error correction codes (ECC) was developed [DFM98]. The process of an off-line biometric system is the following. A user, during an initialization step, is provided with a storage device / token (e.g., a magnetic strip, smartcard, etc.) by a trusted authorization authority. The token contains a signature and other data which can later be used to prove that the user's biometric is cryptographically bound to the signature of the trusted third party. During a future reading, the user first provides the token to a reader, the reader then obtains a new iris scan template from the user, and finally the reader determines if the new scan is cryptographically bound to the signature (of the trusted authority) on the card. It should be noted that the purpose of the signature is to enable the verification process to be done off-line, i.e. without connectivity to the trusted authority during future verifications. Moreover, it should be observed that the system must handle differences (within the allowed Hamming distance) from the original reading and future readings of the user's iris scan, because the digital signature verification will fail if there is any differences from its original input (message). Off-line biometric authentication protocols, of course, can be used in an online mode by replacing the token entry as a record on an online database.

A biometric identification system which provides the user's biometric template in the clear may not be acceptable to the user, because template could be used for unacceptable purposes if the template is obtained by an unauthorized individual. For instance, an iris scan may be used for medical purposes by an insurance company instead of the legitimate identification process the user was told to submit to.

In the work [DFM98], the feasibility of protecting the privacy of a user's biometric on an insecure storage device was studied. It was suggested that providing additional privacy for the user's biometric may provide for stronger user acceptance. An additional constraint to make the system scalable was

that neither the user nor the reader have private keys (or passwords) when the user must have authorization amongst multiple readers and when password protection is inappropriate. Providing for authorization bound to a biometric template appears to be inherently difficult in this model, because the user's biometric template cannot exist in the clear on the storage device. Since the original template is not stored on the token, a new verification algorithm, different from measuring Hamming distance from original template to new reading, had to be developed.

Here we study the relation of error correction and cryptography to an off-line biometric based identification scheme presented in [DFM98]. In particular, we will study the role of majority decoding, along with algebraic decoding, in the authentication scheme.

1 Background

1.1 Error correcting codes

Majority decoding: In the rest of the paper, we will consider only binary error correcting codes. We will denote by $a||b$ the concatenation of two strings a,b.

Let $\vec{v_i} = v_{i,1}||v_{i,2}||\ldots||v_{i,n}$ be n bit code vectors. Given odd M vectors $\vec{v_i}$, a majority decoder computes vector $\vec{V} = V_1||V_2\ldots||V_n$, where $V_j = \text{majority}(v_{1,j},\ldots,v_{M,j})$, i.e., V_j is the majority of 0's or 1's of bit j from each of the M vectors. We shall use majority decoding primarily to get the best biometric reading possible, thus reducing the Hamming distance between successive $final$ readings $\vec{V}$.

In the biometric authentication protocol, described in Section 1.2 the biometric being measured will be estimated by sampling since the actual unique iris is not measured with precision. The samples that are taken of the iris will converge to the actual unique individual biometric, with majority decoding, with high probability.

Algebraic decoding: An $[n,k,d]$ code [Ber68, MS78, PW88] is a code of n bit codewords (vectors) where k is the number of information digits and d is the minimum distance of code. Such a code can correct $t = (d-1)/2$ errors.

Note: Bounded distance decoding: In the rest of the paper, we assume that the decoding performed at the point of verification is to correct at most $(d-1)/2$ errors. This is necessary to ensure that no bogus biometric is decoded into a valid one. Bounded distance decoding can be readily implemented through a simple count of the Hamming weight of the error vector computed. In some decoding schemes, the error locations that are computed are the roots of some polynomial $\sigma(z)$ over $GF(2^m)$ of degree $t' = degree(\sigma(z))$. If $t' > t = (d-1)/2$ then the biometric is rejected.

1.2 An off-line biometric system

The basic idea of [DFM98] is that a user's biometric template can be used as the information bits of an error correcting code. Now instead of including the biometric template on the storage device, only the error correction bits are necessary. Since only the check bits are stored on the user's card, the available information about the biometric template is reduced. On the other hand, the reader can take a new reading of the user's biometric template, append the error correcting bits, remove the errors using bounded distance decoding, and finally reproduce the original template, which can be verified with the signature on the token.

One other hurdle has to be overcome to provide security. The signature may itself leak the user's template. Observe that $\langle M, \mathrm{SIG}(M)\rangle$ is a signature for message M which leaks all bits of M, yet is a valid signature of M. To resolve this problem, special hash functions were used.

Here is a brief summary of the basic off-line biometric protocol presented in [DFM98].

System Setup: The authorization authority generates its public and private keys and disseminates its public key to the biometric readers. The system also sets up an algebraic $[n, k, d]$ code. We remind the reader that we use bounded distance decoding.

User Initialization: To register, M biometric templates of length k are independently generated for the legitimate user. Majority decoding is then applied to the M biometrics to obtain the user's k bit template $\vec{T}$. Given the k information digits $\vec{T}$, an n digit codeword $\vec{T}||\vec{C}$ is constructed, where $\vec{C}$ are the check digits, in the $[n, k, d]$ code defined at system setup. A storage device is constructed with the following information:

1. Name of the individual, NAME.
2. Other public attributes ATTR, such as the issuing center and a user's access control list.
3. The check digits $\vec{C}$, of the biometric.
4. Sig(Hash(NAME, ATTR, $\vec{T}||\vec{C}$)) where Sig(x) denotes the authorization officer's signature of x, and Hash($\cdot$) is a partial information hiding hash function [Can97] (e.g., Sig(Hash($\cdot$)) is a content-hiding signature) or a random oracle (See [BR93]).

Biometric verification process: When a user presents herself/himself and the card with the information described above, M biometric templates are independently generated for the user. Majority decoding is applied to the M biometric vectors to obtain the user's k bit template $\vec{T}'$. Error correction is performed on codeword $\vec{T}'||\vec{C}$ to obtain the corrected biometric $\vec{T}''$. The signature Sig(Hash(NAME, ATTR, $\vec{T}''||\vec{C}$)) is then verified. Successful signature verification implies the user passed the identification step. For simplicity of exposition, we assume that occasional rejection of a valid user is acceptable (the user would simply repeat the scan). In applications where rejection of a valid user is not acceptable, the parameters of the system can be changed so that such an event has negligible probability. Determining the correct parameters in such a case involves bounding the area under the tail of a binomial distribution (or a Normal approximation to the binomial distribution via the Central Limit Theorem).

Proof of security and in particular the choice of hash functions were discussed in [DFM98]. We now discuss how majority decoding will provide for enhancement to the system.

2 The role of Error Correction

Error correction is performed at two crucial points in the scheme described: The majority decoding at the point of biometric template generation (when the user token is generated) and at the point of verification, when the user presents herself and the token is issued by an authorization center.

To help understand the role of error correction at the various points in the process, we need to consider the probability of per bit error in a measured

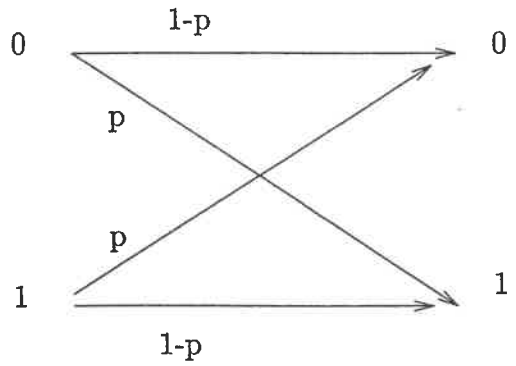


Figure 1: A Binary Symmetric Channel

iris. In [Dau93], the probability of mismatch in each corresponding bit of two samples from the same person, taken at different times, was found to be 0.084, while the probability of a mismatch of corresponding bits of iris scan for two different individuals was found to be 0.45, i.e. about one in two, approaching a random toss of a coin. In addition, each bit of the iris scan appears to be a random variable over a population of different individuals. The errors in the measurement of one individual have been found to be independent over the entire iris scan. Thus one can view errors in the measurements as a binary symmetric channel (see Figure 1).

With this important assumption, which has been empirically validated ([Dau92],[Dau93]), it is possible to apply error correction at the point of acquisition of the iris code. One possible error correction that can be applied is majority decoding of M samples taken at the time of enrollment (or verification). Applying majority decoding at the time of acquisition of the iris scan, one obtains a "reference" iris scan for which we then compute a set of check digits that can correct t errors in the iris scan using an $[n, k, d]$ error correcting code. We note that the check digits, which will be stored on the user's card, will not have any errors in them when error correction is performed. Thus all the t errors that the ECC will correct will come from the k bits of the iris scan, namely the information bits of the error correcting code.

At the point of verification, the user presents herself and the data from, say, a magnetic card, containing the check digits that will be used in correct-

ing a (possibly) erroneous iris scan vector. We now consider the method of acquiring the iris scan at the point of verification.

With majority decoding, using M samples per bit, the probability of an error in each bit then is

$$Prob(M/2 \text{ or more errors}) = P_e = \sum_{i=M/2}^{M} \binom{M}{i} p^i q^{M-i} \qquad (1)$$

What we are interested in is $n * P_e$, the expected number of errors in a final biometric. Let M_r and M_v be the sampling rates at registration and verification times, respectively. Let P_{e-r} and P_{e-v} be the expected final per bit error rates of the biometric at registration and verification times, after majority decoding. Note that P_{e-r} and P_{e-v} decrease as M_r and M_v increase, respectively. To protect against t errors at verification time, then we choose M_r such that

$$n * P_{e-r} < 1$$

For verification time, we choose M_v such that

$$n * P_{e-v} < t$$

When we compute the check digits C, they are stored on the user's card. The check digits protect against t errors when the biometric is read at the verification point. Observe that t cannot be too large (i.e. if we were to correct, for example, 32 percent errors in the 2048 vector, then we risk accepting an imposter). In fact, for reasons of space efficiency we need to reduce the error rate to something that does not lead to substantial expansion of the data on the storage device carried by the user. In addition, for computational efficiency the ECC needs to be reasonably fast at the point of verification.

At the verification point we consider the cases:

1. The presenter is authentic.In this case, the biometric read, using M_v samples to compute the final biometric, should result in $n * P_{e-v} < t$ errors. The errors are then corrected using the C check digits from the user's card. If the biometric has more than t errors, it is rejected.

2. The presenter is an imposter.Empirical data shows that the average Hamming distance between imposters and authentics is almost $n/2$. Thus an imposter's biometric presented for correction, along with C

will be rejected with high probability, since the ECC will correct at most t errors.

The error rates in the biometric and the error correction capability of the ECC are critical to secure biometric computation. Using majority decoding, we stabilize the biometric at both the registration point and the verification point to allow the correction of the biometric at the verification point to a reference value. We control the errors corrected at the verification point to achieve the most efficient computation at the verification point, since that is where delays can be problematic.

3 Artificial Iris

While it is believed that artificial devices will not likely succeed at the present time, this may not be the case in the near future, as lcd devices improve in density, color accuracy and other features. If an artificial iris is feasible, then we need to consider defending against one.

Consider an artificial iris that targets one individual. Also, assume that the artificial iris can get as close as distance d_a to a target. In such a case we choose M_r, as before, such that

$$n * P_{e-r} < 1$$

and choose M_v such that

$$t < \frac{d_a - s}{2}$$

where s is a security parameter chosen to make the probabilities of false positives and false rejections acceptable.

4 Concluding Remarks

The role of error correcting codes in reliable, secure and authenticated communication and applications ranging from identification to electronic commerce is an important one. Already there is a significant amount of work, such as the McEliece public key cryptosystem, the Shamir threshold key sharing and the Davida-DeMillo-Lipton key sharing, that has been a link

between the two seemingly separate areas of coding theory and cryptography. This work combines the two to present an identification system that facilitates the use of non-invasive biometrics using an off-line identification scheme that is secure and reliable. Error correction is an important part of this work. Without *ECC* this work would have been more difficult. For example, in the schemes implemented in online systems where a user database of biometrics is stored, the systems are slow even for a moderate number of users. If the number of users is large, in the millions, earlier schmes may very well be too slow for many applications.

References

[BAW96] F. Bouchier, J. S. Ahrens, and G. Wells. Laboratory evaluation of the iriscan prototype biometric identifier. Technical Report SAND96-1033, Sandia National Laboratories USA, April 1996.

[Ber68] E. R. Berlekamp. *Algebraic Coding Theory.* McGraw-Hill, 1968.

[BR93] M. Bellare and R. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *Proceedings of the 1st ACM conference on Computers and Communications Security*, 1993.

[Can97] R. Canetti. Towards realizing random oracles: Hash functions which hide all partial information. In *Advances in Cryptology. Proc. of Crypto'97*, pages 455–469, 1997.

[Dau92] J. Daugman. High confidence personal identifications by rapid video analysis of iris texture. In *IEEE International Carnahan Conference on Security Technology*, pages 50–60, 1992.

[Dau93] J. Daugman. High confidence personal identifications by a test of statistical independence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):648–656, November 1993.

[DFM98] G. I. Davida, Y. Frankel, and B. J. Matt. On enabling secure applications through off-line biometric identification. In *1998 IEEE Symposium on Security and Privacy*, pages 148–157, 1998.

[HMW90] J. P. Holmes, R. L. Maxell, and L. J. Wright. A performance evaluation of biometric identification devices. Technical report, Sandia National Laboratories, July 1990.

[MS78] F. J. MacWilliams and N. J. A. Sloane. *The theory of error–correcting codes.* North – Holland Publishing Company, 1978.

[PW88] W. W. Peterson and E. J. Weldon. *Error Correcting Codes.* The MIT Press, 1988.

[Wil96] G. O. Williams. Iris recognition technology. In *IEEE International Carnahan Conference on Security Technology*, pages 46–59, 1996.

Verifiable Self-Certified Public Keys *

Seungjoo Kim[1], Soo-Hyun Oh[1], Sangjoon Park[2] and Dongho Won[1]

[1] Dept. of Information Engineering, Sungkyunkwan Univ.,
300 Chunchun-dong, Suwon, Kyunggi-do, 440-746, Korea
E-mail : {sjkim, shoh, dhwon}@dosan.skku.ac.kr
URL : http://dosan.skku.ac.kr/~sjkim/
[2] #0710, ETRI, Yusong P.O.BOX 106, Taejon, 305-600, Korea
E-mail : sjpark@dingo.etri.re.kr

Abstract. Self-certified public keys, introduced by Girault allow the authenticity of public keys to be verified implicitly during the use of the keys. This paper first presents new concept of verifiable self-certified public keys and provides concrete examples satisfying our conditions. Verifiable self-certified public keys combine the benefit of certification-based schemes and Girault's self-certified public keys. Furthermore, we will also cryptanalyze Petersen's pseudonymous self-certified keys.

1 Introduction

In [1], by W. Diffie and M. Hellman, the genius notion of public key cryptosystems was proposed. In such schemes, every user has a key pair (secret key s, public key P), where public keys don't need to be protected for confidentiality. But this "publicity" makes them particularly vulnerable to active attacks.

The obvious solution to this fraud is to provide witness W, that confirms P is really the public-key of user I, by the authority. In this simplest approach, W, often called "certificate", takes the form of digital signature of the pair (I, P). While this solution reaches Girault's trust-level 3 [3], this approach requires the additional cost of storage and computation for P and W.

This leads to an "identity-based" schemes, introduced by A. Shamir [16]. The advantage of this scheme is that P is nothing but I and W is nothing but the corresponding s. There is, however, a penalty for this gain : the authority knows users' secret keys, since s is computed from I and some trapdoor originated by the authority. (level 1) [4].

In [2], M. Girault proposed more sophisticated technique, called "self-certified public keys", which is intermediary between certification-based and identity-based ones. In such schemes, (contrary to identity-based schemes) each user I

* This work was partially supported by KSEF(Korea Science and Engineering Foundation) under projects 976-0900-001-2 and 97-01-00-13-01-5.

[3] At level 3, the authority does not know users' secret keys, and it can be proven that it generates false witnesses of users if it does so.

[4] At level 1, the authority knows the users' secret keys and therefore can impersonate any user at any time without being detected.

Table 1. Comparison of several types of schemes by the form of witness. I is user's identification string (or identity), (s, P) means user's key-pair (secret key, public key), and G is a witness that P is really the public key of user I, and not the one of an impostor I'.

	System	Public key (generation)	Secret key (generation)	Witness (generation)	Trust level	Verify
Public-key	(s, P)	P : user	s : user			
Certification-based schemes	(I, s, P, W)	I, P : user W : authority	s : user	authority's sig. on (I, P)	3	explicit
Identity-based schemes	(I, s) $P = I, W = s$	I	s : authority	authority	1	(implicit)
Self-certified public keys	(I, s, P) $W = P$	(I, P)	s : user	authority	2, 3	implicit
Self-certified identity	(I, s) $P = I = W$	I	s : user	authority	2, 3	(implicit)

chooses his secret key s, and creates his real public key P'. Then the authority computes the witness W from the pair (P', I), in such a way that W may not be computed without the knowledge of some trapdoor. Now, (contrary to certification-based schemes) the witness W is embedded in the public key P itself, and therefore does not take the form of separate value. This achieves level 2 or 3 [5].

Furthermore, S.J. Park *et al.* introduced "self-certified identity-based schemes", combining the advantages of certification-based and identity-based schemes [10][11]. In their schemes, like identity-based schemes, $P = I = W$ and like certification-based schemes, each user chooses his own s (level 2 or 3). [6] The properties of these schemes are in the Table 1. [7]

However, one disadvantage of self-certified public keys is their repudiability [12]. In certification-based schemes, the authenticity of the public key P can be verified directly after knowing a witness W, but, in self-certified schemes, the authenticity is verified at the time, when the key is used for encryption, signature verification, key exchange or any other cryptographic use. For example, if the verification of a digital signature fails using a self-certified public key it is uncertain, whether the signature or the public key is incorrect.

2 Girault's self-certified public keys

For simplicity we only describe a scheme using RSA/Rabin digital signature scheme. In the set-up phase, the CA(Certificate Authority) chooses a RSA mod-

[5] At level 2, the authority does not know users' secret keys, but the authority can still impersonate a user by generating false witnesses.

[6] [6], [7] and [5] have presented similar notion, called "*one-time self-certified public keys*".

[7] The basic idea of categorization has been borrowed from [2].

ulus $n = p \cdot q$ such that p and q are large prime integers, generates an integer e coprime to $p-1$ and $q-1$, and the inverse d of e modulo $(p-1)\cdot(q-1)$. Then it computes an integer α of maximal order in the multiplicative group $(Z/nZ)^*$. The CA makes n, e and α public, whilist keeps p, q and d secret.

The key generation phase consists of two steps. First, each user randomly chooses a secret key x, computes his public key $y = \alpha^{-x} \pmod n$ and gives y to the CA. Then he proves to the CA that he knows x without revealing it.

Afterwards the CA computes the witness as a RSA signature of the modular difference of α^{-x} and user's identity I :

$$w = (y - I)^d \pmod n.$$

So the following equation holds :

$$w^e + I = y \pmod n. \tag{1}$$

However, in Girault's self-certified keys, anyone can obtain the pair (y, w) satisfying the equation (1) by only the known-key attack :

$$w \in_R Z_n,$$

$$y = w^e + I \pmod n.$$

So, if the verification of a digital signature fails using a self-certified public key it is uncertain, whether the signature or the public key is incorrect.

3 Verifiable self-certified public keys

Definition 1. (*verifiable self-certified public keys*) Verifiable self-certified public key scheme satisfies the following two conditions ;

1. (*self-certification*) The witness is equal to the public key. The user's attributes – identity, secret key, public key, etc – satisfy a computational unforgeable relationship, which is verified implicitly during the proper use of keys in any cryptographic protocol.
2. (*verifiability*) Furthermore, if necessary, there is an efficient way to verify the authenticity of the public key after knowing a witness.

3.1 A scheme using RSA digital signature scheme

Set–up

- an integer $n \geq 2^{512}$ as the product of two large distinct random primes p and q of almost the same size, such that $p = 2p' + 1$ and $q = 2q' + 1$, where p' and q' are also prime integers,
- a base $\alpha \neq 1$ of order $r = p' \cdot q'$ (i.e., $\alpha^r = 1 \pmod n$),

- a large integer $u < r$, and
- a one-way hash function h which will output positive odd integers between 2^{128} and p'(and q'). This can be easily implemented by flipping the least significant bit, if even.

The CA makes n, α, u, f public, keeps p and q secret.

Verifiable self-certified key generation

Alice who visits the CA receives a witness w_A, if her legitimacy is accepted by the CA. The witness w_A is generated as follows :

1. Alice chooses her secret key $x_A < u$ as a random integer and computes her public key as
$$y_A = \alpha^{-x_A} \pmod{n}.$$
Next, Alice visits the CA and gives y_A to it.
2. The CA, after having checked the Alice's identity, prepares the corresponding ID_A, and computes
$$w_A = (y_A - ID_A)^{h(y_A)^{-1}} \pmod{n}. \quad (2)$$
CA transmits w_A to Alice. [8]
3. Then, Alice's verifiable self-certified public key is w_A and $e_A = h(y_A)$.

Now, this set-up will enable Alice to explicitly check the authenticity of public key y_A, since , given the pair (y_A, w_A), the following equation holds :

$$w^{h(y_A)} + ID_A = y_A \pmod{n}.$$

Key exchange protocol

Let (ID_A, x_A, w_A, e_A) be the attributes of Alice, (ID_B, x_B, w_B, e_B) those of Bob. They can simply exchange an authenticated key by choosing (See Fig.1) :

$$K_{AB} = (w_B^{e_B} + ID_B)^{x_A} = (w_A^{e_A} + ID_A)^{x_B} = \alpha^{-x_A \cdot x_B} \pmod{n}.$$

[8] Note that in [14], the (not verifiable, in our sense) equation $w_A = y_A^{h(ID_A)^{-1}} \pmod{n}$ was used as an alternative to the congruence (2). But this doesn't meet Girault's notion of self-certified public keys with trust-level 3, because there is the possibility for each user to create other witnesses corresponding to his identity, after he has been given one by the authority. As a consequence, a judge cannot distinguish between a cheating authority and a cheating user, so Saeednia's proposal only achieves level 2 [4].

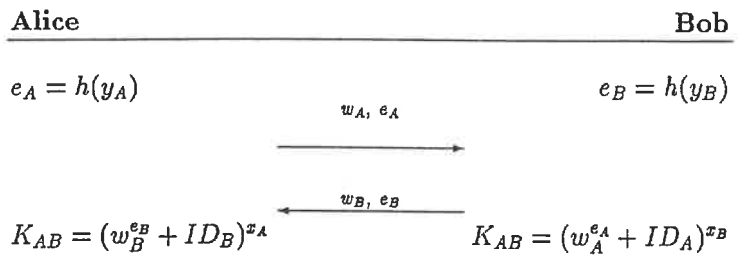


Fig.1 : Verifiable self-certified key exchange protocol

This protocol is related to Girault's one. But contrary to it, if the key exchange protocol fails, each user can verify the authenticity of the public key by computing :

$$\tilde{y}_{A(B)} = w_{A(B)}^{e_{A(B)}} + ID_{A(B)} \pmod{n},$$
$$e_{A(B)} \stackrel{?}{=} h(\tilde{y}_{A(B)}).$$

3.2 A scheme using Schnorr digital signature scheme

Set–up

- large primes p, q such that $|p| \geq 512$, $|q| \geq 140$ and $q|(p-1)$,
- a generator α of a multiplicative subgroup of Z_p^* with order q (i.e., $\alpha^q = 1 \pmod{p}$),
- a one-way hash function h of output length ≥ 128,
- CA chooses a random number $x_{CA} \in_R Z_q^*$ as her secret key and computes her public key $y_{CA} = \alpha^{x_{CA}} \pmod{p}$.

The CA makes p, q, α, h and h public, keeps x_{CA} secret.

Verifiable self-certified key generation [9]

1. After confirming her legitimacy, CA generates a random number $\tilde{k}_A \in_R Z_q^*$ and computes $\tilde{r}_A = \alpha^{\tilde{k}_A} \pmod{p}$. CA transmits $\tilde{r}_A$ to Alice.
2. Alice chooses a random $a \in_R Z_q^*$ and computes $r_A = \tilde{r}_A \cdot \alpha^a \pmod{p}$. Alice gives ID_A and r_A to the CA.
3. CA computes the signature

$$\tilde{s}_A = x_{CA} \cdot h(ID_A, r_A) + \tilde{k}_A \pmod{q}.$$

This value $\tilde{s}_A$ is transmitted to Alice.

[9] The proposed scheme is constructed from the Petersen's self-certified key issuing protocol (trust level 3 version) [12].

4. Alice obtains her secret key as

$$x_A = \tilde{s}_A + a \pmod{q}.$$

5. Furthermore, Alice computes $r'_A = g^{k'_A} \pmod{p}$ with $k'_A \in_R Z_q^*$, and generates (e'_A, s'_A) as

$$e'_A = h(r'_A)$$
$$s'_A = k'_A - x_A \cdot e'_A \pmod{q}$$

Now, Alice's verifiable self-certified public key is r_A, e'_A and s'_A.

Then, given the public parameters y_{CA}, ID_A, r_A, the following equation holds :

$$y_A = \alpha^{x_A} = y_{CA}^{h(ID_A, r_A)} \cdot r_A \pmod{p}.$$

Furthermore, if encryption, signature verification, key exchange or any other cryptographic use fails, given (e'_A, s'_A), user can verify the authenticity of the public key by checking :

$$e'_A \stackrel{?}{=} h(\alpha^{s'_A} \cdot (y_{CA}^{h(ID_A, r_A)} \cdot r_A)^{e'_A} \pmod{p}).$$

Comparison among different types of witnesses : From the point of computational advantage, the verifiable self-certified public keys reduce the amount of computational work over the certification-based schemes [10] and, from the point of organization, if need, our scheme can be explicitly verified unlike the self-certified public keys (See Table 2. and 3., where, a method used by Kaliski[3] is adopted to assess the amount of computational work. Numbers means the amount of work to perform modular multiplication in 512 bits modulus, $WI(b)$ means the amount of work to perform b-bit modular inversion, and $WH(b)$ means the amount of work to compute a hash function with a b-bit long input. Refer to [8] for detail).

4 Pseudonymous self-certified keys

In [2], Girault distinguished several types of certification schemes, under different trust level – three levels (1, 2 and 3) –. Also, in [12], Petersen extended it to level 4 (called "pseudonymous self-certified keys").

Definition 2. (*pseudonymous self-certified public keys*) The authority issues a self-certified public key to a user with pseudonymous PS, such that the real identity of the user is hidden to the authority. Nevertheless, all operations using the same pseudonym are linkable for any person.

[10] If very low public exponent (e.g., $e = 3$) is used in the original scheme, we can use $w_A = (y_A - ID_A || h(y_A))^e \pmod{n}$ or $w_A = (y_A - ID_A \oplus h(y_A))^e \pmod{n}$ etc, as an alternative to the congruence (2).

Table 2. Performance of verifiable self-certified public keys for the RSA scheme ($|n| = 512, |e| = |h(\cdot)| = 128$).

		Total length	Computational work	
			Verify (implicit)	Verify (explicit)
RSA	Certification-based schemes [13]	1024	not available	$160 + WH(\|n\| + \|ID\|)$
	Self-certified public keys [2]	512	160	not available
	Ours	640	160	(optional) $WH(\|n\|)$

Table 3. Performance of verifiable self-certified public keys for the Schnorr scheme ($|p| = 512, |q| = 140, |h(\cdot)| = 128$).

		Total length	Computational work	
			Verify (implicit)	Verify (explicit)
Schnorr	Certification-based schemes [15]	780	not available	$217 + WH(\|p\| + \|ID\|)$
	Self-certified public keys [12]	512	$161 + WH(\|p\| + \|ID\|)$	not available
	Ours	780	$161 + WH(\|p\| + \|ID\|)$	(optional) $217 + WH(\|p\|)$

So, pseudonymous self-certified keys have applications, such as electronic cash, etc. Petersen *et al.* showed the pseudonymous self-certified key issuing protocol, by the use of blind Schnorr signature scheme [9] (See Fig.2). Here, Alice uses the pseudonym PS_A instead of her real identity ID_A. Furthermore, CA uses a different certified keypair $(\tilde{x}_{CA}, \tilde{y}_{CA})$ in this protocol to distinguish it from the protocol above.

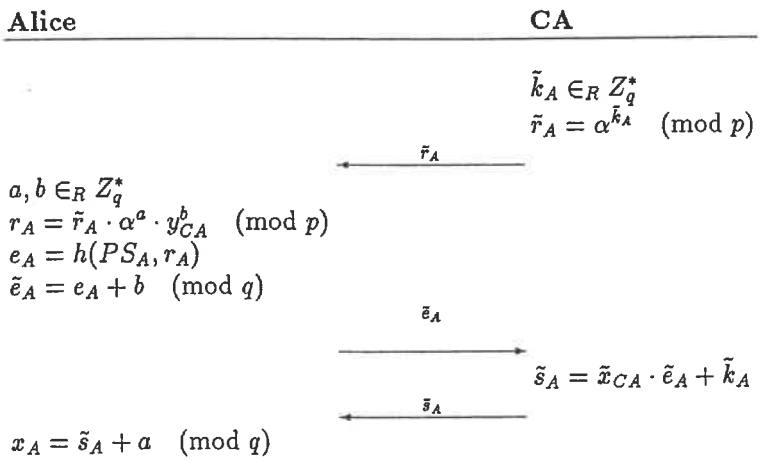


Fig.2 : Petersen's key issuing protocol

However, in Petersen's construction (at level 4), a cheating one can use Alice's pseudonym PS_A instead of his own pseudonym without any detection. So, as in Saeednia's proposal, a judge cannot distinguish between a cheating CA and a cheating user, and then Petersen's construction doesn't reach even the level 3.

In order to prevent a cheating user from having a chance to get the self-certified public key with same pseudonym, we split the key generation function into the CA and the RA(Registration Authority). The RA will register information about user identity, pseudonym, etc, and will blind the CA's signatures.

Pseudonymous self-certified key generation [11]

1. CA chooses $\tilde{k}_A \in_R Z_q^*$, computes
$$\tilde{r}_A = \alpha^{\tilde{k}_A} \pmod p,$$
and send $\tilde{r}_A$ to Alice.
2. Alice chooses a random $a \in_R Z_q^*$ and sends ID_A, PS_A and
$$\tilde{r}_A \cdot \alpha^a \pmod p$$
to RA in a secure manner (e.g., with non-deterministic encryption scheme).
3. The RA, after having checked the Alice's identity, registers PS_A, and computes
$$b \in_R Z_q^*,$$
$$r_A = (\tilde{r}_A \cdot \alpha^a) \cdot y_{CA}^b \pmod p,$$
$$\tilde{e}_A = h(PS_A, r_A) + b \pmod q.$$
Then, RA gives $\tilde{e}_A$ and a signature on it to Alice.

[11] This scheme is similar to the fair version of Petersen's pseudonymous self-certified key issuing protocol. But, in [12], they didn't mention the above problem.

4. Alice presents $\tilde{e}_A$ and signature to CA.
5. After having verified the RA's signature, CA sends

$$\tilde{s}_A = \tilde{x}_{CA} \cdot \tilde{e}_A + \tilde{k}_A \pmod{q}$$

to Alice.
6. Now, Alice computes her secret key as

$$x_A = \tilde{s}_A + a \pmod{q}.$$

And, her corresponding public key is computed as

$$y_A = \alpha^{x_A} = \tilde{y}_{CA}^{h(PS_A, r_A)} \cdot r_A \pmod{p}.$$

Here, the RA cooperates in producing the above pseudonymous self-certified keys in a way that prevents the cheating user from using the same pseudonym without any detection. The proposed system achieves anonymity for the honest user, but only by the cooperation of RA (when presented with an appropriate court order), user anonymity can be revoked or suspended.

5 Conclusion

We proposed the new notion of verifiable self-certified public keys. Verifiable self-certified keys have a computational advantage over certification-based schemes and, if necessary, can be explicitly verified unlike self-certified public keys.

Moreover, we showed that the pseudonymous self-certified key scheme presented by H. Petersen and P. Horster doesn't meet the trust-level 3, and proposed the repaired scheme.

References

1. W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Trans. Inform. Theory*, vol. 22, 1976, pp. 644-654
2. M. Girault, "Self-certified public keys," *Advances in Cryptology* (Proceedings of *EuroCrypt*'91), Lecture Notes in Computer Science, vol. 547, Springer-Verlag, 1991, pp. 490-497
3. B.S. Kaliski, "A response to DSS," Nov. 1991.
4. Seungjoo Kim, Soo-Hyun Oh, and Dongho Won, "Cryptanalysis and enhancement of Saeednia's key-exchange protocols," *Proc. of KICS (Korea Institute of Communication Sciences) Conference*, Vol. 17/No. 2, Korea, (1998), pp. 1001-1004
5. Seungjoo Kim, Soo-Hyun Oh and Dongho Won, "One-time self-certified public keys, revisited," *Proc. of ICISC'98, International Conference on Information Security and Cryptology*, 1998, To appear.
6. C.H.Lim and P.J.Lee, "Authentication and digital signature schemes using one-time self-certified public keys", *Proc. of JCCI'95, Joint Conference on Communication & Information*, 4/1995, pp.33-36.

7. C.H.Lim and P.J.Lee, "A method for obtaining authentication and digital signature schemes using one secret key", *Korea patent, Serial No. 95-10019* 27/4/1995.
8. M. Mambo, K. Usuda, and E. Okamoto, "Proxy signatures: Delegation of the power to sign messages," IEICE Trans. Fundamentals, vol.E79-A, no.9, 1996, pp.1338-1354
9. T. Okamoto, "Provable secure and practical identification schemes and corresponding signature schemes," LNCS 740, *Advances in Cryptology* : Proc. Crypto'92, Springer, (1992), pp.31-53
10. Sungjun Park, Chungryong Jang, Kyungsin Kim and Dongho Won, "A "paradoxical" identity-based scheme based on the γ^{th}-residuosity problem and discrete logarithm problem," *An International Conference on Numbers and Forms, Cryptography and Codes*'94, 1994
11. Sungjun Park and Dongho Won, "A "paradoxical" identity-based scheme based on the γ^{th}-residuosity problem and discrete logarithm problem," *Journal of The Korean Institute of Information Security and Cryptology*, vol. 4/No. 2, 1994, pp. 113-118
12. H. Petersen and P. Horster, "Self-certified keys – concepts and applications," *Proc. 3. Conf. on Communications and Multimedia Security*, September 22-23, 1997, Chapman & Hall
13. R.L. Rivest, A. Shamir and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *CACM*, Vol.21, no.2, Feb. 1978, pp. 120-126
14. S. Saeednia, "Identity-Based and Self-Certified Key-Exchange Protocols," *Information Security and Privacy* (Proceedings of *ACISP*'97), Lecture Notes in Computer Science, vol. 1270, Springer-Verlag, 1997, pp. 303-313
15. C.P. Schnorr, "Efficient identification and signatures for smart cards," *Advances in Cryptology* (Proceedings of *Crypto*'89), Lecture Notes in Computer Science, vol. 435, Springer-Verlag, pp. 239-252
16. A. Shamir, "Identity-based cryptosystems and signature schemes," *Advances in Cryptology* (Proceedings of *Crypto*'84), Lecture Notes in Computer Science, vol. 196, Springer-Verlag, 1985, pp. 47-53

This article was processed using the LaTeX macro package with LLNCS style

Authentication frauds from the point of view of rate-distortion theory

Andrea Sgarro
DSM - University of Trieste, Italy - *sgarro@univ.trieste.it*

Abstract. Surprisingly, the rate-distortion function appears in a powerful lower bound to the probability of an authentication fraud; we argue that the link between authentication theory and rate-distortion theory is not just formal, but also conceptual. We show that a source code with zero distortion-level provides a strategy to attack an authentication system.

1. The rate-distortion function in authentication frauds

Basically, authentication is a game with three participants: a transmitter, a receiver and an intruder [1]. Transmitter and receiver communicate over a publicly exposed channel; their interest is safeguarding the communication authenticity, while the intruder observes the communication channel and acts over it attempting to deceive the receiver into accepting a fraudulent cryptogram as genuine. For a formal definition of an *authentication code* cf Appendix 1; here we just recall that its basic elements are a set of (clear)*messages* (or state-sources) to be encoded, a set of encoding rules, or *keys*, and a set of *codewords* (or encoded messages, or cryptograms). Actually, in the sequel the term used will be *codeletter* rather than codeword: this is unusual, but more convenient in our context. The binary *authentication matrix* specifies which codeletters are authenticated by which keys, in the sense that there is a message encoded to that codeletter under that key. Unlike encoding, decoding is assumed to be deterministic(ally successful).

A powerful lower bound to the probability of an authentication fraud was given in [2]; it subsumes all lower bounds, both combinatorial and information-theoretic, given in the literature, which are re-obtained either literally or as straightforward corollaries (weakenings). The fraud considered in [2] is rather more "abstract" than those usually dealt with (impersonation, substitution, chosen-content attack, etc.), which explains why the bound is so general. Actually, the starting point for [2] had been the wish to unify derivations of bounds which were often quite similar, but had each time to be obtained separately almost from scratch, which was boring, indeed. In [2] an "abstract" structure was introduced, called a *fraud scheme*, which immediately particularizes to specific frauds. Its elements are a random variable Z called the (scheme-)*key*, which represents the authentication code key K as known to the intruder, and a binary matrix ϕ called the *fraud matrix*, which tells which codeletters cheat which keys under the given attack: $\phi(z, y) = 1$ iff codeletter y cheats scheme-key z. We stress that in a fraud scheme all trace is lost of source states and of encoding procedures: the point of view is that of the intruder who wants to insert an illegal codeletter and have it passed by the authentication system. In

Partially supported by MURST

impersonation, say, the fraud matrix ϕ coincides with the authentication matrix χ of the code, and so any codeletter authenticated by the current key brings about a fraud. In *chosen-content attacks*, as introduced in [3] (cf also [2]; these attacks were re-introduced later but independently in [4]) fraud succeeds only if the decoded clear-message belongs to a special target set $\mathcal{M}$ of messages favourable to the intruder; in this case the matrix ϕ is obtained from ϕ by "cancelling" (turning to zeroes) all the ones which correspond to decoding outside $\mathcal{M}$. In *substitution*, instead, a re-insertion of the intercepted codeletter c does not cause any fraud, and so the ones in the c-column of the code's authentication matrix χ are cancelled in the fraud matrix ϕ (to be precise, whenever the code has splitting, one should also cancel, for each key, the homophones of the intercepted codeletter; cf appendix 1). In impersonation and in chosen-content attacks the scheme-key Z and the code-key K coincide (have the same probability distribution: $dist(Z) = dist(K)$), while in substitution the distribution of the scheme-key is obtained by conditioning the distribution of the code-key to the information possessed by the intruder: $dist(Z) = dist(K|C = c)$, C being the random *legal* codeletter just intercepted. The (abstract, general) fraud probability is defined as

$$P = \max_{y} \text{Prob}\{\phi(Z, y) = 1\} \tag{1}$$

Below we need the rate-distortion function $R(S, d)$ for probability distribution $dist(S)$ (S is the random source letter or primary letter), for distortion matrix d, and for a distortion-level Δ equal to zero:

$$R(S, d, 0) \equiv R(S, d) = \min I(S \wedge Y)$$

(the distortion level 0 will be omitted in our notation; to avoid trivial specifications we delete zero-probability primary letters and no-zero columns in d, if any; $I(S \wedge Y)$ is mutual information, $I(S \wedge Y) = H(S) + H(Y) - H(S, Y)$ in terms of Shannon entropy; the minimum is taken over all random couples SY where Y is a random secondary letter constrained to give $d(S, Y) = 0$ with probability 1; for the basics of rate-distortion theory, which are here taken for granted, cf e.g. [5] or [6]). Now, the lower bound of [2] is:

$$P \geq 2^{-R(Z, \phi^*)} \tag{2}$$

where ϕ^* is the binary complement of ϕ (the star indicates that ones and zeroes are swapped). Appendix 2 is meant to convince the reader of the power of the bound by showing how one soon re-finds bounds met in the literature, e.g. Simmons bound, or the "historical" combinatorial bound of Gilbert, Mac Williams and Sloane [7]. The appearance of the rate-distortion function R in a context where we are certainly not trying to reproduce (encode) keys by means of codeletters (meant to cheat the system, on top of that!) was judged as "intriguing". Below we show that this coincidence is not just a formal one: surprisingly, authentication theory has an meaningful overlapping with rate-distortion theory.

We begin by revisiting the authentication problem in a new garment, which is rate-distortion theoretic. Assume that we want to code by means of a reproducing sequence $\underline{y}$ the random primary sequence $\underline{S}$, which however we cannot observe; what we need is sort

of a "universal" reproducing sequence, such that $\text{Prob}\{d_n(\underline{S}, \underline{y}) > \Delta\}$ be small. In the special case $\Delta = 0$ inequality (2) ensures that:

$$\exists\, \underline{y} : \quad \text{Prob}\{d_n(\underline{S}, \underline{y}) \leq \Delta\} \geq 2^{-R(\underline{S}, d_n)} = 2^{-nR(S,d)}$$

(for the last equality to hold one has to assume that the primary source is stationary and memoryless).

2. Authentication frauds and source coding at zero distortion-level: an asymptotic view

In rate-distortion theory the rate-distortion function appears in an asymptotically correct equality, rather than in a finite-length incorrect (though tight) inequality as (2) is: this is why it would be nice to have a Shannon-type asymptotic theorem such as to show that bound (2) can be manipulated to an asymptotically correct result. The direct theorem we sketch below will be limited to the case of (row-)*uniform* fraud matrices (i.e. the number of ones per row, i.e. per key, is constant); the limitation is not as severe as it seems, since uniform matrices are those usually met in practice. We stress that our chief aim is making explicit the link between the two theories, and this will suggest the choice of our setting, which is standard in rate-distortion theory, but not necessarily the most interesting from the point of view of authentication. In a final remark we shall shortly comment on how the needs of authentication might influence rate-distortion theory, and conversely how the ideas of the latter of might have a bearing on authentication coding.

We start by an asymptotic lower bound, i.e. by a converse theorem. Not to depart from the usual setting of rate-distortion theory we assume that the key source is stationary and memoryless, even if this assumption is reasonable only for frauds like impersonation or chosen-content attacks, where the scheme-key and the code-key coincide (stationarity is unacceptable in the case of substitution, since the distribution of the scheme-key depends on the intercepted codeletter). At each time-instant a value of the key (i.e. a *key-letter*) is chosen by the "system" and the intruder tries to insert a codeletter which brings about a fraud and which he hopes to be authenticated by the current value of the key. (The true key-sequence of the code is communicated to the *legal* user by means of a secure channel, as is standard in unconditionally secure cryptography; it makes sense to assume that he knows *a fortiori* also the true key-sequence of the *scheme*.) The assumption to follow is itself not always the most natural from the point of view of authentication (cf the final remark for a different assumption). Namely, we assume that the fraud succeeds only if it succeeds at all time instants; so, we are thinking of "multiple checks": the user is accepted as legal (and served) only when he has been able to pass a long sequence of checks required by the system; as soon as he fails once, the system turns him down. In this situation the fraud matrix ϕ_n for n-sequences is obtained by a min operation, $\phi_n(\underline{z}, \underline{y}) = \min_i \phi(z_i, y_i)$, i.e. the distortion matrix ϕ_n^* for n-sequences (the binary complement of the former) is obtained from ϕ^* by a max operation, or, which amounts to the same at zero-distortion (only the dichotomy zero/non-zero counts) by an arithmetic average, as is standard in rate-distortion theory. We have already used the well-known fact that under our assumptions $R(Z^n, \phi_n^*) = nR(Z, \phi^*)$. So, the lower bound becomes:

$$P_n \geq 2^{-nR(Z,\phi^*)} \tag{3}$$

We go to a direct theorem. We assume that the scheme-key Z is uniform; then, in the case of a row-uniform matrix ϕ with k ones per row, (2) soon weakens to the *combinatorial* bound

$$P \geq \frac{k}{|Y|} \tag{4}$$

(to compute size $|Y|$, first delete codeletters y with an all-zero ϕ-column). If Φ is *doubly*-uniform and has, say, r ones per column, this is no weakening; the minimum in the definition of the function R can be made explicit and bound (2) holds with equality: $P = 2^{-R(Z,\phi^*)} \equiv \frac{k}{|Y|}$. Now, whenever one finds an *incidence structure* ϕ such as to meet both requirements (each row has the same number k of ones; each column has the same number r of ones), then also ϕ_n is doubly-uniform (ϕ_n has exactly k^n ones per row and r^n ones per column); such a structure leads to optimal schemes for which (3) holds with equality: $P_n = 2^{-nR(Z,\phi^*)}$, or:

$$-n^{-1} \log P_n = R(Z, \phi^*)$$

(As a matter of fact, much more refined combinatorial designs have been needed in the literature when constructing actual authentication codes, and not just "abstract" schemes.) So far, this is an exact result; in the spirit of Shannon theory one might want to go further. Here we only quickly hint at how this might be accomplished, omitting the technical details of the derivation. Just observe that the uniformity of the key is "asymptotically true" when the source is stationary and memoryless, while the constraints for the existence of incidence structures as those we need become "irrelevant" when the sizes involved grow exponentially; so, if one accepts the typical assumptions of Shannon-theoretic block-coding settings, after some toil one arrives at a Shannon-theoretic asymptotic exact result involving the rate-distortion function, though limited to the case when the *rate* of ones per each row of ϕ_n is bound to be "almost" constant.

3. Codebooks and fraud strategies

We now take the point of view of the intruder who wants to pass the multiple checks (n "single-letter" checks); we design a procedure meant to cheat the system and based on a source code for encoding the key source, as in rate-distortion theory. Before, we shortly and sloppily hint at how such source codes work. Source sequences, or *primary sequences*, which in our case are n-sequences of key-letters, are first partitioned into two sets; the first set has an overall negligible probability and gives rise to encoding errors (by the way, this set is a nuisance from the point of view of authentication, and we shall have to come to it again); the (almost equiprobable) primary sequences in the second set (*typical* keys) are further partitioned into subsets, or "clouds", and each cloud is encoded by a codeword, or, more specifically, by a *reproducing sequence*, which properly reproduces each primary sequence in its cloud (has zero-distortion from it; we stress that reproducing sequences are all distinct, without loss of optimality). In our case reproducing sequences are n-sequences of codeletters. Whenever the code is "optimal", the reproducing sequences output by the encoder are "almost" equiprobable: were it not so, further data compression would be feasible, and the code would not be optimal; this in turns means that also the clouds of

primary sequences are themselves "almost" equiprobable. A side observation: a primary sequence might be properly reproduced by two reproducing sequences, and so might be moved from one cloud to another without changing the performance of the source code; this would slightly decrease the probability of the first cloud and slightly increase the probability of the second; however, in an optimal code, clouds are necessarily "almost" equiprobable, and so "double reproducibility" is irrelevant.

Assume that the codebook of such an optimal source code is available to the intruder, even if he does not know which specific key-letters are output by the source, and consequently cannot feed the encoder so as to activate the encoding procedure. Now, each of the reproducing sequences listed in the codebook of the source code can be viewed as a fraud *strategy*, as follows. Before transmission begins, the intruder chooses a reproducing sequence totally at random, and he inserts its letters, one at each time-instant, into the authentication system. If the key is typical (which is almost sure), and if he has been lucky enough to pick the correct reproducing sequence, the fraud succeeds. Since the code is optimal, the total number of available strategies is "small" and approximately equal to $2^{nR(Z,\phi^*)}$; so the probability of having picked the correct strategy is (comparatively) "large" and approximately equal to $2^{-nR(Z,\phi^*)}$, which achieves bound (3). To put it differently: *the codebook of a good zero-distortion code can be interpreted as a family of good strategies meant to carry through an authentication fraud.*

We are forgetting about a set of vanishing probability, namely the set of primary sequences (key sequences) which are badly reproduced. This is a nuisance in authentication, since the fraud probability is itself vanishing, actually exponentially vanishing. So we had better use a zero-error model of rate-distortion theory: fortunately, the zero-error problem in rate-distortion theory is solved. We recall that the place of the rate-distortion function, $R(Z,\phi^*)$, is taken by its maximum over $dist(Z)$, $R(\phi^*) = \max_Z R(Z,\phi^*)$. Significantly enough, a case when $R(\phi^*) = R(Z,\phi^*)$ is the one covered in our "direct theorem" (Z uniform, ϕ doubly uniform); in this case, roughly speaking, zero-error source coding and negligible-error source coding have the "same price". From the point of view of authentication, all this means that the attack strategy based on a zero-distortion zero-error source code is "essentially" optimal.

4. Final remark. In many situations when authentication is needed, a reasonable fraud matrix between sequences had better be based on a max operation: $\phi_n(\underline{z},\underline{y}) = \max_i \phi(z_i,y_i)$; this means that the fraud succeeds whenever it succeeds at least at *one* time-instant; however, this would lead to a queer distortion measure between sequences based on the *minimum* distortion between letters: $d_n(\underline{z},\underline{y}) = \min_i d(z_i,y_i)$. As we have already stressed, zero-error models are particularly interesting for authentication theory. Also, an attack such as substitution makes it interesting to study appropriate non-stationary sources in rate-distortion theory. Conversely, positive distortion levels make sense also in authentication coding, in a situation when the legal user is recognized as such whenever a "sufficiently high fraction" of the received codeletters are authenticated.

Appendix 1: Authentication codes

We define an *authentication code.* One has three finite (non-empty) sets, $\mathcal{X}$, $\mathcal{C}$ and $\mathcal{K}$: $\mathcal{K}$ is

the set of *keys* or *encoding rules*, $\mathcal{X}$ is the set of *clearmessages* or *source states*, and $\mathcal{C}$ is the set of *cryptograms* or *codewords* ($|\mathcal{X}| \leq |\mathcal{C}|$; bars denote size). Two independent random variables K and X are given over $\mathcal{K}$ and $\mathcal{X}$, the *random key* and the *random clearmessage*, respectively. An *encoding function*, or *enciphering function*, is assigned, i.e. a possibly random function f from $\mathcal{K} \times \mathcal{X}$ onto the set of codewords $\mathcal{C}$; if f is not deterministic we say that the code allows for *splitting*, or also that encoding is *homophonic*. The encoding function f defines a new random variable $C = f(K, X)$ over $\mathcal{C}$, the *random cryptogram*. More explicitely: we write $c \in f(k, x)$ to mean that c is a possible cryptogram for x under key c, i.e $\text{Prob}\{C = c | K = k, X = x\} \neq 0$. Conditional probabilities of the latter type specify the *splitting strategy* of the code; when the code is deterministic these probabilities are either zero or one. We assume that there are no collisions when encoding, i.e. $x \neq x'$ implies that $f(k, x)$ and $f(k, x')$ are disjoint sets of codewords. Due to this, the *decoding function*, or *deciphering function*, g from $\mathcal{K} \times \mathcal{C}$ onto $\mathcal{X} \cup \{?\}$ is deterministic ($x = g(k, c)$ iff $c \in f(k, x)$; we set $g(k, c) =?$ whenever c encodes no clearmessage x under key k, i.e. whenever $\text{Prob}\{K = k, C = c\} = 0$; the symbol ? is not in $\mathcal{X}$). An authentication code is a random triple XCK on $\mathcal{X} \times \mathcal{C} \times \mathcal{K}$, or rather, to be very formal, a quintuple (X, C, K, f, g) as above. Observe that, for each fixed key (encoding rule) k, the authentication code boils down to a (possibly probabilistic) source code: so, an authentication code is a family of source codes indexed by the random key. Starting from f we build the *authentication matrix* χ. The authentication matrix χ is a binary matrix whose rows are labelled to $\mathcal{K}$, whose columns are labelled to $\mathcal{C}$, whose values show which key k authenticates which cryptogram c:

$$\chi(k, c) = 1 \iff \exists x \in \mathcal{X} : c \in f(k, x) \iff g(k, c) \neq ? \iff \text{Prob}\{K = k, C = c\} \neq 0$$

Else $\chi(k, c)$ is zero. Since decoding is deterministic each of the $|\mathcal{K}|$ rows must have at least $|\mathcal{X}|$ ones; if there is no splitting the number of ones must be *exactly* $|\mathcal{X}|$ in each row.

Appendix 2: Some applications of bound (2)

When the attack is impersonation, $Z = K$, $\phi = \chi$, and (2) becomes literally the bound on impersonation given by Johannesson and this author in [8]; as $\chi^*(K, C) = 0$ with probability 1 (this means that the random couple K, C belongs to the minimization set defining the rate-distortion function), one has $R(Z, \phi^*) \leq I(K \wedge C)$, and one re-obtains the well-known Simmons bound $P_I \geq 2^{-I(K \wedge C)}$. When the attack is substitution of codeletter c the very crude inequality $R(Z, \phi^*) \leq H(Z) \equiv H(K|C = c)$ gives $P_S(c) \geq 2^{-H(K|C=c)}$, and so for the average probability of substitution $P_S \doteq \sum_c \text{Prob}\{C = c\} P_S(c)$ a trivial convexity argument gives $P_S \geq 2^{-H(K|C)}$. For a chosen-content attack with target set $\mathcal{M}$, (2) becomes literally the bound given in [2]; the combinatorial bound (4) gives $P_{\mathcal{M}} \geq \frac{|\mathcal{M}|}{|\mathcal{C}|}$ for deterministic codes, as in [2], [3] and [4]; for more bounds cf [9]. For the *deception* attack, which consists in choosing whichever attack of either impersonation or substitution has higher fraud probability, one has $P_D = \max[P_I, P_S] \geq \frac{1}{2} P_I + \frac{1}{2} P_S \geq 2^{-\frac{H(K)}{2}}$ (use again a trivial convexity argument and the bounds to P_I and P_S just given); since $H(K) \leq \log |K|$, one re-obtains also the "historical" combinatorial bound of Gilbert, Mac Williams and

Sloane [7], $P_D \geq \frac{1}{\sqrt{|K|}}$. However, our bound (2) makes more than just giving these old bounds; since the inequality $R(Z, \phi^*) \leq H(Z)$ can be quite weak as it ignores a term which is large essentially when the key equivocation of the code is large, the general bound (2) soon informs us that the "old" specific bounds to P_S and P_D are quite optimistic in the case of an authentication code which has to have also good cryptographic (secrecy) properties (we are being sloppy, but one can be much more precise by working on the ignored term $H(Z|Y)$, Y being now the random *fake* codeletter, which is constrained to give $\phi^*(Z, Y) = 0$).

References

[1] G.J. Simmons: *A Survey of Information Authentication*, Ch. 7 of *Contemporary Cryptology*, IEEE Press (1992) 379-419

[2] A. Sgarro: *Information-Theoretic Bounds for Authentication Frauds*, Journal of Computer Security, **2** (1993) 53-63; a preliminary version is to be found in: *Advances in Cryptology - Eurocrypt '92* ed. by R.A. Rueppel, Lectures Notes in Computer Science **658**, Springer (1993) 467-471

[3] A. Mereu, A. Sgarro: *Giochi di autenticazione*, Stato e prospettive della ricerca crittografica in Italia, Atti del Secondo Simposio, Fondazione Ugo Bordoni, Roma (1989) 256-262

[4] R. Safavi-Naini, L. Tombak: *Authentication Codes in Plaintext and Chosen-content Attacks*, Designs, Codes and Cryptography, **7-1/2** (1996) 83-99

[5] I. Csiszár, J. Körner: *Information Theory*, Academic Press (1981)

[6] Th. Cover, J. Thomas: *Elements of Information Theory*, Wiley (1991)

[7] E. N. Gilbert, F. J. MacWilliams, N. J. A. Sloane: *Codes which Detect Deception*, Bell System Technical Journal, **53-3** (1974) 405-424

[8] R. Johannesson, A. Sgarro: *Strengthening Simmons' Bound on Impersonation*, IEEE Transactions on Information Theory, **37-4** (1991) 1182-1185

[9] C. Crevato, A. Sgarro: *General Lower Bounds for Chosen-Content Attacks against Authentication Codes*, Journal of Discrete Mathematical Sciences & Cryptography, **1-1** (1998) 63-72

Cheating in split-knowledge RSA parameter generation

Marc Joye[1,*] Richard Pinch[2]

[1] LCIS, Tamkang University
Tamsui, Taipei Hsien, Taiwan 25137, R.O.C.
Email: joye@ug.ee.tku.edu.tw

[2] Queens' College
Silver Street, Cambridge CB3 9ET, U.K.
Email: rgep@chalcedon.demon.co.uk

Abstract

Cocks (1997) has recently described a protocol for two parties to generate an RSA modulus $N = PQ$ where neither party has knowledge of the factorisation, but which enables the parties to collaborate to decipher a encrypted message. We describe a number of ways in which it is possible for one of the parties to the protocol to cheat and obtain knowledge of the factorisation, and suggest modifications to the protocol to guard against cheating.

1 Introduction

At the last IMA Conference on Cryptography and Coding, Cocks [3] described a protocol for two parties to generate an RSA modulus $N = PQ$ where neither party has knowledge of the factorisation, but which enables the parties to collaborate to decipher a encrypted message. An alternative method is described by Boneh and Franklin [2].

His protocol allows two parties A and B to form an RSA modulus $N = PQ$ in such a way that neither party has knowledge of the factorisation, but the two parties can combine later to decipher a encrypted message. In the proposal of Boneh and Franklin, A and B need the services of a trusted

*Sponsored by the National Science Council, Republic of China, under grant NSC88-2811-E-032-0001.

helper H to carry out the distributed computation to form the modulus N: the proposal of Cocks requires no additional parties.

We show that there are a number of ways in which the parties to Cocks's protocol can cheat: that is, obtain knowledge of the factorisation of N or force their own choice of N. In each case the cheating party would later be able to decrypt messages without the collaboration of the other party.

2 The protocol of Cocks

We briefly describe the proposed protocol. There are three stages: generation of N; testing that N is a product of two primes; sharing the encryption and decryption exponents. The protocol of Boneh and Franklin differs in the first stage, where the trusted helper H is employed: the remaining two stages are broadly similar.

At the beginning of the first stage, A secretly chooses two random numbers p_1 and q_1 and B similarly chooses p_2 and q_2. Their intention is to form $N = PQ$ where $P = p_1 + p_2$ and $Q = q_1 + q_2$, and P and Q are prime. We assume that A and B have their own RSA moduli N_A and N_B with enciphering exponents e_A and e_B and deciphering exponents d_A and d_B.

In the first stage A transmits $p_1^{e_A}$ and $q_1^{e_A}$ to B. B now chooses three sets of k numbers $x_{1,i}$, $x_{2,i}$, $x_{3,i}$, such that for each j we have $\sum_i x_{j,i} = 1$. He then forms the set of $3k$ numbers

$$X = \left\{ x_{1,i}^{e_A} p_1^{e_A} q_2^{e_A}, x_{2,i}^{e_A} p_2^{e_A} q_1^{e_A}, x_{3,i}^{e_A} p_2^{e_A} q_2^{e_A} : i = 1, \ldots, k \right\}$$

and transmits the elements of X in some order to A. We call this the *splitting process* and say that B has split the values p_1q_2, p_2q_1 and p_2q_2. A now forms

$$N = p_1q_1 + \sum_{x \in X} x^{d_A} = p_1q_1 + p_1q_2 + p_2q_1 + p_2q_2 = PQ$$

and transmits the result to B. It is also proposed that A and B might wish to carry out this stage again with rôles reversed: we refer to this as the *symmetric case.*

In the second stage A and B agree a random value b. A forms $b^{N+1-p_1-q_1}$ mod N and B forms $b^{p_2+q_2}$ mod N. If these disgree then $\phi(N)$ cannot equal $N+1-P-Q = (P-1)(Q-1)$ and so P and Q cannot both be prime. A and B exchange hashes of these values and if they disagree then N is rejected and the protocol restarts at the first stage. If sufficiently many trial b pass this test, then P and Q are declared (probably) prime.

Boneh and Franklin suggest a modification of this test to avoid the possibility that P or Q might be a Carmichael number (which would pass the proposed test unless b actually has a factor in common with N). They propose that P and Q should be chosen to be $\equiv 3 \bmod 4$ by suitable conditions on the p_i and q_i. Then b is chosen so that the Jacobi symbol $\left(\frac{b}{N}\right) = +1$ and the necessary condition $b^{\phi(N)/4} \equiv \pm 1 \bmod N$ is tested by comparing $\pm b^{(N+1-p_1-q_1)/4}$ and $b^{(p_2+q_2)/4}$. Again if these do not agree then N is rejected, otherwise if N passes sufficiently many of these tests then it is allowed to proceed to the next stage. The advantage of this refinement is that if P and Q are not both prime then the probability of the test succeeding with a randomly chosen b is less than $1/2$ [2, Lemma 2].

In the third stage an enciphering exponent e is agreed on (perhaps fixed in advance). A and B exchange the values $p_i + q_i \bmod e$. Each can now compute $f = P + Q - N - 1 \bmod e$ and its inverse $g = f^{-1} \bmod e$. A forms

$$d_1 = \left\lfloor \frac{1+\left(\frac{N+1}{2} - p_1 - q_1\right)g}{e} \right\rfloor$$

and B forms

$$d_2 = \left\lceil \frac{\left(\frac{N+1}{2} - p_2 - q_2\right)g}{e} \right\rceil$$

Since

$$d = d_1 + d_2 = \frac{1+(N+1-P-Q)g}{e} ,$$

we have $de \equiv 1 \bmod \phi(N)$. Hence to decrypt $C = M^e \bmod N$, A can form C^{d_1} and B can form C^{d_2}, so that $M = C^{d_1} \cdot C^{d_2}$.

3 Preliminary remarks

From the discussion of the protocol, it is clear that either A or B can cheat if they can obtain the secrets p_i and q_i of the other party. With this information the cheater can mimic the computations of the other party in order to recover the deciphering exponent d, or predict the other's responses in order to impose a different value N' and have it accepted as the correct modulus.

In particular, if either party can obtain both the values of $p_i + q_i$ at any point, then from that point on they can privately simulate the other party's computations. This will enable the cheater to ensure that any given N (whether or not the result of carrying out that stage correctly) is passed as being of the appropriate form, and also allow the cheater to participate in joint deciphering, without the cheating being detected.

We further note that the protocol may be expected to be repeated a large number of times (Cocks [3] suggests 10^4 times for a modulus of 1024 bits). A cheater can therefore afford to wait for a reasonable amount of good luck, deliberately causing the protocol to repeat until it occurs.

4 Cheating in the first stage

Clearly A can cheat in the first stage by announcing any value N' she pleases as the result of the computation. However, in order to have any chance of passing the second stage, it seems that A needs to obtain p_2 and q_2 in order to predict B's contribution $b^{p_2+q_2}$. We shall see in the next section that this is not the case. However, a possible way to obtain p_2, q_2 is for A to privately carry out the computation correctly in order to obtain the true value of N. She may then attempt to factor this. If it is possible to factor N quickly, for example if it has only small prime factors and so is easy to factor by trial division, then A may be able to obtain the secrets p_2 and q_2 from the resulting factors.

It is also possible for B to cheat. Suppose that B applies the splitting process to the numbers $(-p_1q_1)^{e_A}$, α^{e_A} and β^{e_A}. We note that B can do this, using $(-p_1q_1)^{e_A} = -p_1^{e_A}q_1^{e_A}$, since e_A must be odd. A will then form $p_1q_1 - p_1q_1 + \alpha + \beta = \alpha + \beta$. So B can force an arbitrary modulus N' by taking α at random and $\beta = N' - \alpha$. However, B will have no more information about p_1 and q_1 than he would have had by applying the protocol correctly (that is, B will possess only $p_1^{e_A}$ and $q_1^{e_A}$).

Here we see that performing the first stage symmetrically actually weakens it by making it possible for B to cheat in the same way as A: since he is playing the rôle of A in the reversed computation, he can obtain the true value of N privately and has the chance of factoring it.

It is also possible for B to cheat in the asymmetric case. He chooses a prime r and applies the previous method with $\alpha = p_1r$ and $\beta = q_1r$. A will now form $N' = \alpha + \beta = (p_1 + q_1)r$. A returns N' to B, who knows r and hence obtains $p_1 + q_1$. Although B has not imposed a modulus of his own choice, he now has the factors of the modulus N' agreed on and enough information required to force it through the remaining stages.

5 Cheating in the second stage

Suppose now that one of the parties has imposed a false value N' on the other during the first stage. We have already observed that it is possible

to force this number to pass the second stage given information about the $p_i + q_i$.

We now show that it is possible to cheat without this knowledge: for example, if B has cheated in an asymmetric first stage. Suppose that N' has been imposed by the cheater after choosing it so that all its prime factors p satisfy $p - 1 \mid L$ for some L. For example, taking $L = 2^5 3^4 5 \cdot 7 = 90720$, there are 49 such primes, with a product in excess of 10^{270}. It is certainly possible to form a value of N' of say 512 bits. Then the exponent $\lambda(N')$ of the multiplicative group modulo N' will divide L and there is a chance of at least $1/L$ that the value $b^{N'+1-p_1-q_1}$ or $b^{p_2+q_2}$ formed during the computation will be $\equiv 1 \bmod N'$. So the cheater simply hopes that this will be the case, and tries again if not.

6 Preventing cheating

We note that certain forms of cheating will require the cheater to undertake a certain, possibly considerable, amount of extra computation compared to an honest participant. We suggest that, as a general principle, timing requirements should be included in any such protocol and that a session should be aborted if either party fails to reply in a timely manner.

A related requirement is that the protocol should specify a maximum number of repetitions consistent with the probability of finding a number of the required form. If the cheater is waiting for some other event which is less likely than the legitimate one, this should lead to the protocol being aborted before the cheater succeeds.

We further suggest that at the end of the first stage, either party should have the option of randomly requiring that the round terminate with each side immediately revealing their choice of random p_i, q_i. It would then be open to both parties to verify that the data which has passed is consistent with the revealed secret. A cheater would not be able to do this and so will be detected. As a further security measure we suggest that, as in all similar protocols for generation of secret keys, an audit trail should show the pseudo-random generation from an initial seed, and that this should be done through a cryptographically strong generator to prevent a 'seed' being determined from the output.

We note that a number of the methods of cheating that we have proposed are in fact easy to detect. For example, one method leads to a modulus N' divisible by $p_1 + q_1$: another leads to N' being divisible only by small primes. One might advocate building checks against these into the protocol. How-

ever, we note that they do little to guard against other, as yet undiscovered methods of cheating, and hence may be thought to lead an unwarranted degree of belief in the validity of a run which passes these tests alone.

7 Does cheating matter?

At first sight, it might appear that cheating is not an important issue for the intended application. The parties involved will presumably be acting as "trusted third parties" and it might be held that trusted parties do not cheat. We do not reject this out of hand: but it is perhaps worth noting that the parties involved are likely to be large corporations or government departments, and it is not so clear that every employee or sub-contractor will be equally trustworthy.

We might also note that for political acceptability of such arrangements, it is desirable to minimise the amount of reliance which the general public is required to place in the trustworthiness of these parties. Fielding a protocol which is known to be vulnerable to cheating is likely to diminish public confidence and encourage opposition from those already predisposed to resist the principles involved.

Acknowledgements

We gratefully acknowledge an anonymous referee for some useful remarks.

References

[1] Simon Blackburn, Simon Blake-Wilson, Mike Burmester, and Steven Galbraith, *Shared generation of shared RSA keys*, Technical Report CORR98-19, Department of Combinatorics & Optimization, University of Waterloo, 1998.

[2] Dan Boneh and Matthew Franklin, *Efficient generation of shared RSA keys*, Advances in Cryptology – CRYPTO '97 (Burton S. Kaliski Jr., ed.), Lecture Notes in Computer Science, vol. 1294, Springer-Verlag, 1997, pp. 425–439.

[3] Clifford Cocks, *Split knowledge generation of RSA parameters*, Cryptography and Coding (Michael Darnell, ed.), Lecture Notes in Computer Science, vol. 1355, Springer-Verlag, 1997, pp. 89–95.

[4] ———, *Split generation of RSA parameters with multiple participants*, Available from <http://www.cesg.gov.uk/>, February 1998.

A Modified protocol

Cocks later modified his protocol and extended it to multiple participants [4]. In this appendix, we will show that it is also possible to cheat in this modified scheme.

We only describe the protocol for two parties, called "basic algorithm" in [4]. As before each party secretly chooses p_i and q_i; their intention is to obtain a shared RSA modulus $N = (p_1 + p_2)(q_1 + q_2)$. Using the previous notations, the (basic) protocol goes as follows. A sends $p_1^{e_A}$ and $q_1^{e_A}$ to B, and B similarly sends $p_2^{e_B}$ and $q_2^{e_B}$. A then splits the values p_1q_2, q_1p_2 and a_{12} (where a_{12} is a positive number secretly chosen) resulting in the set X_A. Likewise, B forms the set X_B by splitting p_2q_1, q_2p_1 and a_{21}. A and B exchange the elements of X_A and X_B. Next, A computes $N_1 = 2p_1q_1 - a_{12} + \sum_{x \in X_B} x^{d_A}$; similarly B computes $N_2 = 2p_2q_2 - a_{21} + \sum_{x \in X_A} x^{d_B}$. They finally exchange N_1 and N_2 to form $N = (N_1 + N_2)/2 = (p_1 + p_2)(q_1 + q_2)$.

Suppose that A cheats. At the end of the protocol, the RSA modulus will be $N' = PQ$.

A first chooses a number α and a prime P. She takes $q_1 = 2P$. Then, instead of sending X_A and N_1 to B, she sends X'_A and N'_1 given by

$$X'_A = \{\text{splitting of } -2p_2q_2,\ -p_1q_2 \text{ and } 2Pq_2\}$$

and

$$N'_1 = 2\alpha P + \sum_{x \in X_B} x^{d_A} \ .$$

From X'_A, B will then compute

$$N_2 = 2p_2q_2 - a_{21} + \sum_{x \in X'_A} x^{d_B} = -a_{21} - p_1q_2 + 2Pq_2.$$

Noting that A has chosen $q_1 = 2P$, the resulting RSA modulus will be thus equal to

$$N' = \frac{N'_1 + N_2}{2} = \frac{(2\alpha P + 2p_2P + q_2p_1 + a_{21}) + (-a_{21} - p_1q_2 + 2Pq_2)}{2} = P(\alpha + p_2 + q_2) \ .$$

Notice that from N', A can obtain $p_2 + q_2$ and go through the remaining stages. Notice also that sending in advance a hash value (as suggested in [4, §5.1]) does not prevent cheating.

Fair and Efficient Proof that Two Secrets are (not) Equal – How to Solve the Socialist Millionaires' Problem

Fabrice Boudot and Jacques Traoré

France Télécom
Centre National d'Etudes des Télécommunications - Site de Caen
42 rue des Coutures BP 6243 14066 Caen Cedex 4, France
fabrice.boudot@cnet.francetelecom.fr
jacques.traore@cnet.francetelecom.fr

Abstract. We present in this paper a solution to the *Tiercé*[1] *problem*, in which two backers, Alice and Bob, want to know if they have backed the same combination or a different one (but neither Alice nor Bob wants to disclose her (his) combination). This problem is also known as the *socialist millionaires' problem* [JY], in which two socialist millionaires want to know if they own exactly the same amount of wealth. In our solution, both backers will be convinced of the equality (or inequality) of their combinations and will get no extra information about the other backer's combination. Moreover, its complexity is polynomial in the size of the secrets (the combinations). Finally, this protocol is *fair*: one party cannot get the result of the comparison while preventing the other one from getting it.

1 Introduction

1.1 Description of the problem

Alice and Bob are two backers and both have decided to back a combination for the next Tiercé. Let x be Alice's combination and y be Bob's one. Alice and Bob, together, want to know whether they have the same combination or not, but neither Alice nor Bob wants to disclose their respective combinations. This problem called *Tiercé problem* [Zém] is a variant of the *millionaires' problem*, where two millionaires wish to know who of two of them is the wealthier, but without disclosing their respective wealth. The latter problem has been introduced by Yao [Ya1,Ya2].
There is a naive method to solve the *Tiercé problem*: Alice sends $h(x)$ to Bob and Bob sends $h(y)$ to Alice, where h is a one-way hash-fonction. If $h(x) = h(y)$, then $x = y$ with overwhelming probability. But this solution cannot be applied when the

[1] Tiercé is a French betting game, where the backers are expected to guess the winning combination which consists of the three first of a horse race.

secrets are small (because it is easy to compute $h(\tilde{x})$ for all possible values of $\tilde{x}$ and determine the one which is equal to $h(x)$, the value sent by Alice). Moreover, this solution is not fair: Bob can send nothing, or send back $h(x)$ to Alice even when $y \neq x$, or else send back a value different from $h(y)$ even when $x = y$. Furthermore, even if x is large, Bob learns some information about x (i.e. $h(x)$) and is able to test whether a given value is Alice's secret or not.
A solution to the millionaires' problem is described by Salomaa in [Sal] (and also by Schneier [Schn]). From this solution, we can easily deduce a solution to the *Tiercé problem*. Unfortunately, such a solution is only efficient when x and y are very small (the complexity is polynomial in x and y). This solution cannot be used, for example, for integers x and y of at least 40 bits.
Recently, at Crypto'96, Jakobsson and Yung [JY] presented a solution to the *Tiercé problem* (or *socialist millionaires' problem*, using their terminology). Unfortunately, this solution uses a cut-and-choose method and so is relatively inefficient.
The protocol we present in this paper is more efficient than Jakobsson and Yung's one. Besides, its complexity is polynomial in the size of the secrets and so allows one to compare integers of all sizes. Moreover, it allows a *fair revelation* of the result of the comparison of x and y, to prevent a party from stopping the protocol after getting the result of the comparison, and consequently preventing the other party from getting the same result[2].

1.2 Organization of the paper

In section 2, we define the notations we use in this paper, then we introduce the assumptions on which the security of our protocol relies. Finally, we present the *non-interactive proofs of knowledge* we use in the new protocol. In section 3, we describe this new protocol which solves the *Tiercé problem*. Then, we give heuristic arguments for the security of our protocol. Finally, we conclude in section 4.

2 Notations, Assumptions and Proofs of knowledge

2.1 Notations

We will denote Z_n the residue class ring modulo n and Z_n^* the multiplicative group of invertible elements in Z_n. If h is a hash-fonction, we will denote $h(a,b)$ the image by h of the concatenation of the strings a and b. The symbol $\oplus$ will denote the bitwise operator " exclusive-or ".

---

[2] The protocol presented by Jakobsson and Yung [JY] is not fair (like the one described by Salomaa). So we have compared the efficiency of their protocol to the *non fair* version of ours.

Finally, we will denote $\log_g(a)$ the discrete logarithm of a in base g (when it exists), i.e. the smallest integer x such that $g^x \equiv a \bmod p$ (where p is prime).

2.2 Assumptions

The security of our protocol relies on three standard assumptions in cryptography:
Assumption (A1) (Discrete logarithm): Let p and q be prime numbers such that $q \mid p-1$, and let g be an element of order q in Z_p^*. Let x and y be two integers such that $y = g^x \bmod p$. Then, it is hard to compute x given only y, p, q and g. We will denote $\log_g(y)$ the value of the discrete logarithm of y in base g.
Assumption (A2) (Decision Diffie-Hellman): Let p and q be prime numbers such that $q \mid p-1$, and let g be an element of order q in Z_p^*. Given $g^x \bmod p$, $g^y \bmod p$ and $g^z \bmod p$, it is hard to *decide* whether $z = xy \bmod q$ or not.
Assumption (A3) (Unicity of a decomposition in a base): Let p and q be prime numbers such that $q \mid p-1$, and let $g_1,\ldots,g_n$ be n elements of order q in Z_p^*. If the values $\log_{g_i}(g_j)$, $i \neq j$ are not known, then it is hard to find $(x_1,\ldots x_n) \neq (x'_1,\ldots x'_n)$ such that $g_1^{x_1}\cdots g_n^{x_n} = g_1^{x'_1}\cdots g_n^{x'_n} \pmod p$. We will denote by $(x_1,\ldots x_n)$ the coordinates (or the representation) of $G = g_1^{x_1}\cdots g_n^{x_n} \bmod p$ in base $(g_1,\ldots,g_n)$.
Note: In fact, the assumptions (A1) and (A3) are equivalent, and (A2) is stronger than (A1) and (A3). As a consequence, the security of our protocol only relies on assumption (A2).

2.3 Non-interactive proofs of knowledge

The following protocols are non-interactive *zero-knowledge* proofs of knowledge, *derived* from well-known interactive *zero-knowledge* proofs of knowledge. The generic transformation (from an interactive protocol into a non-interactive one) we use has been introduced by Fiat and Shamir [FS], and preserves the properties of the original protocol: Bob is convinced by Alice's proof, if and only if she holds the secret whose knowledge she wants to prove, and at the end of this protocol Bob will learn no information about this secret.
Note: We assume, in order to simplify the description of the proofs of knowledge we will introduce, that the verifier (in this case Bob) already knows all the public parameters related to the assertion the prover (in this case Alice) wants to prove.

2.3.1 Proof of knowledge of a discrete logarithm

Let p and q be prime numbers such that $q \mid p-1$. Let g be an element of order q in Z_p^*, and h be a hash-fonction.

The protocol described below allows Alice to prove to Bob that she knows an element x in Z_q^* which satisfies: $y = g^x \bmod p$ [3].

Alice randomly selects an integer r in Z_q^*, computes $W = g^r \bmod p$, $c = h(W)$ and $D = r - xc \bmod q$. Then Alice sends *the proof* (c, D) to Bob.

Bob is convinced (accepts the *proof*) if: $c = h(g^D y^c \bmod p)$.

This protocol is due to Schnorr [Sch].

2.3.2 Proof of knowledge of discrete coordinates

Let p and q be prime numbers such that $q | p-1$. Let g_1 and g_2 be two elements of order q in Z_p^* such that $\log_{g_2}(g_1)$ is unknown to both Alice and Bob, and h be a hash-fonction.

Using the protocol described below, Alice is able to prove to Bob that she knows a couple (x_1, x_2) of elements in Z_q^* verifying: $y = g_1^{x_1} g_2^{x_2} \bmod p$.

Alice randomly selects two integers r_1 and r_2 in Z_q^*, computes $W = g_1^{r_1} g_2^{r_2} \bmod p$, $c = h(W)$, $D_1 = r_1 - x_1 c \bmod q$ and $D_2 = r_2 - x_2 c \bmod q$. Then Alice sends the proof (c, D_1, D_2) to Bob.

Bob is convinced if: $c = h(g_1^{D_1} g_2^{D_2} y^c \bmod p)$.

This protocol is due to Okamoto [Oka].

2.3.3 Proof of equality of two discrete logarithms

Let p and q be prime numbers such that $q | p-1$. Let g_1 and g_2 be two elements of order q in Z_p^* such that $\log_{g_2}(g_1)$ is unknown to both Alice and Bob, and h be a hash-fonction.

The protocol described below allows Alice to prove to Bob that she knows an element x in Z_q^* verifying: $y_1 = g_1^x \bmod p$ and $y_2 = g_2^x \bmod p$.

Alice randomly selects an integer r in Z_q^*, computes $W_1 = g_1^r \bmod p$, $W_2 = g_2^r \bmod p$, $c = h(W_1, W_2)$ and $D = r - xc \bmod q$. Then Alice sends the proof (c, D) to Bob.

Bob is convinced if: $c = h(g_1^D y_1^c \bmod p, g_2^D y_2^c \bmod p)$.

This protocol is due to Chaum and Pedersen [CP].

2.3.4 Proof of equality of two discrete coordinates

Let p and q be prime numbers such that $q | p-1$. Let $g_1, \ldots, g_m$ and $g'_1 \ldots, g'_n$ be $m+n$ elements of order q in Z_p^* such that $\log_{g_i}(g_j)$, $i \neq j$, and $\log_{g'_i}(g'_j)$, $i \neq j$, are unknown to both Alice and Bob, and h be a hash-fonction. Let $(x_1, \ldots x_m)$ be m

---

[3] As previously specified, we will implicitly assume that Bob knows : q, p, g and y.

elements such that $P = g_1^{x_1} \cdots g_m^{x_m} \bmod p$ and $(x'_1,\ldots x'_n)$ such that $Q = g_1'^{x'_1} \cdots g_n'^{x'_n} \bmod p$.

The protocol described below allows Alice to prove to Bob that she knows $(x_1,\ldots x_m)$ and $(x'_1,\ldots x'_n)$ and that $x_1 = x'_1$.

Alice randomly selects $m+n$ integers $r_1,\ldots,r_m,r'_2,\ldots r'_n$ in Z_q^*, computes $W = g_1^{r_1} \cdots g_m^{r_m} \bmod p$ and $W' = g_1'^{r_1} g_2'^{r'_2} \cdots g_n'^{r'_n} \bmod p$, then $c = h(W,W')$, and finally $D_1 = r_1 - x_1 c \bmod q$, ... , $D_m = r_m - x_m c \bmod q$, $D'_2 = r'_2 - x'_2 c \bmod q$, ... , $D'_n = r'_n - x'_n c \bmod q$.

Alice sends the proof $(c, D_1, D_2, \ldots D_m, D'_2, \ldots, D'_n)$ to Bob.

Bob is convinced if $c = h(g_1^{D_1} g_2^{D_2} \cdots g_m^{D_m} P^c \bmod p,\ g_1'^{D_1} g_2'^{D'_2} \cdots g_n'^{D'_n} Q^c \bmod p)$.

2.3.5 Proof that a coordinate is equal to 0 or to 1

Let p and q be prime numbers such that $q | p-1$. Let g_1, ..., g_m be m elements of order q in Z_p^* such that $\log_{g_i}(g_j)$, $i \neq j$, and $\log_{g'_i}(g'_j)$, $i \neq j$, are unknown to both Alice and Bob, and h be a hash-function. Let $(x_1,\ldots x_m)$ be m integers such that $P = g_1^{x_1} \cdots g_m^{x_m} \bmod p$.

The following protocol allows Alice to prove to Bob that she knows $(x_1,\ldots x_m)$ and that $(x_2,\ldots x_m) \in \{0,1\}^{m-1}$. For this purpose, she proves that for all $i = 2,\ldots,m$ she knows either the representation of P (in case $x_i = 0$), or the representation of P / g_i (in case $x_i = 1$) in base $(g_1,\ldots,g_{i-1},g_{i+1},\ldots g_m)$. The idea to prove the knowledge of one out of two secrets is from [CDS].

For i from 2 to m **do**:

If $x_i = 0$:

Alice randomly selects $m-1$ elements $r_1,\ldots,r_{i-1},r_{i+1},\ldots,r_m$ in Z_q^* and computes $W = g_1^{r_1} \cdots g_{i-1}^{r_{i-1}} g_{i+1}^{r_{i+1}} \cdots g_m^{r_m} \bmod p$. Moreover, she randomly selects an element $c' \in \mathrm{Im}(h)$ and $m-1$ elements $D'_1,\ldots,D'_{i-1},D'_{i+1},\ldots,D'_m$ in Z_q^* and computes $W' = g_1^{D'_1} \cdots g_{i-1}^{D'_{i-1}} g_{i+1}^{D'_{i+1}} \cdots g_m^{D'_m} (P / g_i)^{c'} \bmod p$.

Alice computes $C = h(W,W')$, then $c = C \oplus c'$.

Finally, she computes $D_1 = r_1 - x_1 c \bmod q$, ... , $D_{i-1} = r_{i-1} - x_{i-1} c \bmod q$, $D_{i+1} = r_{i+1} - x_{i+1} c \bmod q$, ..., $D_m = r_m - x_m c \bmod q$.

If $x_i = 1$:

Alice randomly selects $m-1$ elements $r'_1,\ldots,r'_{i-1},r'_{i+1},\ldots,r'_m$ in Z_q^* and computes $W = g_1^{r'_1} \cdots g_{i-1}^{r'_{i-1}} g_{i+1}^{r'_{i+1}} \cdots g_m^{r'_m} \bmod p$. Moreover, she randomly selects an element $c \in \mathrm{Im}(h)$ and $m-1$ elements $D_1,\ldots,D_{i-1},D_{i+1},\ldots,D_m$ in Z_q^* and computes $W' = g_1^{D_1} \cdots g_{i-1}^{D_{i-1}} g_{i+1}^{D_{i+1}} \cdots g_m^{D_m} P^c \bmod p$.

Alice computes $C = h(W,W')$, then $c' = C \oplus c$.
Finally, she computes $D'_1 = r'_1 - x_1 c' \mod q$, ..., $D'_{i-1} = r'_{i-1} - x_{i-1} c' \mod q$, $D'_{i+1} = r'_{i+1} - x_{i+1} c' \mod q$, ..., $D'_m = r'_m - x_m c' \mod q$.
Alice sends the proof $(c, c', D_1, \ldots, D_{i-1}, D_{i+1}, \ldots D_m, D'_1, \ldots, D'_{i-1}, D'_{i+1}, \ldots, D'_m)$ to Bob.
Bob checks that:
$c \oplus c' = h(g_1^{D_1} \cdots g_{i-1}^{D_{i-1}} g_{i+1}^{D_{i+1}} \cdots g_m^{D_m} P^c \mod p,\ g_1^{D'_1} \cdots g_{i-1}^{D'_{i-1}} g_{i+1}^{D'_{i+1}} \cdots g_m^{D'_m} (P / g_i)^{c'} \mod p)$.
end (for)
Bob is convinced if the final check succeeds.

3 Presentation of a new protocol

The following protocol allows Alice and Bob to prove to each other that their respective secrets x and y are equal (or not) in such a way that Bob learns nothing about x and Alice learns nothing about y (except, of course, that their secrets are equal or different). This protocol is well suited to the case when x and y are small.

3.1 Parameter generation

Alice (or Bob) chooses a large prime number q (at least 160 bits). Then, Alice (or Bob) chooses a large prime p (at least 512 bits) such that $q | p-1$, and g_1 an element of Z_p^* of order q. The generation of such numbers does not pose problems.
We assume, to simplify the description of this protocol, that x and y are two elements in Z_q^* (if x or y is greater than q, then we use this protocol to compare $h(x)$ and $h(y)$, where h is a hash-fonction that maps $\{0,1\}^*$ to $\{0,1\}^{160}$).

3.2 Generation of elements of order q

Let k be a security parameter, such that it is computationally infeasible to do 2^k computations in a human-scale time and with human-scale computation resources (nowadays, k is taken equal to 80).
Alice and Bob must generate $k+2$ elements, g_2, g_3, f_1, f_2, ..., f_k, of the multiplicative group generated by g_1 such that neither of them knows $\log_{g_1}(f_i)$ for $i \in \{1,..,k\}$, $\log_{g_2}(f_i)$ for $i \in \{1,..,k\}$, $\log_{f_i}(f_j)$ for $i \neq j$, $\log_{g_1}(g_2)$ and $\log_{g_1}(g_3)$.
Alice and Bob use the following protocol to determine g_3, an element of the multiplicative group generated by g_1:

Alice randomly selects an element x_a in Z_q^*, computes $h_A = g_1^{x_a} \bmod p$ and proves to Bob that she knows $\log_{g_1}(h_A)$ by using the protocol 2.3.1. Bob randomly selects an element x_b in Z_q^*, computes $h_B = g_1^{x_b} \bmod p$ and proves to Alice that he knows $\log_{g_1}(h_B)$ by using the protocol 2.3.1. Then, Alice and Bob respectively compute $g_3 = g_1^{x_a \cdot x_b} = h_A^{x_b} = h_B^{x_a} (\bmod p)$. They also check that $h_A \neq 1 \bmod p$ and that $h_B \neq 1 \bmod p$.

Alice stores x_a and h_B and Bob stores x_b and h_A.

Alice and Bob also generate the elements g_2, f_1, ..., f_k, for example in the same way that they generated g_3, but without storing any intermediate value.

3.3 Development of the protocol

Step 1:

Alice randomly selects an element a in Z_q^*, and k bits $c_1,\ldots,c_k$, then computes $P_1 = g_1^a g_2^x \bmod p$[4] and $P_2 = g_3^a f_1^{c_1} f_2^{c_2} \cdots f_k^{c_k} \bmod p$ and sends (P_1, P_2) to Bob. Then she proves that she knows the representation of P_2 with respect to $(g_3, f_1, \ldots, f_k)$ and that the k least coordinates belong to $\{0,1\}$ by using the protocol 2.3.5. Then, by using the protocol 2.3.4, she proves that the first coordinate of P_1 with respect to (g_1, g_2) is equal to the first coordinate of P_2 with respect to $(g_3, f_1, \ldots, f_k)$.

Bob randomly selects an element b in Z_q^*, and k bits $e_1,\ldots,e_k$, then computes $Q_1 = g_1^b g_2^y \bmod p$[5] and $Q_2 = g_3^b f_1^{e_1} f_2^{e_2} \cdots f_k^{e_k} \bmod p$ and sends (Q_1, Q_2) to Alice. Then, he proves that he knows the representation of Q_2 with respect to $(g_3, f_1, \ldots, f_k)$ and that the k least coordinates belong to $\{0,1\}$ by using the protocol 2.3.5. Then, by using the protocol 2.3.4, he proves that the first coordinate of Q_1 with respect to (g_1, g_2) is equal to the first coordinate of Q_2 with respect to $(g_3, f_1, \ldots, f_k)$.

We denote $P_3 = g_3^a \bmod p$, $Q_3 = g_3^b \bmod p$ and $R = P_1 / Q_1 \bmod p$. Then,

$$x = y \bmod q \Leftrightarrow R^{x_a \cdot x_b} = P_3 / Q_3 \bmod p \qquad \text{(Test 1)}$$

In the second step of this protocol Alice and Bob both compute $R^{x_a \cdot x_b}$.

---

[4] Alice commits to her secret x.

[5] Bob commits to his secret y.

Step 2: Computation of $R^{x_a \cdot x_b}$

Alice computes $T = R^{x_a} \bmod p$ and sends T to Bob. Then, she proves to Bob that $\log_R T = \log_{g_1} h_A$ by using the protocol 2.3.3. This proof convinces Bob that T is actually equal to R^{x_a}.

Bob computes $U = R^{x_b} \bmod p$ and sends U to Alice. Then, he proves to Alice that $\log_R U = \log_{g_1} h_B$ by using the protocol 2.3.3. This proof convinces Alice that U is actually equal to R^{x_b}.

Alice and Bob are now able to compute $R^{x_a x_b} = T^{x_b} = U^{x_a} \pmod p$

In the third step of this protocol, Alice reveals P_3 to Bob (and proves to him by using the protocol 2.3.4 that she knows a couple (a, x) of elements of Z_q^* such that $P_1 = g_1^a g_2^x \bmod p$ and $P_3 = g_3^a \bmod p$). Bob reveals Q_3 to Alice (and proves to her by using the protocol 2.3.4 that he knows a couple (b, y) of elements of Z_q^* such that $Q_1 = g_1^b g_2^y \bmod p$ and $Q_3 = g_3^b \bmod p$). But if Alice immediately reveals P_3 to Bob, he will be able to check the equality (Test 1). Bob can then refuse to send Q_3 to Alice and deny her to check the equality. To alleviate this drawback, Alice and Bob must fairly transmit the values P_3 and Q_3 from the values P_2 and Q_2. For this purpose, Alice and Bob will gradually reveal the bits c_i and e_i.

Step 3: Fair exchange of the values P_3 and Q_3

For i from 1 to $k-1$ **do**:

Alice reveals the bit c_i to Bob and proves to him that she knows the representation of $P_2 / f_i^{c_i}$ with respect to $(g_3, f_{i+1}, \ldots, f_k)$ using (a generalization of) protocol 2.3.2.

Bob reveals the bit e_i to Alice and proves to her that she knows the representation of $Q_2 / f_i^{e_i}$ with respect to $(g_3, f_{i+1}, \ldots, f_k)$ using (a generalization of) protocol 2.3.2.

Alice and Bob set $P_2 = P_2 / f_i^{c_i} \pmod p$ and $Q_2 = Q_2 / f_i^{e_i} \pmod p$.

end (for).

Alice reveals the last bit c_k to Bob and proves to him that she knows the discrete logarithm of $P_2 / f_k^{c_k}$ in base g_3 by using the protocol 2.3.1.

Bob reveals the last bit e_k to Alice and proves to him that he knows the discrete logarithm of $Q_2 / f_k^{e_k}$ in base g_3 by using the protocol 2.3.1.

Alice and Bob set $P_2 = P_2 / f_k^{c_k} \pmod p$ and $Q_2 = Q_2 / f_k^{e_k} \pmod p$.

Remark: In order to check the equality (Test 1), Bob needs to know $P_3 = g_3^a \bmod p$, but this value is *hidden*[6] (by $f_1^{c_1} \cdot f_2^{c_2} \cdots f_k^{c_k}$). Alice needs to know $Q_3 = g_3^b \bmod p$, which is also hidden (by $f_1^{e_1} \cdot f_2^{e_2} \cdots f_k^{e_k}$). The protocol described above allows Alice (respectively Bob) to reveal to Bob (respectively Alice), bit after bit, the string

[6] It is computationally infeasible to find this value by exhaustive research (if k is well chosen).

$c = c_1 c_2 ... c_k$ (respectively $e = e_1 e_2 ... e_k$), and so to gradually reveal $g_3^a \bmod p$ (respectively $g_3^b \bmod p$).

Alice and Bob have respectively got (at the end of this step) Q_3 (= Q_2) and P_3 (= P_2). So they can check the final test (Test 1): $x = y \bmod q \Leftrightarrow R^{x_a x_b} = P_3 / Q_3 \bmod p$.

The assumption (A3) on the unicity of the representation ensures that each revealed bit is correct.

This step can be regarded as fair, because if Bob (for example) deliberately stops the protocol, he will only have one bit ahead of Alice to find, by exhaustive research, the missing bits c_i needed to compute P_3 (the same holds for Alice).

3.4 Security

In this section, we give heuristic arguments for the security of our protocol.

Correctness:

At the end of the protocol, Alice and Bob are convinced of the validity of the result of the comparison of their two secrets. Indeed, all the (non-interactive) *proofs* (of knowledge) given by Alice and Bob allow both of them to be mutually convinced of the validity of P_3, Q_3, U and T, and consequently of the validity of the final test (Test 1).

Secrecy:

If $x = y$, Bob knows Alice's secret, since it is equal to his secret.

If not, let us show, for example, that Bob learns no information about Alice's secret.

As the (non-interactive) *proofs* (of knowledge) used during this protocol are zero-knowledge, Bob has learnt no information about x_a, a and x from these proofs. Nevertheless, Alice has revealed the following values: $h_A = g_1^{x_A} \bmod p$, $P_1 = g_1^a g_2^x \bmod p$, $P_2 = g_3^a \bmod p$ and $T = g_1^{x_A(a-b)} g_2^{x_A(x-y)} \bmod p$. It is clear that the knowledge of these values is equivalent to the knowledge of: $(g_1^{x_A} \bmod p, g_1^a g_2^x \bmod p, g_1^{ax_A} \bmod p, g_2^{x_A(x-y)} \bmod p)$ because $g_1^{ax_A} = P_2^{x_B^{-1}} \bmod p$ and $g_2^{x_A(x-y)} = T / (P_2^{x_B^{-1}} \times h_A^{-b}) \bmod p$.

Bob may have some presumptions about x and want to know if one of these presumed values $\tilde{x}$ is actually Alice's secret. Let $\tilde{x}$ be the value Bob chooses for x. Let $\tilde{a}$ be the integer such that $g_1^{\tilde{a}} g_2^{\tilde{x}} = g_1^a g_2^x \bmod p$ and $\tilde{x}_A$ be the number such that $g_2^{\tilde{x}_A(\tilde{x}-y)} = g_2^{x_A(x-y)} \bmod p$. Bob can compute $g_1^{\tilde{a}} \bmod p$ and $g_2^{\tilde{x}_A} \bmod p$. To check that the value of $\tilde{x}$ (and so of $\tilde{a}$ and of $\tilde{x}_A$) is correct, he must be able to *decide* whether $\log_{g_1}(g_1^{x_A}) = \log_{g_2}(g_2^{\tilde{x}_A})$ or not, or whether $(g_1^{x_A}, g_1^{\tilde{a}}, g_1^{ax_a})$ is a valid Diffie-Hellman triplet or not. But Bob cannot decide (otherwise, this would contradict assumption (A2)).

Hence, under the decision Diffie-Hellman assumption (A2), this protocol does not reveal the secret x (resp. y) to Bob (resp. to Alice).

4 Conclusion - Open problem

We have presented in this paper a protocol which allows one to *fairly* check the equality (or the inequality) of two secrets, and this gives an answer to the *Tiercé problem*. The complexity of this protocol is polynomial in the size of the secrets. However, designing a fair and efficient solution for the *millionaires' problem* (*not socialist*) remains an open problem.

Acknowledgements

The authors would like to thank Gilles Zémor for stating the *Tiercé problem*, and Marc Girault for helpful comments.

References

[CDS] R. Cramer, I. Damgård and B. Schoenmakers, Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols, *Proceedings of CRYPTO'94, LNCS 839, pp. 174-187.*

[CP] D. Chaum and T.P. Pedersen, Wallet Databases with Observers, *Proceedings of CRYPTO'92, LNCS 740, pp. 89-105.*

[FS] A. Fiat and A Shamir, How to Prove Yourself: Practical Solutions to Identification and Signature Problems, *Proceedings of CRYPTO'86, LNCS 263, pp. 186-194.*

[JY] M. Jakobsson and M. Yung, Proving Without Knowing: On Oblivious, Agnostic and Blindfolded Provers, *Proceedings of CRYPTO'96, LNCS 1109, pp.186-200.*

[Oka] T. Okamoto, Provably Secure and Practical Identification Schemes and Corresponding Signatures Schemes, *Proceedings of CRYPTO'92, LNCS 740, pp. 31-53.*

[Sal] A. Salomaa, Public-Key Cryptography, *Springer-Verlag, 1990.*

[Sch] C.P. Schnorr, Efficient Signature Generation by Smart Cards, *Journal of Cryptology, vol. 4, pp.161-174.*

[Schn] B. Schneier, Applied Cryptography, *p. 551.*

[Ya1] A. Yao, Protocols for Secure Computations, *Proceedings of the 23th FOCS, 1982, pp.160-164.*

[Ya2] A. Yao, How to Generate and Exchange Secrets, *Proceedings of the 27th FOCS, 1986, pp.162-167.*

[Zém] G. Zémor, *Private Communication.*

Finite Fields: Primitive Normal Bases with Prescribed Trace

Dirk Hachenberger
Institut für Mathematik der Universität Augsburg,
86159 Augsburg, Germany
hachenberger@math.uni-augsburg.de

It is well-known that the multiplicative group E^* of a finite field E is cyclic, i.e., free on one generator as a module over the ring of integers; the generators of E^* are called *primitive elements of* E. A further classical result is the *normal basis theorem*: if G is the Galois group of a finite Galois extension E over a field F, then the additive group of E is free on one generator as a module over the group algebra FG. An element $w \in E$ for which $w^G = \{g(w)|g \in G\}$ is an F-basis of E is therefore called *free in* E *over* F or *normal in* E *over* F; w^G is called a *normal basis of* E *over* F. If in particular F is a finite field, then each finite extension E over F is a Galois extension. A combination of primitivity and normality was considered by Lenstra and Schoof [LeSc] in 1987: they proved the *primitive normal basis theorem*, which states that for each extension E/F of finite fields there exists a primitive element for E which likewise is normal over F.

The study and determination of primitive and normal elements are fundamental problems for the theory of finite fields. Those elements allow representations of a finite field, which have proved to be very useful when calculations in the field are required. Concrete applications are the decoding for error-correcting codes, the encryption and key-exchange in public-key-cryptosystems, or the generation of pseudorandom numbers.

In the talk we present recent results on the existence of primitive normal bases having additional properties. We consider triples (q, k, e), where $q > 1$ is a prime power and $k, e \geq 1$ are integers. To each such triple there corresponds a triple (F, K, E) of finite fields: $F = \mathrm{GF}(q)$ is the Galois field with q elements, and, in a fixed algebraic closure $\bar{F}$ of F, K and E are the unique extensions over F having degrees $[K : F] = k$ and $[E : F] = ke$, respectively (i.e., K is a subfield of E).

Notation. *Let* $\mathcal{T}$ *denote the set of triples* (q, k, e), *which satisfy the following condition: for* **every** $a \in K = GF(q^k)$ *which (necessarily) is normal*

over $F = GF(q)$ there exists an element $w \in E = GF(q^{ke})$, which (1) is primitive in E, (2) normal in E over F and (3) whose (E,K)-trace is equal to a (recall that (E,K)-trace of $w \in E$ is defined as $Tr_{E,F}(w) := \sum_{h \in H} h(w)$, where H is the Galois group of E/K).

The **Main Problem** is to decide which triples (q,k,e) belong to $\mathcal{T}$.

First, in a joined paper with Stephen D. Cohen [CoHa], we have proved the following extension of the primitive normal basis theorem of Lenstra and Schoof, and hence settled a conjecture of Morgan and Mullen [MoMu].

Theorem 1. ([CoHa]) *$(q,1,n) \in \mathcal{T}$ for all prime powers $q > 1$ and all integers $n \geq 1$, i.e., for any extension E/F of finite fields and any nonzero $a \in F$ there exists a primitive element $w \in E$ which is normal over F and whose (E,F)-trace is equal to a.*

The general problem, incorporating an arbitrary intermediate field, was first considered in [Ha]. The motivation for studying this generalization was to determine whether there exist trace-compatible sequences of primitive normal elements for certain towers of extensions over finite fields.

Definition. A *tower over F* is a set $\mathcal{L}$ of finite extensions over F which is totally ordered by inclusion. A sequence $w = (w_L)_{L \in \mathcal{L}}$ is called *trace-compatible for $\mathcal{L}$* if $\mathrm{Tr}_{E,K}(w_E) = w_K$ for all $E, K \in \mathcal{L}$ such that K is a subfield of E. Furthermore, w is called *normal over F* if w_L is normal in L over F for all $L \in \mathcal{L}$, and w is called *primitive* if w_L is primitive in L over F for all $L \in \mathcal{L}$.

As pointed out by Scheerhorn [Sche] (see also Section 2.1.6 in [CiD]), computationally simple embeddings of finite field extensions, which rely on a normal-basis representation consist of trace-compatible normal sequences. This additive presentation is used for instance in the computer algebra system Axiom. The following results of [Ha] are interesting: they state that under certain conditions, there exist trace-compatible normal sequences which are likewise primitive.

First, we have the following asymptotic result.

Theorem 2. ([Ha]) *Consider a tower $\mathcal{L}$ of finite fields. Assume that $[E:K] \geq e_0 \geq 3$ for all $K, E \in \mathcal{L}$ such that K is a subfield of E. Then there exists a constant q_0 such that the following is true: if $q \geq q_0$, where q is the cardinality of the smallest field $F \in \mathcal{L}$, then there exists a trace-compatible sequence $(w_L)_{L \in \mathcal{L}}$ which is primitive and normal over F.*

For example, if $e_0 = 4$, one can choose $q_0 = 3104$.

Concerning extensions of prime power degree, we have proved the following assertion, which holds without restrictions on the cardinality q of the ground field.

Theorem 3. ([Ha]) *Let p be the characteristic of $GF(q)$. Then $(q, r^a, r^b) \in \mathcal{T}$ for all $a \geq 0$ and all $b \geq 1$ if $r \geq 5$ or $r = p$; $(q, 3^a, 3^b) \in \mathcal{T}$ for all $a \geq 0$ and all $b \geq 2$; $(q, 8 \cdot 2^a, 2^b) \in \mathcal{T}$ for all $a \geq 0$ and all $b \geq 2$.*

A consequence of Theorem 3 is the following theorem, which again holds without further conditions on q.

Theorem 4. ([Ha]) *Let $F = GF(q)$ be a finite field with characteristic p and let r be a prime. Let $\delta = 1$ if $r = p$ or $r \geq 5$, let $\delta = 2$ if $r = 3$ and $r \neq p$, and let $\delta = 3$ if $r = 2$ and $r \neq p$. For an integer $n \geq 0$ let $E_{r,n} := GF(q^{r^{\delta n}})$ and let $\mathcal{L}_{F,r} = \{E_{r,n} \mid n \geq 0\}$. Then there exists a trace-compatible sequence w for $\mathcal{L}_{F,r}$ which is primitive and normal over F.*

In general, the decision whether (q, k, e) belongs to $\mathcal{T}$ is a very difficult problem, which is more harder the smaller the parameter e is. For example $(q, k, 1) \in \mathcal{T}$ if and only if **every** normal element of $\mathrm{GF}(q^k)$ over $\mathrm{GF}(q)$ is primitive in $\mathrm{GF}(q^k)$. This strange condition holds e.g. when $(q^k - 1)/(q - 1)$ is a prime, whence necessarily q and k are primes. Thus, $(2, k, 1) \in \mathcal{T}$ for all Mersenne primes $2^k - 1$ (e.g., for $n = 859433$). Also, if $e = 2$ then not much is known, and for $e = 3$ the methods in [Ha] only yield asymptotic but no concrete bounds for q.

References.

[CiD] Computeralgebra in Deutschland (Bestandsaufnahme, Möglichkeiten, Perspektiven), Herausgegeben von der Fachgruppe Computeralgebra der GI, DMV, GAMM, Passau und Heidelberg (1993).

[CoHa] *S. D. Cohen* and *D. Hachenberger*, Primitive normal bases with prescribed trace, AAECC, to appear.

[Ha] *D. Hachenberger*, Primitive normal bases for towers of field extensions, submitted.

[LeSc] *H. W. Lenstra, Jr.* and *R. J. Schoof*, Primitive normal bases for finite fields, Math. Comp. **48** (1987), 217-231.

[MoMu] *I. H. Morgan* and *G. L. Mullen*, Primitive normal polynomials over finite fields, Math. Comp. **63** (1994), 759-765.

[Sche] *A. Scheerhorn*, Trace- and norm-compatible extensions of finite fields, AAECC **3** (1992), 199-209.

Bounds on the Bilinear Complexity of Multiplication in Any Extension of $\mathbb{F}_q$

S. Ballet, C.N.R.S. Institut de Mathématiques de Luminy
Luminy Case 907, F13288 Marseille CEDEX 9
e-mail: ballet@iml.univ-mrs.fr

1 Introduction - Notations

1.1 Bilinear complexity of multiplication

Let $\mathbb{F}_q$ be a finite field with q elements where q is a prime power and let $\mathbb{F}_{q^n}$ be a $\mathbb{F}_q$ extension of degree n. The multiplication M is a bilinear map from $\mathbb{F}_{q^n} \times \mathbb{F}_{q^n}$ into $\mathbb{F}_{q^n}$, thus we can also represent M by a tensor $t_M = \sum_{l=1}^{\lambda} a_l \otimes b_l \otimes c_l \in \mathbb{F}_{q^n}^* \bigotimes \mathbb{F}_{q^n}^* \bigotimes \mathbb{F}_{q^n}$ where $\mathbb{F}_{q^n}^*$ denotes the dual of $\mathbb{F}_{q^n}$ as a $\mathbb{F}_q$-vector space. Hence the product of two elements x and y of $\mathbb{F}_{q^n}$ is :

$$x.y = t_M(x \otimes y) = \sum_{l=1}^{\lambda} a_l(x) b_l(y) c_l. \tag{1}$$

Every expression (1) is called a bilinear multiplication algorithm $\mathcal{U}$. The number λ is called the multiplicative complexity $\mu(\mathcal{U})$ of $\mathcal{U}$. Let us set

$$\mu_q(n) = \min_{\mathcal{U}} \mu(\mathcal{U})$$

where $\mathcal{U}$ is running over all bilinear multiplication algorithms in $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$. Then $\mu_q(n)$ is called the bilinear complexity of multiplication in $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$, and it corresponds to the minimum possible number of summands in any decomposition of tensor t_M.

1.2 Known results

For any integer n, the bilinear complexity $\mu_q(n)$ of multiplication in $\mathbb{F}_{q^n}$ is $\geq 2n-1$ by a result of Winograd [8]. Indeed, the multiplication in $\mathbb{F}_{q^n}$ consists on multiplying two polynomials of degree $\leq n-1$ modulo an irreducible polynomial of degree n. So, the result follows from the notion of interpolation. Then it is necessary to consider two cases:

a) If $\mathbb{F}_q$ is sufficiently large

De Groote [4] has shown that $\mu_q(n) = 2n - 1$ if $n \leq \frac{1}{2}q + 1$. Moreover, Winograd [8] has given a characterization of the class of algorithms realizing this complexity: these are interpolation algorithms of the projective line over $\mathbb{F}_q$ i.e polynomial interpolation algorithms.

b) If $\mathbb{F}_q$ is not sufficiently large

D.V. Chudnovsky and G.V. Chudnovsky in [2] have succeeded in obtaining a principle of construction of fast multiplication algorithms by using interpolation on algebraic curves. This principe generalizes the polynomial interpolation. Moreover, Shparlinsky, Tsfasman and Vladut in [6] have studied this principle, and found new asymptotic bounds. More precisely, they have proved that for any prime power $q \geq 3$, $M_{q^2} \leq 2(1 + \frac{1}{q-2})$, $M_q \leq 6(1 + \frac{1}{q-2})$ and $M_2 \leq 27$ where $M_q = lim\ sup_{n\to\infty} \mu_q(n)/n$. By using the D.V. and G.V. Chudnovsky algorithm applied to elliptic curves, M.A Shokrollahi has shown in [5] that the bilinear complexity of multiplication is equal to $2n$ for $\frac{1}{2}q + 1 < n < \frac{1}{2}(q + 1 + 2\epsilon(q))$ where ϵ is the function defined by:

$$\epsilon(q) = \begin{cases} \text{greatest integer} \leq 2\sqrt{q} \text{ prime to } q & \text{if } q \text{ is not a perfect square} \\ 2\sqrt{q} & \text{if } q \text{ is a perfect square.} \end{cases}$$

1.3 New results presented in this paper.

From the existence of algebraic function fields having some good properties, we obtain some new upper bounds on the bilinear complexity of multiplication in any extension of the finite field $\mathbb{F}_q$ where q is an arbitrary prime power q. More precisely, we prove the following result :

Theorem 1.1 *Let q be an arbitrary prime power and let n be a integer. Then the bilinear complexity of multiplication in any finite field $\mathbb{F}_{q^n}$ is such that $\mu_q(n) \leq B_q n$ where B_q is defined by:*

$$B_q = \begin{cases} 6(1 + \frac{q}{q-3}) & \text{if } q > 3 \\ 2(1 + \frac{\sqrt{q}}{\sqrt{q}-3}) & \text{if } q > 9 \text{ and } q \text{ is a perfect square} \\ 45 & \text{if } q = 3 \\ 90 & \text{if } q = 2. \end{cases}$$

So, we prove that the bilinear complexity of multiplication in the finite field $\mathbb{F}_{q^n}$ is linear uniformly in q with respect to the degree n. To this end, we use the following general theorem established in [1]:

Theorem 1.2 *Let q be a prime power and n be a natural number. If there exists an algebraic function field $F/\mathbb{F}_q$ of genus g, satisfying the following conditions:*
1) $F/\mathbb{F}_q$ contains a prime divisor Q of degree n.
2) $N(F/\mathbb{F}_q) > 2n + 2g - 2$.
Then

$$\mu_q(n) \leq 2n + g - 1.$$

A very important corollary of this theorem is the following theorem:

Theorem 1.3 *Let q be a prime power and n be a natural number. If there exits an algebraic function field $F/\mathbb{F}_q$ of genus g, satisfying the following conditions:*
1) $2g + 1 \leq q^{\frac{n-1}{2}}(q^{\frac{1}{2}} - 1)$ (in particular if $g \leq \frac{n-3}{4}$)
2) $N(F/\mathbb{F}_q) > 2n + 2g - 2$.
Then

$$\mu_q(n) \leq 2n + g - 1.$$

Then, we show in section 2 that the tower of Artin-Schreier extensions of function fields attaining the Drinfeld-Vladut bound, described by A. Garcia and H. Stichtenoth in [3], satisfies the properties of Theorem 1.3. From it, we deduce Theorem 1.1 in section 3.

2 Existence of algebraic function fields having required properties

In this section, we consider the tower $F_1 \subsetneqq F_2 \subsetneqq F_3 \subsetneqq \ldots$ of function fields $F_i/\mathbb{F}_{q^2}$ over $\mathbb{F}_{q^2}$ such the ratio N_i/g_i tends to the Drinfeld-Vladut bound $A(q^2) = q - 1$, constructed by A. Garcia and H. Stichtenoth in [3]. From it, we show that for any $q \geq 3$ and for almost all $n \in \mathbb{N}$, there exists an algebraic function field having more than $2n + 2g - 2$ places of degree one. First, let us recall the two following results established by A. Garcia and H. Stichtenoth in [3].

Theorem 2.1 *The genus $g_k = g(F_k)$ is given by the following formula:*

$$g_k = \begin{cases} q^k + q^{k-1} - q^{\frac{k+1}{2}} - 2q^{\frac{k-1}{2}} + 1, & \text{if } k \equiv 1 \ mod\ 2 \\ q^k + q^{k-1} - \frac{1}{2}q^{\frac{k}{2}+1} - \frac{3}{2}q^{\frac{k}{2}} - q^{\frac{k}{2}-1} + 1, & \text{if } k \equiv 0 \ mod\ 2 \end{cases}$$

Let N_k be the number of places of $F_k/\mathbb{F}_{q^2}$ of degree one. Then, for any $k \geq 3$, we have:

$$N_k \geq (q^2 - 1).q^{k-1} + 2q$$

Corollary 2.1 $\lim_{k\to\infty} N_k/g_k = q - 1$

From now, let us set $M_k = (q^2 - 1).q^{k-1} + 2q$.

Definition 2.1 *Let*

$$\Delta_{q,k} = M_k - (2g_k - 2) = (q^2 - 2q - 3)q^{k-1} + f(k)$$

where

$$f(k) = \begin{cases} 2q^{\frac{k+1}{2}} + 4q^{\frac{k-1}{2}} + 2q, & \text{if } k \equiv 1 \ mod\ 2 \\ q^{\frac{k}{2}+1} + 3q^{\frac{k}{2}} + 2q^{\frac{k}{2}-1} + 2q, & \text{if } k \equiv 0 \ mod\ 2 \end{cases}$$

Let us consider the set

$$\Theta_{q,k} = \{n \in \mathbb{N} | \Delta_{q,k} > 2n\},$$

which, if $q \geq 3$, is not empty. Then we will call upper ray of $F_k/\mathbb{F}_{q^2}$:

$$R_{q,k} = sup\ \Theta_{q,k} = card\ \Theta_{q,k} - 1 = \begin{cases} \lfloor \frac{\Delta_{q,k}}{2} \rfloor & \text{if } \Delta_{q,k} \text{ is odd.} \\ \frac{\Delta_{q,k}}{2} - 1 & \text{if } \Delta_{q,k} \text{ is even.} \end{cases}$$

Let us consider the set

$$\Phi_{q,k} = \{n \in \mathbb{N} | 2g_k + 1 \leq q^{n-1}(q-1)\}$$

which is clearly not empty. Then we will call lower ray of $F_k/\mathbb{F}_{q^2}$:

$$\Gamma_{q,k} = inf\ \Phi_{q,k}$$

and we will call action domain of $F_k/\mathbb{F}_{q^2}$, the set $I_{q,k}$ defined by:

$$I_{q,k} = \Theta_{q,k} \cap \Phi_{q,k} = [\Gamma_{q,k}, R_{q,k}]$$

First, we are going to consider the case $q > 2$.

Lemma 2.1 *Let q be a prime power ≥ 3. Then $(\Delta_{q,k})_{k\geq 3}$ is an increasing sequence such that $\lim_{k\to\infty} \Delta_{q,k} = \infty$.*

Proof : Let q be an arbitary prime power. For any $k \geq 3$, $\Delta_{q,k} = (q^2 - 2q - 3)q^{k-1} + f(k)$ where $f(k)$ is a positive and increasing function. Consequently, if $q > 2$, $(\Delta_{q,k})_{k\geq 3}$ is increasing and tends to infinity. □

Lemma 2.2 *Let q be a prime power ≥ 3. Then for any $k \geq 3$, the action domain of $\Theta_{q,k}$*

$$I_{q,k} = \Theta_{q,k} \cap \Phi_{q,k} = [\Gamma_{q,k}, R_{q,k}]$$

is not empty.

Proof : It follows from $\Gamma_{q,k} < R_{q,k}$ for any $k \geq 3$ and for any $q \geq 3$. Indeed, it is clear that g_k satisfies $2g_k + 1 \leq 2q^{k-1}(q+1)$. Moreover, for any $q \geq 3$, if $n \geq k+2$, then we have $2q^{k-1}(q+1) \leq q^{n-1}(q-1)$. Thus $\Gamma_{q,k} \leq k+2$. Further, for any $q \geq 3$, we know that $\Delta_{q,k} \geq f(k)$ which yields $R_{q,k} \geq \frac{f(k)}{2}$. Consequently, as for any $k \geq 3$ we have $k + 2 < \frac{f(k)}{2}$, we obtain $R_{q,k} > \Gamma_{q,k}$ and the proof is complete. □

Lemma 2.3 *Let q be a prime power ≥ 3. Then for any $k \geq 3$, $I_{q,k} \cap I_{q,k+1}$ is not empty.*

Proof : It follows from $\Gamma_{q,k+1} < R_{q,k}$ for any $k \geq 3$ and for any $q \geq 3$. Indeed, in the proof of Lemma 2.2, for any $k \geq 3$ and for any $q \geq 3$ we obtained $\Gamma_{q,k} \leq k+2$. Thus we have $\Gamma_{q,k+1} \leq k+3$. Moreover, $R_{q,k} \geq \frac{f(k)}{2}$. Consequently, as for any $k \geq 3$ we have $k + 3 < \frac{f(k)}{2}$, we obtain $R_{q,k} > \Gamma_{q,k+1}$ and the proof is complete. □

Lemma 2.4 *Let $J_{q,\infty} = \bigcup_{k=3}^{\infty} I_{q,k}$, then $J_{q,\infty}$ is a covering of $[\Gamma_{q,3}, \infty[\subset \mathbb{N}$.*

Proof : First, $\Gamma_{q,k}$ is an increasing function for any $k \geq 3$. Consequently, the result immediatly follows from Lemma 2.2 and Lemma 2.3. □

Proposition 2.1 *Let q be a prime power ≥ 3. Then for any $n > \Gamma_{q,3}$, there exists an algebraic function field of genus g_k having more than $2n + 2g_k - 2$ places of degree one and at least a place of degree n. In particular, it is true for n such that $n > \frac{1}{2}q^2 + 1$.*

Proof : By [7] Corollary v.2.10.c p179, for any n such that $2g+1 \leq q^{(n-1)}(q-1)$, there exists at least one place of degree n over an algebraic function field $F/\mathbb{F}_q$ of genus g. Consequently, by the definition of $I_{q,k}$ and by Lemma 2.4, for any $n \in I_{q,\infty}$, there exists an algebraic function field of genus g_k having more than $2n+2g_k-2$ places of degree one and at least a place of degree n. Moreover, for any $q \geq 3$, $\Gamma_{q,3} < \frac{1}{2}q^2+1$ and the proof is complete. □

3 Bounds on the multiplication complexity

In this section, we give some new bounds obtained from the results of section 2 and Theorem 1.3.

Theorem 3.1 *Let q be a prime power such that $q \geq 3$ and let g_k be the integer such that for any $k \geq 3$,*

$$g_k = \begin{cases} q^k + q^{k-1} - q^{\frac{k+1}{2}} - 2q^{\frac{k-1}{2}} + 1, & \text{if } k \equiv 1 \ mod\ 2 \\ q^k + q^{k-1} - \frac{1}{2}q^{\frac{k}{2}+1} - \frac{3}{2}q^{\frac{k}{2}} - q^{\frac{k}{2}-1} + 1, & \text{if } k \equiv 0 \ mod\ 2 \end{cases}$$

and

$$M_k = (q^2-1).q^{k-1} + 2q.$$

Then for all integers $n \in \mathbb{N}$ such that $n > \frac{1}{2}q^2+1$, we have

$$2n \leq \mu_{q^2}(n) \leq 2n + g_{k_0} - 1$$

where k_0 is the integer such that $g_{k_0} = min\{g_k | n < \frac{M_k - 2g_k + 2}{2}\}$.

Proof : It follows directly from Proposition 2.1 and Theorem 1.3. □

Corollary 3.1 *Let q be an arbitrary prime power such that $q > 3$. Then for any integer n,*

$$\mu_{q^2}(n) \leq 2(1 + \frac{q}{q-3})n.$$

Moreover, if $\frac{1}{2}q^{k-1}(q^2-2q-3) \leq n < \frac{1}{2}q^{k-1}(q^2-2q-3) + \frac{f(k)}{2}$ where

$$f(k) = \begin{cases} 2q^{\frac{k+1}{2}} + 4q^{\frac{k-1}{2}} + 2q, & \text{if } k \equiv 1 \ mod\ 2 \\ q^{\frac{k}{2}+1} + 3q^{\frac{k}{2}} + 2q^{\frac{k}{2}-1} + 2q, & \text{if } k \equiv 0 \ mod\ 2 \end{cases}$$

then

$$\mu_{q^2}(n) \leq 2(1 + \frac{1}{q-3})n.$$

Proof : For any integer n, let k be the smallest such that $2n < M_k - 2g_k + 2$, then $2n \geq M_{k-1} - 2g_{k-1} + 2$. Consequently, we have $k - 1 \leq log_q(2n) - log_q(q^2 - 2q - 3) + 1$. Moreover, we have $\mu_{q^2}(n) \leq 2n + g_k - 1 \leq 2n + q^k + q^{k-1}$ by Theorem 3.1, then $\mu_{q^2}(n) \leq 2n + q^{log_q(2n) - log_q(q^2-2q-3)+1}(q+1)$, i.e $\mu_{q^2}(n) \leq 2(1 + \frac{q}{q-3})n$. Further, if $\frac{1}{2}q^{k-1}(q^2 - 2q - 3) \leq n < \frac{1}{2}q^{k-1}(q^2 - 2q - 3) + \frac{f(k)}{2}$, then we have $k - 1 \leq log_q(2n) - log_q(q^2 - 2q - 3)$. Consequently, we obtain $\mu_{q^2}(n) \leq 2n + g_k - 1 \leq 2n + q^k + q^{k-1} \leq 2n + q^{log_q(2n) - log_q(q^2-2q-3)}(q+1)$, i.e $\mu_{q^2}(n) \leq 2(1 + \frac{1}{q-3})n$ and the proof is complete. □

Corollary 3.2 *Let q be an arbitrary prime power such that $q > 3$. Then for any integer n,*

$$\mu_q(n) \leq 6(1 + \frac{q}{q-3})n. \tag{2}$$

For $q = 3$, $\mu_3(n) \leq 45n$.
For $q = 2$, $\mu_2(n) \leq 90n$.

Proof : By Lemma 1.2 in [6], for all integers n and m, we have $\mu_q(n) \leq \mu_q(mn) \leq \mu_q(m)\mu_{q^m}(n)$. Hence for $q > 3$, we put $m = 2$ and use $\mu_q(2) = 3$ for any q. Then the inequality (3) follows from Corollary 3.1. For $q = 3$ we put $m = 2$, then we use the inequality (2.3) and $\mu_3(2) = 3$. For $q = 2$ we put $m = 4$, then we use the first inequality of Corollary 3.1 and $\mu_2(4) = 9$ by [2]. □

References

[1] **BALLET, S** *Curves with Many Points and Multiplication Complexity in Any Extension of* $\mathbb{F}_q$. Institut de Mathématiques de Luminy, Preprint 98-20, Juin 1998 (Submit for publication).

[2] **CHUDNOVSKY, D.V, CHUDNOVSKY, G.V.** *Algebraic Complexities and Algebraic Curves over Finite Fields.* J. Complexity, 4, 285-316, 1988.

[3] **GARCIA, A., STICHTENOTH, H.** *A tower of Artin-Schreier extensions of function fields attaining the Drinfeld-Vladut bound.* Inventiones Mathematicae, 121, 211-222, 1995.

[4] **GROOTE, H.F.** *Characterization of Division Algebras of Minimal Rank and the Structure of their algorithm Varieties.* Lectures Notes in Computer Science, 245, Springer-Verlag, New York, Berlin, Heidelberg, Tokyo, 1985.

[5] **SHOKROLLAHI, M.A.** *Optimal Algorithms for multiplication in certain finite fields using algebraic curves.* SIAM J. Comp. 21, No 6, 1193-1198, 1992.

[6] **SHPARLINSKY, I.E, TSFASMAN , M.A, VLADUT, S.G.** *Curves with Many Points and Multiplication in Finite Fields.* Lectures Notes in Mathematics, 1518, 145-169, Springer-Verlag, Berlin 1992.

[7] **STICHTENOTH, H.** *Algebraic Function Fields and Codes.* Springer-Verlag, Berlin, Heidelberg, New York, 1993.

[8] **WINOGRAD, S.** *On Multiplication in Algebraic Extension Fields.* Theorical Computer Science, 8, 359-377, 1979.

The a-invariant of some Reed-Muller type codes

C. Rentería (ESFM, IPN, México, renteri@esfm.ipn.mx)[1]
H. Tapia-Recillas(*)(Dpto. Mat. UAM-I, México, htr@xanum.uam.mx)[2]

1 Introduction

Let K =GF(q) be a finite field with q elements, let $\mathbb{P}_m(K)$ be the m-projective space over K and let $S = \{P_1, \ldots, P_s\}$ be a subset of $\mathbb{P}_m(K)$. Let $\mathcal{L}$ be a finite dimensional K-linear space of functions which are defined on the set S and take values on K. Then the evaluation map:

$$ev : \mathcal{L} \to K^s, \quad ev(f) = (f(P_1)), \ldots, f(P_s))$$

defines a K-linear code: $C_S = ev(\mathcal{L})$.

Let $A = K[X_0, \ldots, X_m] = \oplus_{j\geq 0} A_j$ be the polynomial ring over the finite field K with the natural graduation. If $S \subseteq \mathbb{P}_m(K)$ is as above and $\mathcal{L} = A_d$ is the d-graded homogeneous component of the polynomial ring A, the corresponding linear code $C_S(d) := ev(A_d)$, will be called the Reed-Muller linear code over the set S, which is isomorphic to $A_d/I_S(d)$, where $I_S = \oplus_{j\geq 0} I_S(j)$ is the (graded) vanishing ideal of S. The dimension of this code is given by the Hilbert function of A/I_S. In some cases the minimal distance of this code has been determined ([6], [12]). A generating matrix of this code can be obtained by finding a Gröbner basis for the ideal I_S, so that the cosets with respect to I_S of monomials of degree d that are not in the leading terms ideal $LT(I_S)$ of I_S, form a K-basis for $A_d/I_S(d)$. If $\mathbf{B} \subset R_d$ is this set of monomials then $(ev(M))_{M\in\mathbf{B}}$ is a generating matrix for $C_S(d)$.

The knowledge of the so-called *a-invariant* of the ideal I_S helps to find its Hilbert function, i.e., the dimension of the Reed-Muller code as defined above as well as generators for the ideal I_S, and therefore a generating matrix of the code. The purpose of this note is to determine the a-invariant, a set of generators for the ideal I_S when the set S is the image of the m-projective space over a finite field under the Veronese mapping, and give some examples to illustrate te ideas.

2 Some general results

[1]Partially supported by COFAA–IPN and SNI–SEP, México
[2]Partially supported by CONACyT grant No.L0076-E9607 and SNI-SEP, México.

In this section we recall the definition of the Hilbert function, the *a-invariant* of an ideal and give some results that will be useful later on (cf. [1], [4]).

Let K be any field, let $S \subseteq \mathbb{P}_m(K)$ be a finite set with cardinality s and let $A = K[X_0, \ldots, X_m] = \oplus_{j \geq 0} A_j$ be as above. Let $I := I_S = \{f \in A : f(P) = 0 \ \forall \ P \in S\}$ be the homogeneous vanishing ideal of S in A and let $R = A/I$ be the coordinate ring of S. The *Hilbert function* of R is defined as $H(R, d) := \dim_K A_d - \dim_K I_d$, which will also be denoted by $H_S(d)$, for all $d \in \mathbb{Z}$.

Let $I = \oplus_{r=\gamma_S}^{\infty} I_r$ with $I_{\gamma_S} \neq 0$, so that γ_S is the lowest degree of a non-trivial homogeneous component of the ideal I. There is an integer a_S called the *a-invariant* of R (or the a-invariant of the ideal I_S, or even the a-invariant of S) such that:

1. $H_S(d) = \dim_K A_d = \binom{m+d}{d}$ if and only if $d < \gamma_S$.
2. $H_S(d) < H_S(d+1) < s$ for $0 \leq d < a_S$.
3. $H_S(d) = s$ for $d > a_S$.

The number $a_S + 1$ is called the *regularity index* of A/I_S.

We also recall that a graded free resolution of A/I_S is an exact sequence of the form:

$$0 \to \oplus A(-\alpha_{ir})^{\beta_{ir}} \to \cdots \to \oplus A(-\alpha_{i1})^{\beta_{i1}} \to A \to A/I_S \to 0$$

so that by restriction to the d component we get the general formula for the Hilbert function of R:

$$k_d = \binom{m+d}{d} + \sum_{j=1}^{r} (-1)^j \sum_i \beta_{ij} \binom{m+d-\alpha_{ij}}{d-\alpha_{ij}}$$

In particular if K is a finite field, this relation gives the dimension of the code $C_S(d)$.

We now recall that the *Hilbert series* of S is $F_S(t) = \sum_{j=0}^{\infty} H_S(j) t^j$, and that it is a rational function of the form $F_S(t) = \frac{p(t)}{(1-t)^{1+n}}$, where $p(t)$ is a polynomial with integral coefficients. The next result is useful in practical situations for determining the a-invariant when the Hilbert series is known.

Lemma 1 *With the notation as above, the a-invariant a_S of S is equal to* $\deg(p(t)) - (1+n)$.

The proof of this lemma is easy and is omitted (cf.[11]).

Let K be a field and let $\mathbb{P}_m(K)$ be the m-projective space over K. If $\underline{x} = (x_0, ..., x_m)$ is an element of $\mathbb{P}_m(K)$ the *Veronese map* is defined as

$$v_n : \mathbb{P}_m(K) \to \mathbb{P}_N(K), \quad v_n(\underline{x}) = (..., M(\underline{x}), ...)$$

where $M(\underline{X})$ runs over all the monomials of degree n in the variables $\underline{X} = (X_0, ..., X_m)$, and $N = \begin{pmatrix} n+m \\ n \end{pmatrix} - 1$. If K is algebraically closed, it is well known that v_n is a smooth embedding with the property that every hypersurface of degree n in $\mathbb{P}_m(K)$ becomes a hyperplane section of the Veronese variety, the image of $\mathbb{P}_m(K)$ under the mapping v_n. For $m = 1$ the image of v_n is the *rational normal curve* of degree n, and for $m = 2$ the image is the *Veronese surface.* In the sequel an element of $\mathbb{P}_N(K)$ will be denoted by $\underline{y} = (y_0, ..., y_N)$, the coordinate ring of $\mathbb{P}_m(K)$ will be denoted by $A_m = K[X_0, ..., X_m] = \oplus_{j\geq 0} A_m(j)$ and the corresponding ring of $\mathbb{P}_N(K)$ by $A_N = K[Y_0, ..., Y_N] = \oplus_{j\geq 0} A_N(j)$, both with the standard graduation.

From now on we restrict the above construction to the case where K is the finite field $\mathrm{GF}(q)$ with $q = p^r$ elements (p prime and r a positive integer), and let $S = v_n(\mathbb{P}_m(K)) = \{v_n(P) \in \mathbb{P}_N(K) : P \in \mathbb{P}_m(K)\}$.
Observe that since v_n is an embedding then $\#(S) = \pi_m = \#(\mathbb{P}_m(K)) = \frac{q^{m+1}-1}{q-1}$. Let I_S be the graded vanishing ideal of the set S, i.e., $I_S = \{f(\underline{Y}) \in A_N : f(P) = 0 \ \forall\ P \in S\} = \oplus_{j\geq 0} I_S(j)$. In [10] the a-invariant of the vanishing ideal of the affine and projective space was found, and in [11] the a-invariant and a set of generators for the defining ideal of the rational normal curve were determined. In this note the a-invariant as well as the vanishing ideal of the set S will be determined.

3 The a-invariant of S

For a positive integer d and for an element $f \in A_N(d)$, let $\varphi_f(\underline{X}) = f(M_0(\underline{X}), ..., M_N(\underline{X}))$. Then φ_f is homogeneous of degree nd and it defines a mapping $\varphi : A_N(d) \to A_m(nd)$ which induces an isomorphism between the K-vector spaces $A_N(d)/I_S(d)$ and $A_m(nd)/I_{\mathbb{P}_m}(nd)$. In particular it follows

that

$$H_S(d) = H_{\mathbb{P}_m}(nd) = \sum_{j=0}^{m}\sum_{i=0}^{j}(-1)^i\binom{j}{i}\binom{j+nd-1-iq}{nd-1-iq} \qquad (1)$$

(for the last equality see [10], where we assume that $\binom{d}{d} = 1$ for all integers d and that $\binom{k}{t} = 0$ for $k \neq t$ and $t < 0$). We also recall that the a-invariant of the projective space $\mathbb{P}_m(K)$ is $a_m = m(q-1)$ and that $I_{\mathbb{P}_m} =< X_i^q X_j - X_i X_j^q,$ $i < j;\ i,j = 0,1,...,m >$ (cf. [10]).

Lemma 2 *The a-invariant a_S of the set $S = v_n(\ \mathbb{P}_m(K))$ is equal to:*
$\frac{a_m-1-j}{n}$ *if* $a_m + 1 \equiv j$ *mod.n and* $j > 0$
$\frac{a_m+1}{n} - 1$ *if* $a_m + 1 \equiv 0$ *mod.n*

Proof: If $a_m + 1 \equiv j \mod n$ with $0 < j < n$ let $d = \frac{a_m-1-j}{n}$, then $H_S(d+1) = H_{\mathbb{P}_m}(n(d+1))$. Since $n(d+1) = a_m+1-j+n$ and $n-j > 0$, it follows that $n(d+1) > a_m$ and consequently $H_S(d+1) = \#(S)$. Furthermore, since $j > 0$, we have that $a_m+1-j < a_m+1$ and hence $H_S(d) = H_{\mathbb{P}_m}(nd) = H_{\mathbb{P}_m}(a_m+1-j) < \#(S)$. From the definition of the a-invariant it follows that $d = a_S$. The case $j = 0$ is similar since if $d = \frac{a_m+1}{n}-1$, then $H_S(d) = H_{\mathbb{P}_m}(nd)$ $= H_{\mathbb{P}_m}(a_m + 1 - n) < \#(\mathbb{P}_m) = \#(S)$, and $H_S(d+1) = H_{\mathbb{P}_m}(n(d+1)) = H_{\mathbb{P}_m}(a_m + 1) = \#(\mathbb{P}_m) = \#(S)$, showing that $a_S = \frac{a_m+1}{n} - 1$.

4 The vanishing ideal of S

In this section the vanishing ideal of S, i.e., the image of $\mathbb{P}_m(K)$ under the Veronese mapping is described. In order to do this let $\overline{K}$ be the algebraic closure of K and let $\overline{S}$ be the Veronese variety, i.e., the image of $\mathbb{P}_m(\overline{K})$ under the mapping $v_n : \mathbb{P}_m(\overline{K}) \to \mathbb{P}_N(\overline{K})$, $v_n(\underline{x}) = (..., M(\underline{x}), ...)$. If $A = K[Z_0, ..., Z_t]$ let $\overline{A} = \overline{K}[Z_0, ..., Z_t]$ and let $I_{\overline{S}}$ be the vanishing ideal of $\overline{S}$.

Theorem 3 *For $q \equiv l$ mod.n, $0 \leq l \leq n-1$, let $r = \frac{q-l}{n}$. Then*
$I_S =< I_S(2), I_S(r+1) >$.

Proof: Since $I_{\overline{S}} = \oplus_{d\geq 2} I_{\overline{S}}(d)$, in order to prove the assertion of the theorem it is enough to show that $I_S(d) = \sum_{i=0}^{N} Y_i I_S(d-1)$, for all $d \geq 3$ and $d \neq r+1$. We consider two cases: a) $3 \leq d \leq r$, and b) $d \geq r+2$. First we observe that if d is any positive integer such that $nd < q+1$, since $I_{\mathbb{P}_m} =< I_{\mathbb{P}_m}(q+1) >$

then $\varphi_f = 0$ for all $f \in I_S(d)$, (where φ_f is as defined in section 3), showing that $I_S(d) \subseteq I_{\overline{S}}(d)$. Since $I_{\overline{S}}(d) =< I_{\overline{S}}(2) >$ then for all $d \geq 3$, we have $I_{\overline{S}}(d) = \sum_{i=1}^{N} Y_i I_{\overline{S}}(d-1)$. In the case a): $3 \leq d \leq r$, i.e., $3n \leq nd \leq q-l < q+1$, let $W = \sum_{i=1}^{N} Y_i I_S(d-1)$. Since $\dim_K(W) = \dim_{\overline{K}}(W \otimes_K \overline{K})$ and $W \otimes_K \overline{K} \simeq W\overline{A}_N = I_{\overline{S}}(d)$, it follows that $\dim_{\overline{K}} I_{\overline{S}}(d) = \dim_K I_S(d)$, and therefore $W = I_S(d)$. In case b): $r+2 \leq d$, since $\frac{q-l}{n} + 1 \leq d-1$, we have $q + (n-l) \leq n(d-1)$, and hence $q+1 \leq n(d-1) < nd$. Thus if $f \in I_S(d)$ then $\varphi_f \in I_{\mathbb{P}_m}(nd) =< X_i^q X_j - X_i X_j^q,\ 0 \leq i < j \leq m >$, (cf [10]) and consequently $\varphi_f = \sum_{0 \leq i < j \leq m} g_{ij}(X_i^q X_j - X_i X_j^q)$ where the g_{ij}'s are forms of degree $nd - (q+1)$. Since $n(d-1) \geq q+1$, i.e, $nd - (q+1) \geq n$ the g_{ij}'s can be written as: $g_{ij} = M_{ij}(1)h_{ij}(1) + M_{ij}(l_{ij})h_{ij}(l_{ij})$, where the $M_{ij}(k)$ are monomials of degree n and the $h_{ij}(k)$ are forms of degree $nd - (q+1) - n$. Thus

$$\varphi_f = \sum_{i,j} \sum_{k=1}^{l_{ij}} M_{ij}(k) h_{ij}(k)(X_i^q X_j - X_i X_j^q)$$

Note that for all i, j, k, the form $h_{ij}(k)(X_i^q X_j - X_i X_j^q) \in I_{\mathbb{P}_m}(n(d-1))$. Hence $\varphi_f = M_{i1}H_{i1} + ... + M_{is}H_{is}$ with M_{ij} monomials in the variables $X_0, ..., X_m$ of degree n and $H_{ij} \in I_{\mathbb{P}_m}(n(d-1))$. Therefore $f = Y_{i1}\lambda_{i1}(H_{i1}) + ... + Y_{is}\lambda_{is}(H_{is})$ where each $\lambda_{ij}(H_{ij})$ is in $I_S(d-1)$ and is such that $\varphi_{\lambda_{ij}(H_{ij})} = H_{ij}$. From the above argument we conclude that $f \in \sum_{i=0}^{N} Y_i I_S(d-1)$.

5 The case $m = 2$

In the previous section the ideal I_S was described in terms of the homogeneous piece $I_S(2)$, i.e., quadrics and forms of degree $r+1$, i.e., $I_S(r+1)$ (with r as defined in the theorem above). In this section we treat the case of the projective plane $\mathbb{P}_2(K)$ closely and, in particular, the dimension of $I_S(2)$ is determined and a set of generators for the homogeneous component $I_S(r+1)$ is given.

Since for all positive integers d, the K-vector spaces $A_N(d)/I_S(d)$ and $A_m(nd)/I_{\mathbb{P}_m}(nd)$ are isomorphic it follows from relation (1) above that $H_S(n) = H_{\mathbb{P}_2}(2n) = (n+1)(2n+1)$. The number of forms of degree 2 in the variables

$Y_0,\ldots,Y_N$ is $\frac{1}{8}[(n+1)(n+2)][(n+1)(n+2)+2]$. Therefore

$$dim_K I_S(2) = \dim_K A_N(2) - H_S(n) = \frac{1}{8}(n+1)n(n-1)(n+6)$$

Let r be as defined in the previous theorem, i.e., $r = \frac{q-l}{n}$ if $q \equiv l \mod n$. Since $\dim_K I_S(r+1) = \dim_K I_{\mathbb{P}_2}(n(r+1))$, assuming $q+1 \leq nr \leq 2q+1$, it follows from relation (1) of section 3, that $\dim_K I_S(r) = (nr-q+1)^2 - 1$. We recall that if $F_{(0,1)} = X_0^q X_1 - X_0 X_1^q$, $F_{(0,2)} = X_0^q X_2 - X_0 X_2^q$, $F_{(1,2)} = X_1^q X_2 - X_1 X_2^q$, then $I_{\mathbb{P}_2} =< F_{(0,1)}, F_{(0,2)}, F_{(1,2)} >$, (cf.[10]). Let $t = n(r+1) - (q+1)$. Since $F_{(i,j)} \in I_{\mathbb{P}_2}(q+1)$, then $(X_0^a X_1^b X_2^c)F_{(i,j)} \in I_{\mathbb{P}_2}(n(r+1))$ if $a+b+c = t$. The number of monomials $X_0^a X_1^b X_2^c$ with $a+b+c = t$ is $\binom{t+2}{t}$, so there is a total of $3\binom{t+2}{t}$ elements of the form $(X_0^a X_1^b X_2^c)F_{(i,j)}$, $i,j = 1,2,3$. Since $\dim_K I_{\mathbb{P}_2}(n(r+1)) = (t+2)^2 - 1$, it follows that $\dim_K I_{\mathbb{P}_2}(n(r+1)) \leq 3\binom{t+2}{t}$. Thus there are enough elements in $I_{\mathbb{P}_2}(n(r+1))$ of the form $(X_0^a X_1^b X_2^c)F_{(i,j)}$ to choose from so that a basis for $I_{\mathbb{P}_2}(n(r+1))$ can be taken.

Since the Veronese mapping is given by the monomials of degree n in the variables X_0, X_1, X_2, we may order them with the lexicographic ordering, so that for instance we can identify the coordinate Y_0 of $\mathbb{P}_N$ with the monomial X_0^n, the coordinate Y_1 with the monomial $X_0^{n-1}X_1$, and so on. Furthermore, we introduce the following notation: $X_{(a,b,c)} = X_0^a X_1^b X_2^c$, where $a+b+c = n$, so that if $\underline{x} = (x_0, x_1, x_2) \in \mathbb{P}_2$, then for instance $X_{(n,0,0)}(\underline{x}) = x_0^n$ and $X_{(n-1,1,0)}(\underline{x}) = x_0^{n-1}x_1$, etc.

Now let $q = nr + l$, $0 \leq l \leq n-1$, and let $\widetilde{F}_{(i,j)=}(X_0^a X_1^b X_2^c)F_{(i,j)}$, $i,j = 1,2,3$, $a+b+c = t$, be a basis element of $I_{\mathbb{P}_2}(n(r+1))$ as described above. For instance $\widetilde{F}_{(1,2)} = (X_0^a X_1^b X_2^c)F_{(1,2)} = (X_0^a X_1^b X_2^c)(X_1^q X_2 - X_1 X_2^q) = (X_0^a X_1^{nr+l+b} X_2^{c+1}) - (X_0^a X_1^{b+1} X_2^{nr+l+c}) = X_1^{nr}(X_0^a X_1^{l+b} X_2^{c+1}) - (X_0^a X_1^{b+1} X_2^{c+l})X_2^{nr}$. Observe that $a+(b+l)+(c+1) = n = a+(b+1)+(c+l)$ since $a+b+c = n(r+1) - (q+1)$. With the notation introduced above let $G_{(1,2)} = X_{(0,n,0)}^r X_{(a,b+l,c+1)} - X_{(a,b+1,c+l)} X_{(0,0,n)}^r$. Then $G_{(1,2)} \in I_S(r+1)$ and it is easy to see that $\varphi_{G_{(1,2)}} = \widetilde{F}_{(1,2)}$, where the mapping φ is as defined in section 3. Consequently, a set of generators for the ideal $I_S(r+1)$ can be obtained as the "pull-back" under the mapping φ of a set of generators of the ideal $I_{\mathbb{P}_2}(n(r+1))$.

Some examples over specific fields can be described in detail as well.

References

[1] D. Cox, J. Little, D. O'Shea. *Ideals, Varieties, and Algorithms.* UTM, Springer Verlag, 1992.

[2] P. Delsarte, J.M. Goethals, F.J. MacWilliams. On generalized Reed-Muller codes and their relatives. Inform. and Control, vol.16 (1970), pp. 403-422.

[3] I. Duursma, C. Rentería and H. Tapia-Recillas. Reed-Muller codes on Complete Intersections. Preprint (1998).

[4] A.V. Geramita, M. Kreuzer and L. Robbiano. Cayley-Bacharach schemes and their canonical modules. Transactions of the AMS, vol.339, number 1, Sept. (1993), pp. 163-189.

[5] J.P. Hansen. Zero-dimensional schemes. Proc. Int. Conf., Ravello, 1992. F. Orrechia and L. Chiantini (eds.), Walter de Gruyter, Berlin, (1994).

[6] G. Lachaud. The parameters of the projective Reed-Muller codes. Discrete Mathematics 81 (1990), pp. 217-221.

[7] D.R. Grayson, M. Stillman: Macaulay 2 (1998).

[8] D.J. Mercier, R. Rolland. Polynômes homogènes à plusieurs variables sur un corps fini $\mathbf{F_q}$ qui s'annulent sur l'espace projectif $\mathbb{P}_m(\mathbf{F_q})$. J. of Pure and Applied Algebra 124 (1998), pp. 227-240

[9] C. Rentería, H. Tapia-Recillas. Linear codes associated to the ideal of points in $\mathbb{P}^d$ and its canonical module. Communications in Algebra **24** (3), (1996), pp. 1083-1090.

[10] C. Rentería, H. Tapia-Recillas. Reed-Muller Codes: An Ideal Theory Approach. Communications in Algebra, **25**(2), (1997), pp. 401-413.

[11] C. Rentería, H. Tapia-Recillas. The a-invariant of some Reed-Muller Codes. Submitted for publication.

[12] A.B. Sörensen. Projective Reed-Muller codes. IEEE Trans. on Inf. Theory, vol.37, No.6, Nov. (1991), pp. 1567-1576.

MULTIPLICATIVE CHARACTERS AND DESIGN OF SEQUENCES WITH GOOD AUTOCORRELATION

CARINE BOURSIER

G.E.C.T.
Université de Toulon et du Var
83957 LA GARDE cedex, FRANCE
fax : (33).04.94.14.26.33 - tel : (33).04.94.14.22.20
E-mail address : boursier@univ-tln.fr

ABSTRACT. The purpose of this paper is to find sequences with a good autocorrelation function. The complex sequence $s : t \mapsto s(t) = \dot{\chi} \circ \mathrm{tr}_{L/K}(\alpha^t)$ is studied, where K is a finite field with q elements, q a power of a prime, L a finite extension of degree s of K, α a primitive element of L, χ a multiplicative character of $K^\times$ of order k and $\dot{\chi}$ one of the extended mapping in zero by a root of unity or zero. The obtained sequence s is valued in the unit circle, in fact in the set of the k^{th}-roots of unity, plus possibly the origin (if $\dot{\chi}$ is such that $0 \mapsto 0$). We give its autocorrelation function and we show how the period of s depends on $[L : K]$ and on the order of χ. The particular case when χ is quadratic is studied. When the extended character $\dot{\chi}$ is such that $0 \mapsto 1$ or $0 \mapsto -1$, then the sequence is binary and has "almost-constant" or almost-perfect autocorrelation function and when $\dot{\chi}$ is such that $0 \mapsto 0$, then it is an almost-perfect ternary sequence. The cases when χ is of order 3 and 4 are also studied.

1. INTRODUCTION

Let $s : t \mapsto s(t)$ be a complex sequence of period n. The autocorrelation function of s is defined by

$$\begin{array}{rccl} C_s : & \mathbb{Z} & \to & \mathbb{C} \\ & u & \mapsto & C_s(u) = \displaystyle\sum_{x \in \mathbb{Z}/n\mathbb{Z}} \overline{s(x)} s(x+u) \end{array}$$

C_s is also periodic of period n. The autocorrelation function in some $u \equiv 0 \pmod{n}$ are called "in-phase" coefficients and are equal to the weight of s, i.e. the number of non-zero elements of s in one period. The other coefficients $C_s(u)$ for $u \not\equiv 0 \pmod{n}$ are called "out-of-phase" coefficients.

If all the out-of-phase coefficients vanish, the sequence is said to be "perfect". If they all vanish except for one value, the sequence is said

Date: November 26, 1998.

to be "almost-perfect". If all the coefficients are identical except one, the sequence will be said "almost-constant". All those sequences are very interesting for many applications like radar and transmission of message.

In this paper, we study the complex sequence $s : t \mapsto s(t) = \dot{\chi} \circ \mathrm{tr}_{L/K}(\alpha^t)$ where K is a finite field with q elements, q not necessarily prime, L a finite extension of degree s of K, α a primitive element of L, χ a multiplicative character of $K^\times$ of order k and $\dot{\chi}$ one of the extended mapping in 0 by a root of unity or zero. The obtained sequence s is valued in the unit circle, in fact in the set of the k^{th}-root of unity plus possibly the origin (in the case when $\dot{\chi} : 0 \mapsto 0$). We prove that s is of least period $k\frac{q^s-1}{q-1}$ and we give its autocorrelation function. It depends on q, on the degree s of the extension L, the order of χ and the way to extend χ in zero.

In particular, when χ is quadratic and such that $0 \mapsto 1$, then s is binary and has "almost-constant" autocorrelation function, that is a constant autocorrelation function except in one value. In fact, when $s = 2$, the sequence is a 1-almost-perfect in the terminology of [20] sequence and this completes the construction of [11] and [16]. When $s \neq 2$, we obtain sequences with almost-constant autocorrelation for lengths for which there is no almost-perfect sequence (n is not a multiple of 4 and $\frac{n}{2} - 1$ is not a power of a prime). When χ is quadratic and such that $0 \mapsto 0$, then s is an almost-perfect ternary sequence with period $2\frac{q^s-1}{q-1}$ and weight $2q^{s-1}$ and this construction completes the one given in [12].

In the case when χ is of order 3, we obtain 3-phase sequences of length $3(q+1)$ and for which the out-of-peak autocorrelation is zero except for 2 values.

Finally, when χ is of order 4, we obtain 4-phase sequences of length $4(q+1)$ and for which the out-of-peak autocorrelation is zero except for 3 values.

2. Study of $t \to \dot{\chi} \circ \mathrm{tr}_{L/K}(\alpha^t)$

In all the sequel, if n is a positive integer, we will note $\mathbb{Z}_n$ for the set $\{0, 1, ..., n-1\}$ of the integer modulo n and ζ_n for the primitive n^{th}-root of unity $\exp(\frac{i2\pi}{n})$ in $\mathbb{C}^*$.

The strategy to study the autocorrelation function of a sequence s of period n is to use its Fourier transform $\hat{s}$, that is, for u in $\mathbb{Z}_n$:

$$\hat{s}(u) = \sum_{x \in \mathbb{Z}_n} s(x)\zeta_n^{ux}$$

It is well known that $|\hat{s}(u)|^2 = \widehat{C_s}(u)$. Indeed,

$$\begin{aligned}\widehat{C_s}(u) &= \sum_{x\in\mathbb{Z}_n}\Big(\sum_{y\in\mathbb{Z}_n}\overline{s(y)}s(x+y)\Big)\zeta_n^{ux}\\ &= \sum_{y\in\mathbb{Z}_n}\bar{s}(y)\zeta_n^{-uy}\sum_{x\in\mathbb{Z}_n}s(x)\zeta_n^{ux}\\ &= \overline{\hat{s}(u)}\hat{s}(u)\\ &= |\hat{s}(u)|^2\end{aligned}$$

So in order to obtain the autocorrelation function of s, we can evaluate the module of its Fourier transform for each u in $\mathbb{Z}_n$, and then apply Fourier transform inversion formula.

In order to calculate the Fourier transform of the sequence s, we will use a result of Clarke in [2]. Let note $a : \mathbb{F}_q \to \mathbb{C}$ a map, $g = \frac{q^s-1}{q-1}$ and $v = (\alpha^{-1})^g$ a primitive root of K. Let note h the sequence of period at most $q-1$ given by :

$$\begin{array}{rccl} h: & \mathbb{Z} & \to & \mathbb{C} \\ & t & \mapsto & h(t) = a(v^t) \end{array}$$

and the sequence s of period at most $q^s - 1$ given by

$$\begin{array}{rccl} s: & \mathbb{Z}_{q^s-1} & \to & \mathbb{C} \\ & t & \mapsto & s(t) = a(\mathrm{tr}_{L/K}(\alpha^t)) \end{array}$$

Clarke stated the following :

$$\begin{aligned}&\hat{s}(0) = (q^{s-1}-1)a(0) + q^{s-1}\hat{h}(0)\\ &|\hat{s}(u)| = \begin{cases} q^{\frac{s-2}{2}}|(q-1)a(0) - \hat{h}(0)| & \text{if } u \in (q-1)\mathbb{Z}_g - \{0\},\\ q^{\frac{s-1}{2}}|\hat{h}(u)| & \text{if } u \notin (q-1)\mathbb{Z}_g. \end{cases}\end{aligned} \tag{2.1}$$

Applying formula (2.1) in the particular case when a is one of the extended mapping of χ and using character sum theory, Fourier transform inversion formula and orthogonality relations, we state the following theorem :

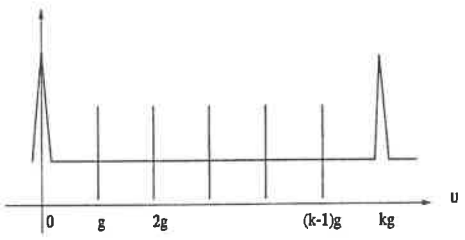
theorem 2.1. *Let the sequence s be*

$$\begin{array}{rccl} s: & \mathbb{Z}_{q^s-1} & \to & \mathbb{C} \\ & t & \mapsto & s(t) = \dot{\chi}\circ\mathrm{tr}_{L/K}(\alpha^t) \end{array}$$

s is a $\frac{q-1}{k}$-times-repetition of the sequence f of length $k\frac{q^s-1}{q-1}$ and $\forall u \in \mathbb{Z}_{k\frac{q^s-1}{q-1}}$, we have

$$\mathrm{f}\times\mathrm{f}(u) = \begin{cases} |\dot{\chi}(0)|^2 k\frac{q^{s-1}-1}{q-1} + kq^{s-1}\zeta_{k\frac{q^s-1}{q-1}}^u & \text{if } u \in \frac{q^s-1}{q-1}\mathbb{Z}_k,\\ |\dot{\chi}(0)|^2 k\frac{q^{s-2}-1}{q-1} & \text{otherwise.} \end{cases}$$

The next scheme gives the modulus of C_f. It is constant except at zero and $(k-1)$ other values :



3. Quadratic case : binary and ternary sequences of period $2\frac{q^s-1}{q-1}$

If q is odd, then $K^\times$ admits a quadratic character. f is binary ($\{+1,-1\}$-valued) or ternary ($\{0,+1,-1\}$-valued), of period $2\frac{q^s-1}{q-1}$ and has the following autocorrelation function :

$$\forall u \in \mathbb{Z}_{2\frac{q^s-1}{q-1}},\ C_{\mathrm{f}}(u) = \begin{cases} |\dot{\chi}(0)|^2 2(\frac{q^{s-1}-1}{q-1}) + 2q^{s-1} & \text{if } u = 0, \\ |\dot{\chi}(0)|^2 2(\frac{q^{s-1}-1}{q-1}) - 2q^{s-1} & \text{if } u = \frac{q^s-1}{q-1}, \\ |\dot{\chi}(0)|^2 2(\frac{q^{s-2}-1}{q-1}) & \text{otherwise.} \end{cases}$$

f is "almost-constant" (all the out-of-phase coefficients are constant except one). Now we can deduce the two following corollaries :

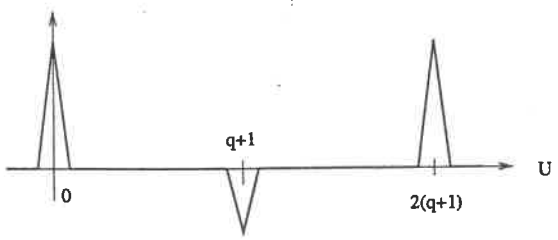
corollary 3.1. *Let α be a primitive root of L, χ the quadratic character of K extended in zero by $0 \mapsto 1$ or $0 \mapsto -1$. The binary $2\frac{q^s-1}{q-1}$-periodic sequence*

$$\begin{array}{rccl} \mathrm{f}: & \mathbb{Z}_{2\frac{q^s-1}{q-1}} & \to & \mathbb{C} \\ & t & \mapsto & \mathrm{f}(t) = \dot{\chi} \circ \mathrm{tr}_{L/K}(\alpha^t) \end{array}$$

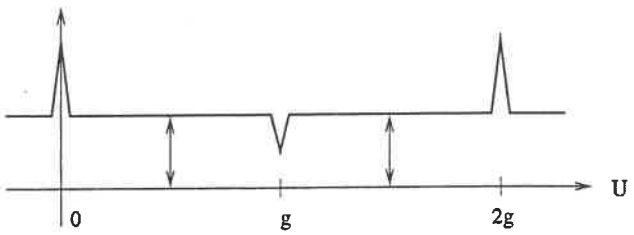
has the following autocorrelation function :

$$\forall u \in \mathbb{Z}_{2\frac{q^s-1}{q-1}},\ C_{\mathrm{f}}(u) = \begin{cases} 2\frac{q^s-1}{q-1} & \text{if } u = 0, \\ 2\frac{q^s-1}{q-1} - 4q^{s-1} & \text{if } u = \frac{q^s-1}{q-1}, \\ 2\frac{q^{s-2}-1}{q-1} & \text{otherwise.} \end{cases}$$

- *If $s = 2$ then f is 1-almost perfect in the terminology of* [20], *of length $2(q+1)$:*



- *If $s \neq 2$ then f is "almost-constant" of length $2\frac{q^s-1}{q-1}$:*



We can note that in the case when $s = 3$, f is almost-perfect of type $IIIb'$ with $\theta = q + 1$ in the terminology of [7] and corresponds to a $(q^2 + q + 1, 2, q^2, 0, \frac{(q-2)(q+1)}{2} + 1)$-relative difference set.

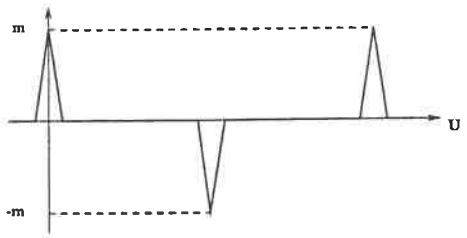
corollary 3.2. *Let α be a primitive root of L, χ the quadratic character of K extended in zero by $0 \mapsto 0$. The ternary $2\frac{q^s-1}{q-1}$-periodic sequence*

$$\begin{array}{rccl} \mathrm{f}: & \mathbb{Z}_{2\frac{q^s-1}{q-1}} & \to & \mathbb{C} \\ & t & \mapsto & \mathrm{f}(t) = \dot{\chi} \circ \mathrm{tr}_{L/K}(\alpha^t) \end{array}$$

has the following autocorrelation function :

$$\forall u \in \mathbb{Z}_{2\frac{q^s-1}{q-1}},\ C_{\mathrm{f}}(u) = \begin{cases} 2q^{s-1} & \text{if } u = 0, \\ -2q^{s-1} & \text{if } u = \frac{q^s-1}{q-1}, \\ 0 & \text{otherwise.} \end{cases}$$

and if we note $m = 2q^{s-1}$ for the weight of f, *we have :*



4. Three-phase sequences

If $q \equiv 1 \pmod 3$, then $K^\times$ admits a character χ of order 3. If the character χ is extended in zero by $0 \mapsto 1$ or $0 \mapsto \exp(i\frac{2\pi}{3})$ or $0 \mapsto \exp(i\frac{4\pi}{3})$, then f is three-phase ($\{+1, \exp(i\frac{2\pi}{3}), \exp(i\frac{4\pi}{3})\}$-valued), of period $3\frac{q^s-1}{q-1}$ and has the following autocorrelation function :

$$\forall u \in \mathbb{Z}_{3\frac{q^s-1}{q-1}},\ C_{\mathrm{f}}(u) = \begin{cases} 3\frac{q^s-1}{q-1} & \text{if } u = 0, \\ 3\frac{q^{s-1}-1}{q-1} + 3q^{s-1}\exp(i\frac{2\pi}{3}) & \text{if } u = \frac{q^s-1}{q-1}, \\ 3\frac{q^{s-1}-1}{q-1} + 3q^{s-1}\exp(i\frac{4\pi}{3}) & \text{if } u = 2\frac{q^s-1}{q-1}, \\ 3\frac{q^{s-2}-1}{q-1} & \text{otherwise.} \end{cases}$$

In the case when L is an extension of degree 2 of K ($s = 2$), we have the following corollary :

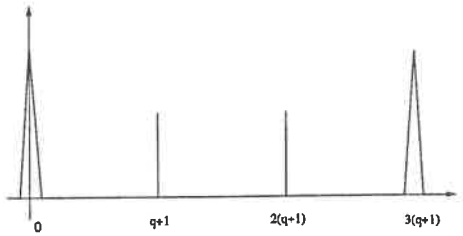
corollary 4.1. *Let χ the character of order 3 of K extended in zero by a third root of unity in $\mathbb{C}$. Let L be the extension of degree 2 of K. Let α be a primitive root of L. The three-phase sequence*

$$\begin{array}{rccl} \mathrm{f}: & \mathbb{Z}_{3\frac{q^s-1}{q-1}} & \to & \mathbb{C} \\ & t & \mapsto & \mathrm{f}(t) = \dot{\chi} \circ \mathrm{tr}_{L/K}(\alpha^t) \end{array}$$

has the following autocorrelation function :

$$\forall u \in \mathbb{Z}_{3(q+1)},\ C_{\mathrm{f}}(u) = \begin{cases} 3(q+1) & \text{if } u = 0, \\ 3(1+q\exp(i\frac{2\pi}{3})) & \text{if } u = q+1, \\ 3(1+q\exp(i\frac{4\pi}{3})) & \text{if } u = 2(q+1), \\ 0 & \text{otherwise.} \end{cases}$$

and the modulus of C_{f} is the following :



5. Four-phase sequences

If $q \equiv 1 \pmod 4$, then $K^\times$ admits a character χ of order 4. If the character χ is extended in zero by $0 \mapsto 1$ or $0 \mapsto -1$ or $0 \mapsto i$ or $0 \mapsto -i$, then f is four-phase ($\{\pm 1, \pm i\}$-valued), of period $4\frac{q^s-1}{q-1}$ and has the following autocorrelation function :

$$\forall u \in \mathbb{Z}_{4\frac{q^s-1}{q-1}},\ C_{\mathrm{f}}(u) = \begin{cases} 4\frac{q^s-1}{q-1} & \text{if } u = 0, \\ 4\frac{q^{s-1}-1}{q-1} + 4iq^{s-1} & \text{if } u = \frac{q^s-1}{q-1}, \\ 4\frac{q^{s-1}-1}{q-1} - 4q^{s-1} & \text{if } u = 2\frac{q^s-1}{q-1}, \\ 4\frac{q^{s-1}-1}{q-1} - 4iq^{s-1} & \text{if } u = 3\frac{q^s-1}{q-1}, \\ 4\frac{q^{s-2}-1}{q-1} & \text{otherwise.} \end{cases}$$

In the case when L is an extension of degree 2 of K ($s = 2$), we have the following corollary :

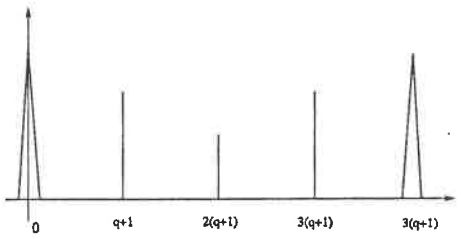
corollary 5.1. *Let χ the character of order 4 of K extended in zero by a fourth root of unity in $\mathbb{C}$. Let L be the extension of degree 2 of K. Let α be a primitive root of L. The four-phase sequence*

$$\begin{array}{rccl} \mathrm{f}: & \mathbb{Z}_{4\frac{q^s-1}{q-1}} & \to & \mathbb{C} \\ & t & \mapsto & \mathrm{f}(t) = \dot{\chi} \circ \mathrm{tr}_{L/K}(\alpha^t) \end{array}$$

has the following autocorrelation function :

$$\forall u \in \mathbb{Z}_{4(q+1)},\ C_f(u) = \begin{cases} 4(q+1) & \text{if } u = 0, \\ 4(1+iq) & \text{if } u = q+1, \\ 4(1-q) & \text{if } u = 2(q+1), \\ 4(1-iq) & \text{if } u = 3(q+1), \\ 0 & \text{otherwise.} \end{cases}$$

and the modulus of C_f is the following :



6. Conclusion

We have constructed sequences of length $n = k\frac{q^s-1}{q-1}$. The binary or ternary sequences are particularly interesting. Let us see the cases when the construction is new.

In the binary case, the sequences are of length $n = 2\frac{q^s-1}{q-1}$. When $s = 2$ we obtain a new construction of 1-almost-perfect sequences of length $2(q+1)$ for q a power of an odd prime which completes the construction of Langevin in [11] and of Pott and Bradley in [16]. If s is odd, n is not a multiple of 4 and according to [20], there is no almost-perfect sequence. In all the cases when $s \neq 2$ (and in particular when s is even), $(\frac{n}{2} - 1) = q^s + ... + q$ is not a power of a prime (because q is a power of a prime) and it is stated in [16] that for $\frac{n}{2} - 1 \leq 10000$ there is no almost-perfect sequences. In consequence, if s is odd or $s \neq 2$, we have constructed "almost-constant" sequences of length n for which there is no almost-perfect sequence.

In the ternary case, we found ternary $\{0, +1, -1\}$ almost-perfect sequences of length $2\frac{q^s-1}{q-1}$ for q a power of a prime and s a positive integer and we have no conditions for the parameters q and s (in [12] the conditions were $(s, \frac{q-1}{2}) = 1$ and $2|\frac{q-1}{2}$ in [12]). The next four tables summerize the parameters for each length. We specify by a star lengths for which no such sequences were known before.

q	3	3	3	3	3	5	5	5
s	2	3	4	5	6	2	3	4
n	8	26	80	242	728	12*	62*	312*

q	7	7	7	9	9	11	11	13	13
s	2	3	4	2	3	2	3	2	3
n	16	114*	800	20*	182*	24	266	28*	366*

q	17	17	19	19	23	23	25	25	27
s	2	3	2	3	2	3	2	3	2
n	36*	614*	40	722*	48	1106	52*	1302*	56

q	29	31	37	41	43	47	49	121	125
s	2	2	2	2	2	2	2	2	2
n	60*	64	76*	84*	88	96	100*	244*	252*

REFERENCES

[1] H.A. Barker. Design of multilevel pseudorandom signals for specified harmonic content. In *Proc. IEE*, volume 332, pages 556–561, 1991.

[2] F. Clarke. The discrete fourier transform of a recurrent sequence. *AAECC*, 8:485–489, 1997.

[3] R.A. Games. The geometry of quadrics and correlation of sequences. *IEEE TIT*, IT-32(3):423–426, may 1986.

[4] T. Hoholdt and J. Justesen. Ternary sequences with perfect periodic autocorrelation. *IEEE TIT*, IT-29(4):597–599, july 1983.

[5] V.P. Ipatov. Ternary sequences with ideal periodic autocorrelation properties. *Radio Eng. Electron. Phys.*, 24:75–79, Oct. 1979.

[6] D. Jungnickel. A theorem of ganley. *Graphs Combin.*, 3:141–143, 1987.

[7] D. Jungnickel and A. Pott. Perfect and almost-perfect sequences. to appear.

[8] S.L. Ma K.H. Leung, S. Ling and K.B. Tay. Almost perfect sequences with $\theta = 2$. *Archiv. Math.*, 70:128–131, 1998.

[9] S.L. Ma K.T. Arasu and N.J. Voss. On a class of almost perfect sequences. *Journal of algebra*, 192:641–650, 1997.

[10] P. Langevin. Almost perfect binary functions. *A.A.E.C.C.*, 4(2):99–102, 1993.

[11] P. Langevin. Construction of almost perfect functions. In *Journées du PRC*, pages 175–185, october 1993. Complexité, codage, compression, cryptographie.

[12] P. Langevin. Some sequences with good autocorrelation properties. In *Contemporary Mathematics*, volume 168, pages 213–216, 1994.

[13] R. Lidle and H. Niederreiter. *Finite fields*, volume 20. Encyclopedia of Mathematics and its Applications, 1983.

[14] J.P. Martin. *Codes et suites à racines multiples.* PhD thesis, Université de Toulon et du Var, 1994.

[15] J.M. Goethals P. Delsarte and J.J. Seidel. Orthogonal matrices with zero diagonal. *Canad. J. Math.*, 23:816–832, 1971.

[16] A. Pott and S.P. Bradley. Existence and nonexistence of almost-perfect autocorrelation sequences. *IEEE TIT*, 41(1):301–304, january 1995.

[17] M.B. Pursley and P.D. Sarwate. Crosscorrelation properties of pseudo-random and related sequences. *IEEE Proc.*, 68:593–619, May 1980.

[18] B. Schmidt. Cyclotomic integers and finite geometry. J. Amer. Math. Soc., to appear.

[19] R.J. Turyn. Character sums and difference sets. *Pacific Journal of Mathematics*, 15(1):319–347, 1965.

[20] J. Wolfmann. Almost perfect autocorrelation sequences. *IEEE Trans. Inform. Theory*, 38:1412–1418, 1992.

G.E.C.T., UNIVERSITÉ DE TOULON ET DU VAR, 83957 LA GARDE CEDEX, FRANCE

E-mail address: boursier@univ-tln.fr

On the Capacity of Distance Enhancing Constraints for High Density Magnetic Recording Channels

Emina Soljanin

Bell Laboratories – Lucent Technologies
Room 2C–177
600 Mountain Avenue
Murray Hill, NJ 07974, U.S.A.

Tel/Fax: +1-908-582-7933/3340
e-mail: emina@bell-labs.com

Adriaan J. van Wijngaarden

Bell Laboratories – Lucent Technologies
Room 2C–175
600 Mountain Avenue
Murray Hill, NJ 07974, U.S.A.

Tel/Fax: +1-908-582-3080/3340
e-mail: alw@bell-labs.com

Abstract—Modulation codes used in almost all contemporary storage products belong to the class of constrained codes. These nonlinear codes encode arbitrary input sequences as sequences that do not contain (globally or at certain positions) a finite number of finite length strings. Distance enhancing constraints of the second type should eliminate some of the possible recorded sequences in order to increase the minimum distance between those that remain. This general goal does not specify how the constraints should be defined, and, until recently, the only known constraints of this type were the matched-spectral-null (MSN) constraints. During the past few years, significant progress has been made in defining high-rate distance-enhancing constraints for high density magnetic recording channels. The most important characteristics of these constraints are that they support the design of high rate codes and that they simplify the Viterbi detectors relative to the uncoded channels. Codes that have been designed based on these constraints are the first distance-enhancing codes implemented in commercial magnetic recording systems. The main idea is to first identify constraints that differences of channel sequences (possible errors) should satisfy, and then infer the constraints that channel sequences should satisfy. The problem we address here is how to define the least restrictive of such constraints.

Keywords—constrained coding, intersymbol interference, magnetic recording.

I. Introduction

Modulation codes used in almost all contemporary storage products belong to the class of constrained codes. These nonlinear codes encode arbitrary input sequences as sequences that do not contain (globally or at certain positions) a finite number of finite length strings. Sets of such sequences are called constrained sets or constraints. Design of constrained codes begins with identifying constraints that achieve certain objectives. Once the system of constraints is specified, the task that remains is to construct two finite state machines: an *encoder* to convert arbitrary user sequences into constrained sequences and a *decoder* to recover user sequences from constrained sequences. Constrained sets and codes are described in Section II.

Two types of constraints are of interest in magnetic recording channels: constraints for improving timing and gain control and simplifying the design of the Viterbi detector for the channel, and constraints for improving noise immunity. Constraints of the first type impose run-length limitations (RLL) on sequences of recorded symbols. Several high rate codes for these constraints have been designed and implemented. Constraints of the second type should eliminate some of the possible recorded sequences in order to increase the minimum distance between those that remain. This general goal does not specify how the constraints should be defined, and, until recently, the only known constraints of this type were the matched-spectral-null (MSN) constraints. They describe sequences whose spectral nulls match those of the channel, and because of that, increase its minimum distance.

During the past few years, significant progress has been made in defining high-rate distance-enhancing constraints for high density magnetic recording channels (see [1] and references therein). The main idea, explained in Section III, is to first identify constraints that differences of channel sequences (possible errors) should satisfy, and then infer the constraints that channel sequences should satisfy. The most important characteristics of these constraints are that they support the design of high rate codes and that they simplify the Viterbi detectors relative to the uncoded channels. Codes designed based on these constraints were the first distance-enhancing codes implemented in commercial magnetic recording systems. The problem we address in Section IV is how to define the least restrictive constraints on channel sequences when constraints on their differences are specified.

II. Constrained Sets and Codes

Constrained codes are most often derived based on constrained systems of finite type (FT). An FT system X over alphabet $\mathcal{A}$ can always be characterized by a finite list of forbidden strings $\mathcal{F} = \{w_1, \ldots, w_N\}$ of symbols in $\mathcal{A}$. Defined this way, FT systems will be denoted by $X_{\mathcal{F}}^{\mathcal{A}}$. A set of constraints for the binary channel can be written in the form of a directed graph with a finite number of states and edges, and edge labels drawn from the binary alphabet. The set of corresponding constrained sequences is obtained by reading the labels of paths through the graph. A constrained code $\mathcal{C}$, $\mathcal{C} \subseteq X_{\mathcal{F}}^{\{0,1\}}$ can be constructed based on a graph representation of $X_{\mathcal{F}}^{\{0,1\}}$ by means of the *state splitting*

algorithm [2].

Translation of constrained sequences into the channel sequences depends on the modulation method. Saturation recording of binary information on magnetic medium is accomplished by converting an input stream of data into a spatial stream of bit cells along a track where each cell is fully magnetized in one of two possible directions, denoted by 0 and 1. There are two important modulation methods commonly used on magnetic recording channels: *non-return-to-zero* (NRZ) and *modified non-return-to-zero* (NRZI). In NRZ modulation, the binary digits 0 and 1 in the input data stream correspond to 0 and 1 directions of cell magnetizations, respectively. In NRZI modulation, the binary digit 1 corresponds to a magnetic transition between two bit cells, and the binary digit 0 corresponds to no transition. For example, the channel constraint which forbids transitions in two neighboring bit-cells, can be accomplished by either $\mathcal{F} = \{11\}$ NRZI constraint or $\mathcal{F} = \{101, 010\}$ NRZ constraint. The graph representation of these two constraints is shown in Fig. 1. The NRZI representation is in this case simpler.

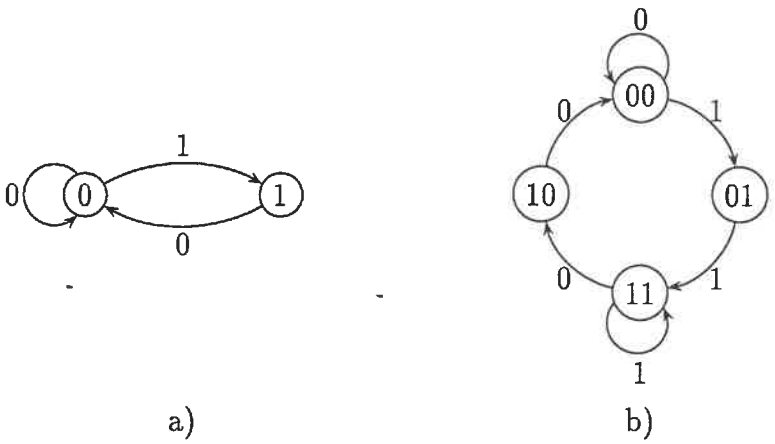


Fig. 1. Two equivalent constraints: a) $\mathcal{F} = \{11\}$ NRZI and b) $\mathcal{F} = \{101, 010\}$ NRZ.

The maximum rate of a code into a constrained system is determined by its *Shannon capacity*. The Shannon capacity or simply *capacity* of a constrained system, denoted by C, is defined as

$$C = \lim_{n \to \infty} \frac{\log_2 N(n)}{n},$$

where $N(n)$ is the number of sequences of length n. The capacity of an FT constrained system represented by a graph G can be easily computed from the *adjacency matrix* (or *state transition matrix*) of G. The adjacency matrix of graph G with r states and a_{ij} edges from state i to state j, $1 \leq i, j \leq r$, is the $r \times r$ matrix $A = A(G) = \{a_{ij}\}_{r \times r}$. The Shannon capacity of G is given by

$$C = \log_2 \lambda(A),$$

where $\lambda(A)$ is the largest real eigenvalue of A.

III. Constraints for ISI Channels

A. Requirements

It has been proposed in a number of recent papers that constrained codes be used to provide coding gain on channels with high intersymbol-interference (ISI). The main idea of this approach can be briefly described as follows [1]. The minimum distance of the uncoded binary channel with transfer function $h(D)$ is defined by

$$d^2_{\min} = \min_{\epsilon(D)\neq 0} \|h(D)\epsilon(D)\|^2,$$

where

$$\epsilon(D) = \sum_{i=0}^{l-1} \epsilon_i D^i,$$

and $\epsilon_i \in \{-1, 0, 1\}$, $\epsilon_0 = 1$, $\epsilon_{l-1} \neq 0$, is the polynomial corresponding to a normalized input error sequence $\epsilon = \{\epsilon_i\}_{i=0}^{l-1}$ of length l, and the squared norm of a polynomial is defined as the sum of its squared coefficients. The minimum distance is bounded from above by $\|h(D)\|^2$, denoted by

$$d^2_{\text{MFB}} = \|h(D)\|^2. \tag{1}$$

This bound is known as the *matched-filter bound* (MFB), and is achieved when the error sequence of length $l = 1$, *i.e.*, $\epsilon(D) = 1$, is in the set

$$\arg \min_{\epsilon(D)\neq 0} \|h(D)\epsilon(D)\|^2. \tag{2}$$

For channels that fail to achieve the MFB, *i.e.*, for which $d^2_{\min} < \|h(D)\|^2$, error sequences $\epsilon(D)$ for which

$$d^2_{\min} \leq \|h(D)\epsilon(D)\|^2 < \|h(D)\|^2 \tag{3}$$

are of length $l \geq 2$ and may belong to the strings excluded by a constrained system $X_{\mathcal{L}}^{\{-1,0,1\}}$, where $\mathcal{L}$ is an appropriately chosen finite list of forbidden strings.

For code $\mathcal{C}$, we define the set of all admissible non-zero error sequences,

$$\mathcal{E}(\mathcal{C}) = \{\epsilon = \{e_i\} | e_i \in \{-1, 0, 1\}, e_0 = 1, \epsilon = \boldsymbol{a} - \boldsymbol{b},\ \boldsymbol{a}, \boldsymbol{b} \in \mathcal{C}\}. \tag{4}$$

Given the condition $\mathcal{E}(\mathcal{C}) \subseteq X_{\mathcal{L}}^{\{-1,0,1\}}$, we seek to identify the least restrictive finite collection $\mathcal{F}$ of blocks over the alphabet $\{0, 1\}$ so that

$$\mathcal{C} \subseteq X_{\mathcal{F}}^{\{0,1\}} \Longrightarrow \mathcal{E}(\mathcal{C}) \subseteq X_{\mathcal{L}}^{\{-1,0,1\}}. \tag{5}$$

B. Definitions

A constrained code is defined by specifying $\mathcal{F}$, the list of forbidden strings for code sequences. Prior to that one needs to first characterize error sequences that satisfy (3) and

then specify $\mathcal{L}$, the list of forbidden strings for error sequences. Error event characterization can be done by using any of the methods described by Karabed, Siegel, and Soljanin in [1]. Specification of $\mathcal{L}$ is usually straightforward.

A natural way to construct a collection $\mathcal{F}$ of blocks forbidden in code sequences based on the collection $\mathcal{L}$ of blocks forbidden in error sequences is the following. From the above definition of error sequences $\epsilon = \{\epsilon_i\}$ we see that $\epsilon_i = 1$ requires $a_i = 1$ and $\epsilon_i = -1$ requires $a_i = 0$, *i.e.*, $a_i = (1 + \epsilon_i)/2$. error sequences. For each error block $\boldsymbol{w}_\mathcal{E} \in \mathcal{L}$, construct a list $\mathcal{F}_{\boldsymbol{w}_\mathcal{E}}$ of code blocks $\boldsymbol{w}_\mathcal{C}$ of the same length l according to the rule:

$$\mathcal{F}_{\boldsymbol{w}_\mathcal{E}} = \{\boldsymbol{w}_\mathcal{C} \in \{0,1\}^l \,|\, w^i_\mathcal{C} = (1 + w^i_\mathcal{E})/2 \text{ for all } i \text{ for which } w^i_\mathcal{E} \neq 0\}. \tag{6}$$

Then the collection $\mathcal{F}$ obtained as $\mathcal{F} = \bigcup_{\boldsymbol{w}_\mathcal{E} \in \mathcal{L}} \mathcal{F}_{\boldsymbol{w}_\mathcal{E}}$ satisfies requirement (5). However, the constrained system $X_\mathcal{F}^{\{0,1\}}$ obtained this way may not be the most efficient.

We illustrate the above ideas on the example of the E^{2}PR4 channel. Its transfer function is $h(D) = (1-D)(1+D)^3$, and its MFB, determined by $\epsilon(D) = 1$, is $\|(1-D)(1+D)^3 \cdot 1\|^2 = 10$. The error polynomial $\epsilon(D) = 1 - D + D^2$ is the unique error polynomial for which $\|(1-D)(1+D)^3\epsilon(D)\|^2 = 6$, and the error polynomials $\epsilon(D) = 1-D+D^2+D^5-D^6+D^7$ and $\epsilon(D) = \sum_{i=0}^{l-1}(-1)^i D^i$ for $l \geq 4$ are the only polynomials for which $\|(1 - D)(1 + D)^3\epsilon(D)\|^2 = 8$ (see for example [1]).

It is easy to show that these error events are not in the constrained error set defined by the list of forbidden error strings $\mathcal{L} = \{+-+00, +-+-\}$, where + denotes 1 and - denotes -1. To see that, note that an error sequence that does not contain string $+-+00$ cannot have polynomial $\epsilon(D) = 1 - D + D^2$ or $\epsilon(D) = 1 - D + D^2 + D^5 - D^6 + D^7$, while an error sequence that does not contain string $+-+-$ cannot have a polynomial of the form $\epsilon(D) = \sum_{i=0}^{l-1}(-1)^i D^i$ for $l \geq 4$. Therefore, by the above procedure of defining the list of forbidden code strings, we obtain the $\mathcal{F} = \{+-+\}$ NRZ constraint. Its capacity is about 0.81, and a rate 4/5 code into the constraint was first given in [3].

In [1], the following approach was used to obtain several higher rate constraints. For each of the error strings in list $\mathcal{L}$, we write all pairs of channel strings whose difference is the error string. To define the list $\mathcal{F}$, we look for the longest string(s) appearing in at least one of the strings in each channel pair. For the example above and $+-+00$ error string, a case-by-case analysis of channel pairs is depicted in Fig. 2. We can distinguish two types

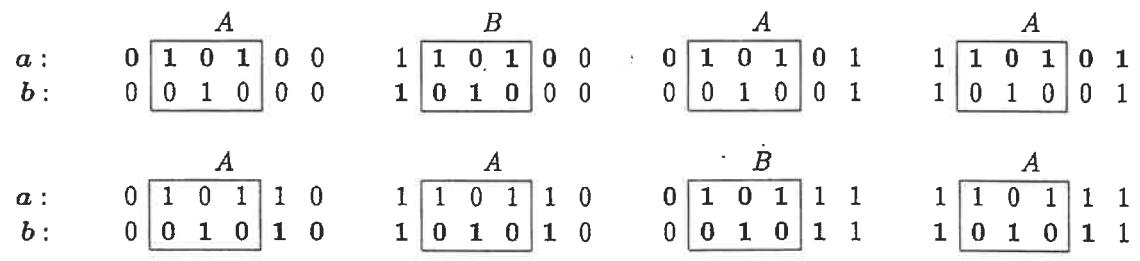


Fig. 2. Possible pairs of sequences for which error event $+ - +00$ may occur.

(denoted by A and B in the figure) of pairs of code sequences involved in forming an error

event. In a pair of type A, at least one of the sequences has a transition run of length 4. In a pair of type B, both sequences have transition runs of length 3, but for one of them the run starts at an even position and for the other at an odd position. This implies that an NRZI constrained system that limits the run of 1s to 3 when it starts at an odd position and to 2 when it starts at an even position, eliminates all possibilities shown bold-faced in Fig. 2. In addition, this constraint eliminates all error sequences containing string $+-+-$. A graph representation of this constraint is given in Fig. 3. Its capacity is

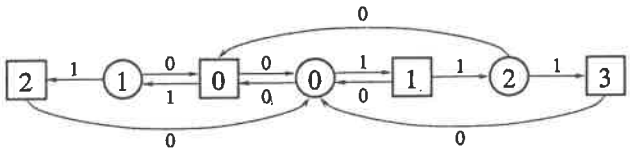


Fig. 3. Graph representation of $\mathcal{F} = \{1111, 111_{\text{even}}\}$ NRZI constraint.

about 0.916.

IV. Analysis of Constraints on Channel Sequences

In this section we will continue our analysis of the constraints that channel sequences should satisfy. The objective is to find the least restrictive constraints on channel sequences when constraints on their differences are specified. As in the previous section, we will concentrate on the E^2PR4 channel, for which the error polynomials have the following form

$$E_1.\ \epsilon(D) = \sum_{i=0}^{l-1} (-1)^i D^i, \text{where } l \geq 3$$

and

$$E_2.\ \epsilon(D) = 1 - D + D^2 + D^5 - D^6 + D^7$$

These error events can not occur if property (6) holds for every pair of codewords in the code set $\mathcal{C}$. Consequently, we can consider the entire set of binary channel sequences and determine for every channel sequence $\boldsymbol{a}$, whether or not there exists a channel sequence $\boldsymbol{b}$ for which the difference $\boldsymbol{a} - \boldsymbol{b}$ would create an error sequence of the form E_1 or E_2. Let $\mathcal{B}_a = \{\boldsymbol{b}_1, \cdots, \boldsymbol{b}_s\}$ denote the set of channel sequences for which the difference $\boldsymbol{a} - \boldsymbol{b}_i$, where $1 \leq i \leq s$, would give an inadmissible error sequence. We then have the option to include element $\boldsymbol{a}$ in the code set $\mathcal{C}$ and exclude the set $\mathcal{B}_a$, or to exclude $\boldsymbol{a}$ and to add one or more elements $\boldsymbol{b}_i$ of $\mathcal{B}_a$ to $\mathcal{C}$, provided that $\mathcal{B}_{b_i} \cap \mathcal{C} = \varnothing$.

It can be easily verified that for every channel sequence $\boldsymbol{a}$ that fulfils the $\mathcal{F} = \{101, 010\}$ NRZ constraint, or equivalently the $\mathcal{F} = \{11\}$ NRZI constraint, the set $\mathcal{B}_a = \varnothing$. Sequences that fulfil this constraint form the basis of the code set $\mathcal{C}$. The Shannon capacity for the given constraint is 0.6942.

We will now try to determine which of the sequences that do not satisfy the $\mathcal{F} = \{11\}$ NRZI constraints can be included in the code set $\mathcal{C}$.

The common characteristic of sequences that do not satisfy the $\mathcal{F} = \{11\}$ NRZI constraints is the presence of at least one subsequence with the property that this subsequence can be represented in NRZI as a run of k consecutive "ones", where $k \geq 2$. The corresponding subsequence is represented in NRZ as a run of alternating symbols of length $k_t = k + 1$. The notation is illustrated by the following example:

$$\begin{aligned} \text{NRZ}: \quad & 001\underline{1010}001\underline{101}10\underline{01010}0 \\ \text{NRZI}: \quad & *0101110010110101111110 \end{aligned}$$

The subsequences with $k_t \geq 3$ are underlined. We will use the symbol $*$ to represent an arbitrary element of alphabet $\mathcal{A}$.

A. Characterization of the Pairs that Form Error Event E_1

Consider a channel sequence $\boldsymbol{a}$ with a segment W for which $k_t = l_{\max}$. If $l_{\max} \geq 3$, there exists a pair $(\boldsymbol{a}, \boldsymbol{b})$, where $\boldsymbol{b}$ differs from $\boldsymbol{a}$ in a part of segment W only. In this way error events of the form E_1 can occur for any $l_{\min} \leq l \leq l_{\max}$, where $l_{\min} = 3$.

Let the subsequence preceding W have $k_t = r_p$ and the subsequence succeeding W have $k_t = r_s$. To characterize the sequence $\boldsymbol{b}_i$ that forms an error event E_1 we distinguish the following cases:

Case A: $l = l_{\max}$. By inverting W we obtain a sequence $\boldsymbol{b}_i$ with a subsequence for which $k_t = r_p + l_{\max} + r + s$. Since $r_p \geq 1$ and $r_s \geq 1$, $k_t \geq l_{\max} + 2$.

Case B: $l_{\min} \leq l < l_{\max}$. In this case only a segment of l consecutive symbols of W will be inverted. Let q denote the first position of the inverted consecutive symbols. Obviously, $1 \leq q \leq (l_{\max} - l + 1)$.

- If $q = 1$, we obtain $\boldsymbol{b}_i$ that has two subsequences, with $k_t = r_p + l$ and $k_t = l_{\max} - l$, respectively.
- If $1 < q < (l_{\max} - l + 1)$, we obtain $\boldsymbol{b}_i$ that has three subsequences with $k_t = q$, $k_t = l$, and $k_t = w - l - q$, respectively.
- If $q = (l_{\max} - l + 1)$, we obtain $\boldsymbol{b}_i$ that has two subsequences with $k_t = q$ and $k_t = l + r_s$, respectively.

As an example, consider the situation where $W = 10101$ and $k_t = l_{\max} = 5$.

$$\begin{array}{lll} *1101011* & *1101011* & *1101011* \\ \quad +-+ & \quad +-+ & \quad +-+ \\ *\underline{1010}011* & *11\underline{101}11* & *110\underline{0101}* \\ \\ *1101011* & *1101011* & *1101011* \\ \quad +-+- & \quad +-+- & \quad +-+-+ \\ *\underline{1010}111* & *11\underline{1010}1* & *\underline{1010101}* \end{array}$$

The number of different pairs $|\mathcal{B}_a|$ that can be formed if there are s subsequences for which $k_t \geq l_{\min}$, each having the value $k_t = l_{\max}$, and $1 \leq i \leq s$, is given by the following expression

$$|\mathcal{B}_a| = \sum_{i=1}^{s} \sum_{j=1}^{l_{\max}-l_{\min}+1} j = \sum_{i=1}^{s} \frac{(l_{\max} - l_{\min} + 1)(l_{\max} - l_{\min} + 2)}{2}$$

B. Characterization of the Pairs that Form Error Event E_2

Since the structure of the error event E_2 is quite different from the E_1 error events, we will have to analyze this case separately. In Table I we show the properties in terms of run lengths of the pair

$$(v_1 0\,1\,0 v_2 v_3 0\,1\,0 v_4, v_1 1\,0\,1 v_2 v_3 1\,0\,1 v_4)$$

for the 16 different values of $v_1 v_2 v_3 v_4$.

TABLE I

PAIRS OF SEQUENCES FOR WHICH ERROR EVENT $1 - D + D^2 + D^5 - D^6 + D^7$ MAY OCCUR

i	$v_1v_2v_3v_4$	$a_i = v_1 0\,1\,0 v_2 v_3 0\,1\,0 v_4$	$b_i = v_1 1\,0\,1 v_2 v_3 1\,0\,1 v_4$	characteristics a_i	characteristics b_i
0	0000	0010000100	0101001010	$k_t = 3$ (twice)	$k_t \geq 5$ (twice)
1	0001	0010000101	0101001011	$k_t \geq 4$	$k_t \geq 5$, $k_t = 4$
2	0010	0010010100	0101011010	$k_t = 5$	$k_t \geq 6$, $k_t \geq 4$
3	0011	0010010101	0101011011	$k_t \geq 6$	$k_t \geq 6$
4	0100	0010100100	0101101010	$k_t = 5$	$k_t \geq 4$, $k_t \geq 6$
5	0101	0010100101	0101101011	$k_t = 5$, $k_t \geq 4$	$k_t \geq 4$, $k_t = 5$
6	0110	0010110100	0101111010	$k_t = 4$ (twice)	$k_t \geq 4$ (twice)
7	0111	0010110101	0101111011	$k_t = 4$, $k_t \geq 5$	$k_t \geq 4$
8	1000	1010000100	1101001010	$k_t \geq 4$	$k_t = 4$, $k_t \geq 5$
9	1001	1010000101	1101001011	$k_t \geq 4$ (twice)	$k_t = 4$ (twice)
10	1010	1010010100	1101011010	$k_t \geq 4$, $k_t = 5$	$k_t = 5$, $k_t \geq 4$
11	1011	1010010101	1101011011	$k_t \geq 4$, $k_t \geq 6$	$k_t = 5$
12	1100	1010100100	1101101010	$k_t \geq 6$	$k_t \geq 6$
13	1101	1010100101	1101101011	$k_t \geq 6$, $k_t \geq 4$	$k_t = 5$
14	1110	1010110100	1101111010	$k_t \geq 5$, $k_t = 4$	$k_t \geq 4$
15	1111	1010110101	1101111011	$k_t \geq 5$ (twice)	$k_t = 3$ (twice)

There are, as shown in Table I, mostly pairs of which at least one of the elements has a double occurrence of $k_t \geq 4$. Only (a_3, b_3) and (a_{12}, b_{12}) contain a single $k_t \geq 6$.

C. Sequence Selection

We partition the set of binary sequences in subsets, each of which is characterized by the maximum value that k_t attains.

$k_t < 3$: As discussed previously, these sequences cannot be involved in forming an error event. The selection of these sequences does not put any restrictions on the choice of the remaining sequences.

$k_t = 3$: If a sequence with $k_t = 3$ is involved in an error event, then the other sequence in the pair has $k_t \geq 5$:

$$\begin{array}{c} *11011* \\ \quad +-+ \\ *10101* \end{array}$$

Consider a sequence $\boldsymbol{a}$ where $k_t = 3$ occurs s times. There are for every sequence s pairs that would give the error sequence $+-+$, and therefore s sequences have to be excluded. The other element of the pair has exactly one

occurrence of $k_t \geq 5$. We will include all sequences $\boldsymbol{a}$ with $k_t \leq 3$, motivated by the fact that $|\mathcal{B}_a| < |\mathcal{B}_b|$, for all $\boldsymbol{b} \in \mathcal{B}_a$.

$k_t = 4$: Whenever $k_t = 4$, there are two possible ways to form the error event $+-+$ and one way to form the error event $+-+-$. This is illustrated for the sequence $*110100*$.

$$\begin{array}{lll} *110100* & *110100* & *110100* \\ \quad +-+ & \quad +-+ & \quad +-+- \\ *\underline{101}000* & *11\underline{1010}* & *\underline{101010}* \end{array}$$

The other sequence in forming the error event either has $k_t \geq 4$ or $k_t \geq 6$. The run at which the sequence with $k_t = 4$ starts is either one position ahead or behind.

Consider a channel sequence where $k_t \leq 4$ and where the subsequence 110100 or 001011 occurs s times in the sequence at the positions $p_1, p_2, \cdots, p_s$. We partition this set in a set of even positions, $\mathcal{P}_e$, and a set of odd positions, $\mathcal{P}_o$. If we allow all sequences for which $|\mathcal{P}_e|$ is an even number, we have to exclude all the sequences where $|\mathcal{P}_e|$ is odd, unless every segment for which $k_t = 4$ is preceded and succeeded by a segment with $k_t > 1$.

Some of the sequences for which $|\mathcal{P}_e|$ is an even number have to be excluded to avoid the error event E_2, in particular pair $(\boldsymbol{a}_6, \boldsymbol{b}_6)$ and pair $(\boldsymbol{a}_9, \boldsymbol{b}_9)$ as in Table I. We therefore exclude sequences with the property that $k_t = 4$ occurs consecutively at even positions, surrounded by two segments for which $k_t = 1$, or separated by two segments with $k_t = 1$.

$k_t = 5$: Consider the situation where $k_t = 5$. The pairs of sequences that would give a forbidden error event are, as previously listed, equal to

$$\begin{array}{lll} *1101011* & *1101011* & *1101011* \\ \quad +-+ & \quad +-+ & \quad +-+ \\ *\underline{1010}011* & *11\underline{101}11* & *110\underline{0101}* \\ \\ *1101011* & *1101011* & *1101011* \\ \quad +-+- & \quad +-+- & \quad +-+-+ \\ *\underline{1010}111* & *11\underline{1010}1* & *\underline{1010101}* \end{array}$$

Based on this analysis, it is obvious that sequences with single occurrences of $k_t = 5$ have to be excluded. Sequences where the number of occurrences of segments with $k_t = 5$ occur an equal number of times, and for which $|\mathcal{P}_e|$ is an even number, can be included, provided that

- they do not contain a segment with $k_t = 4$ starting at an even position.
- they do not contain a segment with $k_t = 5$ that is preceded or succeeded by a segment with $k_t = 2$ or $k_t = 4$.

Channel sequences for which $k_t \geq 6$ occur once, will usually have to be excluded, because they can almost always be paired with channel sequences for which $k_t = 3$ or $k_t = 4$. The fraction of channel sequences with multiple occurrences of $k_t \geq 6$ is very small, and thus, the effect of including these sequences is minor.

Ideally, one would like to represent the rules for the inclusion of sequences in the code set $\mathcal{C}$ as described above by a graph. While this is relatively easy for the rules that limit the

maximum value of k_t, it is very hard to represent the rules for the inclusion of sequences with multiple occurrences of $k_t \geq 4$.

To compare the number of sequences that are specified with the number of sequences that are allowed in the constrained system represented in Fig. 3, we will determine the number of sequences that satisfy the rules for a given length n. The results are given in Table II.

TABLE II

NUMBER OF SEQUENCES OBTAINED USING THE DERIVED RULES OR THE GRAPH

n	$\lvert\mathcal{C}\rvert_{\text{rules}}$	$\lvert\mathcal{C}\rvert_{\text{graph}}$	Δ
5	28	28	0
6	56	50	6
7	104	100	4
8	200	178	22
9	372	356	16
10	720	634	86
11	1342	1268	74
12	2598	2258	340
13	4840	4516	324
14	9370	8042	1328
15	17494	16084	1410

The constrained system represented in Fig. 3 includes all sequences where $k_t = 3$ and all sequences with $k_t = 4$ starting at odd positions. The difference in code size, Δ, indicates that by allowing multiple occurrences of $k_t \geq 4$ one can enlarge the code set. Options to further extend the code set are currently being investigated.

REFERENCES

[1] R. Karabed, P. H. Siegel, and E. Soljanin "Constrained coding for binary channels with high intersymbol interference," *IEEE Trans. Inform. Theory*, to appear.

[2] R. Adler, D. Coppersmith, and M. Hassner, "Algorithms for sliding-block codes," *IEEE Trans. Inform. Theory*, vol. 29, no. 1, pp. 5–22, Jan. 1983.

[3] R. Karabed and P. H. Siegel, "Coding for Higher Order Partial Response Channels," *Proc. 1995 SPIE Int. Symp. on Voice, Video, and Data Communications*, Philadelphia, PA, Oct. 1995, vol. 2605, pp. 115–126.

[4] T. Cover, "Enumerative source encoding" *IEEE Trans. Inform. Theory*, pp. 73–77 Jan. 1973.

[5] A. J. van Wijngaarden and K. A. Schouhamer Immink "Combinatorial construction of high rate runlength-limited codes," *Proc. 1996 IEEE Global Telecommun. Conf. (GLOBECOM '96)*, London, U.K., Nov. 1996, pp. 343–347.

Algebraic Construction of Good Collision Resistant Signal Sets *

M. Greferath† and E. Viterbo‡

November 25, 1998

Abstract

We consider codes of rank 1 over $\mathbb{Z}_{2^m}$ in order to construct signal space codes for the collision channel. To achieve collision resistance we search among codes generated by vectors entirely consisting of units of the underlying ring. It turns out that the resulting signal sets improve over previously known collision resistant signal sets. A further advantage is that they allow for efficient decoding.

Key Words: Codes over Integer Residue Rings, Weight-Functions, Wireless Multiple Access Collision Channel, Fading.

1 Introduction

A novel class of signal space codes called Collision Resistant Modulations (CRM) for bandwidth efficient transmission on a random access collision channel, as in packet-radio applications, was proposed in [3]. There it is shown that CRMs achieve a significant gain over traditional modulation formats under a Slotted ALOHA protocol. We briefly summarize some of the key ideas presented in [3] in order to set the basis of the present paper.

We model the collision channel with AWGN and fading as the on-off vector channel

$$\mathbf{y} = C(G\mathbf{x} + \mathbf{n}) \tag{1}$$

Here $\mathbf{x} = (x_1, \ldots, x_n)$ is a transmitted word taken from the signal set S (signal-space code), $\mathbf{n} = (n_1, \ldots, n_n)$ is additive white Gaussian noise with i.i.d. components $n_i = \mathcal{N}(0, N_0/2)$ and the matrix $G = \mathrm{diag}(g_1, \ldots, g_n)$ represents the channel fading coefficients $g_i \geq 0$. The matrix $C = \mathrm{diag}(c_1, \ldots, c_n)$ represents the collision pattern with $c_i \in \{0, 1\}$ for all $i = 1, \ldots, n$, where $c_i = 0$ indicates the presence of a collision on the i-th signal component. Finally $\mathbf{y} = (y_1, \ldots, y_n)$ is the received signal vector.

The multiple access scheme induces a certain collision statistics. For example, if the users randomly access any n slots of a common channel to transmit the n components of a signal, the c_i may be considered as Bernoulli i.i.d. random variables with $P_c = P(c_i = 0)$ and if C has k collisions

$$P(C) = P_c^k (1 - P_c)^{n-k} \tag{2}$$

*File: crmxwcc99.tex

†AT&T Labs Research, 180 Park Av., Florham Park, NJ 07932, USA. E-mail: greferath@research.att.com

‡Dip. Elettronica, Politecnico di Torino, C. Duca degli Abruzzi 24, 10129 Torino, Italy. E-mail: viterbo@polito.it

The fading coefficients g_i are random variables whose joint statistics depends on the physical characteristics of the propagation channel, such as the multi-path delay profile and the Doppler spectrum.

We assume that the receiver has perfect channel state information (CSI) on both the collisions and the fading, i.e., has a complete knowledge of C and G. In this case, the Maximum Likelihood (ML) detection is obtained by minimizing over all $\mathbf{x} \in S$ the metric

$$d_C^2(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} c_i (y_i - g_i x_i)^2 \tag{3}$$

This detection criterium corresponds to the ML detection for the signal set S_C in an $(n-k)$-dimensional Euclidean space, where S_C is the projection of S on the subspace generated by the $n-k$ axes corresponding to the non zero c_i (k is the number of collisions in the pattern C). According to this detection criterium, in order to avoid systematic errors in the presence of $k < n$ collisions, we require that the points in S_C are distinct, for all C with Hamming distance $w_H(C) > 0$. Then, we have the following

Definition 1.1 A *Collision Resistant Modulation* (CRM) is a signal-space code S with the property that any projection on any coordinate subspace has the same number of distinct points, i.e., $M = |S_C| = |S|$ for all non-zero collision patterns C.

Equivalently, the words in S must have all distinct coordinates or, more precisely, the Hamming distance between any pair of words in S must be n. A similar requirement is imposed in the design of *high diversity* signal constellations for the fading channel where the number of distinct components is called *modulation diversity* [1, 2].

The symbol error probability is given by

$$P(e) = \sum_{C} P(C) P(e|C) \tag{4}$$

where C runs over all 2^n collision patterns, $P(C)$ is given in (2) and by using the Union bound

$$P(e|C) \leq \frac{1}{M} \sum_{\mathbf{x} \in S_C} \sum_{\mathbf{x} \neq \widehat{\mathbf{x}}} P(\mathbf{x} \to \widehat{\mathbf{x}}) \tag{5}$$

where $P(\mathbf{x} \to \widehat{\mathbf{x}})$ is the pairwise error probability averaged with respect to G. $\{\mathbf{x} \to \widehat{\mathbf{x}}\}$ represents the event that the decoder chooses $\widehat{\mathbf{x}}$ when $\mathbf{x}$ was actually transmitted, as if $\mathbf{x}$ and $\widehat{\mathbf{x}}$ were the two only possible decoder outcomes. We refer to this as *pairwise error event.*

Expressions for the pairwise error probability for both the Gaussian and independent Rayleigh fading channels can be found in [3]. These show that in order to reduce the symbol error probability it is important to maximize different types of minimum distances of all the constellations S_C according to the type of channel being used.

Given any two signal words $\mathbf{s}$ and $\mathbf{t}$ in S we consider three types of distances:

- $d_E^2(\mathbf{s}, \mathbf{t}) := \|\mathbf{s} - \mathbf{t}\|_2^2 = \sum_{i=1}^{n} |s_i - t_i|^2$, the Squared Euclidean distance (SED). This is appropriate on the pure AWGN channel without fading.
- $d_P(\mathbf{s}, \mathbf{t}) := \prod_{i=1}^{n} |s_i - t_i|$, the Product distance. This is appropriate in the presence of independent Rayleigh fading.
- $d_L(\mathbf{s}, \mathbf{t}) := \|\mathbf{s} - \mathbf{t}\|_1 = \sum_{i=1}^{n} |s_i - t_i|$, the ℓ_1-distance. This is appropriate in the presence of Rice fading.

Note that the product distance is not a distance function in the narrow sense in that is does not satisfy the triangle inequality in general. In the following we speak about the weights w_E, w_P and w_L of the symbols in S as their distance from the origin.

In [3] some low dimensional CRMs are shown which could only be decoded by exhaustive search through the codebook. Here we are interested in higher dimensional CRMs which provide greater minimum distances in the projected constellations S_C and present a structure that can be effectively exploited for decoding.

2 The algebraic construction

Let $\mathbb{Z}_M$ denote the ring of integers modulo M where $M = 2^m$, and write $\mathbb{Z}_M = \{-M/2 + 1, \ldots M/2\}$ for convenience. Note that the usual mod M operation has to be slightly modified in order to match the symmetric representation of $\mathbb{Z}_M$.

We consider linear codes of rank 1 over $\mathbb{Z}_M$ and first note that the generator matrix of such a code is a single word $\mathbf{v} = (v_1, \ldots v_n)$ that contains at least one unit of $\mathbb{Z}_M$. The $\mathbb{Z}_M$-code generated by it is given by $\mathcal{C} = \{\beta\mathbf{v} | \beta \in \mathbb{Z}_M\}$. As corresponding signal set S we take the code words identically embedded as real-valued vectors into the Euclidean space $\mathbb{R}^n$. Under the above premise this embedding is a weight preserving mapping of $(\mathbb{Z}_M, w_{\text{Lee}})$ into $(\mathbb{R}, \ell_1)$.

In order to obtain the collision resistance property we require that the generating word entirely consists of units of $\mathbb{Z}_M$. In fact the i-th component βv_i of each codeword are all distinct as β runs over $\mathbb{Z}_M$. For $M = 2^m$ the unit group $U(\mathbb{Z}_M)$ is the set of odd numbers in $\mathbb{Z}_M$.

Proposition 2.1 *The minimum weight of a vector in S is equal to the minimum distance between any two vectors of S for all three types of distances.*

Proof: In order to see this denote by w one of the respective weight functions w_L, w_E or w_P on $\mathbb{Z}^n$ and denote by d the corresponding distance function. Then we obviously have

$$w_{\min}(S) \geq d_{\min}(S) \geq d_{\min}(\Lambda) = w_{\min}(\Lambda)$$

where $\Lambda := \mathcal{C} + M\mathbb{Z}^n$ is the lattice resulting from $\mathcal{C}$ by *Construction A* (cf. [5]). Let $\mathbf{x} := (x_1, \ldots, x_n) \in \mathbb{Z}^n$ be given such that $w(\mathbf{x}) = w_{\min}(\Lambda)$. If $\mathbf{x} \notin S$ then exchanging x_1 by $x_1' := x_1 \pmod M$ produces a vector $\mathbf{x}' \in \Lambda$ with $w(\mathbf{x}') \leq w(\mathbf{x})$ and hence $w(\mathbf{x}') = w(\mathbf{x})$ because of minimality. Without loss of generality we may therefore assume $\mathbf{x} \in S$, and hence conclude $w_{\min}(\Lambda) \geq w_{\min}(S)$. All in all we have shown

$$d_{\min}(S) = w_{\min}(S)$$

which proves our claim. □

We finally note that equivalent signal sets S in terms of error performance are obtained by taking permutations and sign-changed versions of the generator vector of $\mathcal{C}$.

2.1 Signal sets in dimension $M/4$

In this section we show that the signal sets generated by the above codes can be easily characterized for all the three metrics when $n = M/4 = 2^{m-2}$. In the following the double factorial $(2\alpha + 1)!!$ denotes the product $(2\alpha + 1)(2\alpha - 1) \cdots 3 \cdot 1$.

Proposition 2.2 *Let $\mathbf{v} = (1, 3, \ldots, M/2 - 1)$ then for the $[M/4, 1]$ code $\mathcal{C}$ generated by $\mathbf{v}$ the following holds:*

(a) *the minimum Lee weight of* $\mathcal{C}$ *is* $M^2/16$,

(b) *the minimum Euclidean weight of* $\mathcal{C}$ *is* $M^3/64$, *and*

(c) *the minimum product weight of* $\mathcal{C}$ *is* $(M/2-1)!!$.

Proof: Let $\beta\mathbf{v}$ be a word in $\mathcal{C}$ and let $\beta = 2^k u$ for appropriate $k \leq m$ and $u \in U(\mathbb{Z}_M)$. Since multiplication by u^{-1} merely applies a permutation and sign changes in $\beta\mathbf{v}$, we may without loss of generality suppose $u = 1$. For our claim concerning the Lee weight we then compute

$$\begin{aligned} \|\beta\mathbf{v}\|_L &= \sum_{i=1}^{n} |2^k(2i-1)| \\ &= 2^{2k} \sum_{i=1}^{2^{m-1-k}} (2i-1) \\ &= 2^{2k}(2^{m-2-k}(2^{m-2-k}+1) - 2^{m-2-k}) \\ &= 2^{2m-4} = \frac{M^2}{16} \end{aligned}$$

as claimed. Note that $\mathcal{C} \setminus \{\mathbf{0}\}$ is a constant Lee weight code. For the Euclidean weight we similarly have

$$\begin{aligned} \|\beta\mathbf{v}\|_E &= 2^k \sum_{i=1}^{\frac{n}{2^k}} 2^{2k}(2i-1)^2 \\ &= \frac{4}{3}n^3 - \frac{4^k}{3}n = \frac{4n}{3}(n^2 - 4^{k-1}) \end{aligned}$$

which obviously assumes its minimum for $k = m-2$. Hence, in this case the resulting minimum weight is given by $8^m/64 = M^3/64$. For the product distance we have

$$\|\beta\mathbf{v}\|_P = \Big(\prod_{i=1}^{\frac{n}{2^k}} 2^k(2i-1)\Big)^{2^k}.$$

If we prove that this is an increasing function of k then the minimum product weight occurs at the unital multiples of $\mathbf{v}$ and hence on $\mathbf{v}$ itself, where it is given by $(M/2-1)!!$. Defining

$$g(k) := \Big(\prod_{i=1}^{\frac{n}{2^k}} 2^k(2i-1)\Big)^{2^k}$$

we have to show that $g(k+1)/g(k) \geq 1$ for all $0 \leq k \leq m-3$. An easy transformation shows that this is equivalent to

$$\frac{\prod_{i=1}^{\frac{n}{2^k}}(2i-1)}{\prod_{i=1}^{\frac{n}{2^{k+1}}}(2i-1)^2} \leq 2^{\frac{n}{2^k}}$$

for all $0 \leq k \leq m-3$. Substituting $s := n/2^k$ and filling up the double factorials the latter holds if and only if

$$\frac{\binom{4s}{2s}}{\binom{2s}{s}} \leq 2^{2s}$$

for all $s = 1, 2, 4, \ldots, 2^{m-k}$. This, however, it is easily verified because of

$$\binom{4s}{2s} = \sum_{i=0}^{2s} \binom{2s}{2s-i}\binom{2s}{i} \leq \binom{2s}{s}\sum_{i=0}^{2s}\binom{2s}{i} = \binom{2s}{s}2^{2s}.$$

□

In Table 1 some minimum distances of the above codes are reported.

M	$d_{L,min}$	$d^2_{E,min}$	$d^{1/n}_{P,min}$
16	16	64	3.201
32	64	512	18.255
64	256	4096	759.473
128	1024	32768	$1.963 \cdot 10^{6}$
256	4096	262144	$2.524 \cdot 10^{13}$
512	16384	2097152	$1.232 \cdot 10^{28}$
1024	65536	16777216	$1.797 \cdot 10^{58}$

Table 1: Codes for $n = M/4$ generated by $\mathbf{v} = (1, 3, \ldots, M/2 - 1)$

2.2 Punctured codes

In order to improve the spectral efficiency of the previous signal sets we should consider some punctured versions of the codes presented in the previous section. The puncturing operation produces lower dimensional signal sets with the same number of points of the original signal set, due to the collision resistance property. We will consider the case $n = m$, which corresponds to 1 bit/dim spectral efficiency.

Tables 2, 3 and 4 show some optimal codes giving the greatest minimum distances. These were found by an exhaustive search through all $\mathbf{v} = (1, v_2, \ldots v_n)$ and $v_i \in U(\mathbb{Z}_M)$ all distinct since they are obtained by puncturing the codes in the previous section. We note that any vector $\beta\mathbf{v}$ with $\beta \in U(\mathbb{Z}_M)$ generates the same code. Then, without loss of generality, we can always assume the first component of $\mathbf{v}$ to be equal to 1. Due to the rapidly increasing number of generator vectors to be tested the search for these optimal codes could only be performed up to dimension $n = 7$.

M	$d_{L,min}$	$\mathbf{v}$
16	16	[1,3,5,7]
32	30	[1,3,5,11,15]
64	68	[1,3,5,17,23,31]
128	159	[1,5,17,25,27,57,63]

Table 2: Optimal punctured codes for the Lee metric $n = m$

M	$d^2_{E,min}$	$\mathbf{v}$
16	64	[1,3,5,7]
32	253	[1,3,5,7,15]
64	1222	[1,3,5,7,17,31]
128	5488	[1,3,5,23,33,49,63]

Table 3: Optimal punctured codes for the Euclidean metric $n = m$

We next consider a construction which we conjecture to produce comparatively good signal sets. These are constructed from the generating vectors $\mathbf{v} = (1, |\alpha|, |\alpha^2| \ldots |\alpha^{n-1}|)$, where α is

M	$d_{P,min}^{1/n}$	v
16	3.201	[1,3,5,7]
32	4.638	[1,3,5,11,15]
64	8.363	[1,7,13,17,23,31]
128	15.554	[1,15,17,19,27,39,63]

Table 4: Optimal punctured codes for the Product metric $n = m$

selected among the elements of maximal order of $U(\mathbb{Z}_M)$.

Remark 2.3 Referring to [6, Chap. 4], we recall that the group of units $U(\mathbb{Z}_M)$ consists of all odd numbers in $\mathbb{Z}_M$ and is isomorphic to $C_2 \times C_{2^{m-2}}$ for all $m \geq 2$, whereas for the finite field $\mathbb{Z}_2$ it is clearly the trivial one-element group. For $M \geq 16$ the elements of maximal (multiplicative) order $M/4$ in $\mathbb{Z}_M$ are given by $(8k \pm 3) \pmod M$. To see this it is enough to show that $z^{M/8} \neq 1 \pmod M$ for all z of that form. It is clear that $(8k \pm 3)^{M/8} = 1 \pmod 4$. An induction over m, where $M = 2^m$, first proves the claim to be true for $m = 3$. In general we have $(8k \pm 3)^{2^{m+1-3}} - 1 = ((8k \pm 3)^{2^{m-3}})^2 - 1 = ((8k \pm 3)^{2^{m-3}} + 1)((8k \pm 3)^{2^{m-3}} - 1)$. If the latter were equal to 0 modulo M, then by induction hypothesis $(8k \pm 3)^{2^{m-3}} + 1 = 0 \pmod 4$, a contradiction. A similar argument shows that the elements of type $8k \pm 1$ have lower order.

Tables 5, 6 and 7 show the codes generated by some of these elements giving the best minimum distance. The generator vector is given only for the first α in the list. In order to show the interest in optimizing the choice of α, we also report in parentheses the distance obtained by the worst choice of α. It is an unsolved problem to relate the minimum weight of these codes to their generator vector and the choice of α.

M	$d_{L,min}$	α	v
16	16	3,5	[1,3,7,5]
32	29	3,5,11,13	[1,3,9,5,15]
64	(60) 66	19,27	[1,19,23,11,17,3]
128	(107) 145	11,19,27,35	[1,11,7,51,49,27,41]
256	(256) 324	35,117	[1,35,55,123,47,109,25,107]
512	(489) 679	109,155	[1,109,105,181,239,61,7,251,223]
1024	(932) 1576	293,339	[1,293,167,221,241,43,311,13,287,123]

Table 5: Good punctured codes for the Lee metric $n = m$

3 Comparison with rotated hypercube

In order to better understand the improvements given by the new signal sets we will compare, as an example, the 4-dimensional rotated hypercube signal set [2, 3] with the $\mathbb{Z}_{16}$ signal set generated by $\mathbf{v} = [1, 3, 5, 7]$. Tables 8,9,10 and 11 show the Euclidean and the product distances of all the projections S_C of both signal sets. Figures 1 and 2 show the two dimensional projections on the six coordinate planes of both signal sets. Finally, Figures 3 and 4 show some approximate curves of the symbol error probabilities $P(e)$ (solid lines) and some error

M	$d^2_{E,min}$	α	$\mathbf{v}$
16	64	3,5	[1,3,7,5]
32	(237) 253	5,13	[1,5,7,3,15]
64	(942) 1118	11,29	[1,3,9,11,17,3]
128	(3023) 4444	3,43	[1,3,9,27,47,13,39]
256	(12392) 18600	5,51	[1,5,25,125,113,53,9,45]
512	(39657) 85897	45,91	[1,45,23,11,17,253,121,187,223]
1024	(159106) 354370	293,339	[1,293,167,221,241,43,311,13,287,123]

Table 6: Good punctured codes for the Euclidean metric $n = m$

M	$d^{1/n}_{P,min}$	α	$\mathbf{v}$
16	3.201	3,5	[1,3,7,5]
32	(4.237) 4.309	3,11	[1,3,9,5,15]
64	(6.289) 7.911	19,27	[1,19,23,11,17,3]
128	(8.287) 14.581	19,27	[1,19,23,53,17,61,7]
256	(14.987) 25.545	35,117	[1,35,55,123,47,109,25,107]
512	(22.383) 49.926	35,117	[1,35,201,133,47,109,231,107,161]
1024	(46.491) 95.459	251,461	[1,251,487,381,399,203,247,467,481,101]

Table 7: Good punctured codes for the Product metric $n = m$

probabilities conditioned on the number of collisions k (dashed curves from left to right $k = 0, 1, 2, 3$). In particular, we write

$$P(e) \leq \sum_{k=0}^{4} \binom{n}{k} P_c^k (1 - P_c)^{n-k} P_{\max}(e|k) \tag{6}$$

where $P_{\max}(e|k)$ is the largest error probability for given a collision pattern with k collisions. In the case of an AWGN channel we simply approximated $P_{\max}(e|k)$ with $\frac{1}{2}\text{erfc}(d_{k,\min}/(2\sqrt{N_0}))$ where $d_{k,\min}$ is the smallest entry in the k-th row of Tables 8 and 9. Comparing Figures 3 and 4 we observe an improvement in the symbol error probability in the range 10–35dB of E_b/N_0 for the $\mathbb{Z}_{16}$ signal set.

k	C with k collisions					
0	86					
1	32.88	32.88	32.88	32.88		
2	12.59	9.08	4.41	4.41	9.08	12.59
3	0.06	0.06	0.06	0.06		

Table 8: Minimum SED of all the projections S_C of the rotated 4D-hypercube signal set arranged according to the number of collisions k.

k	C with k collisions					
0	64					
1	35	35	35	35		
2	10	10	8	8	10	10
3	1	1	1	1		

Table 9: Minimum squared Euclidean distances of all the projections S_C of the $\mathbb{Z}_{16}$ signal set arranged according to the number of collisions k.

k	C with k collisions					
0	184.9					
1	12.48	12.48	12.48	12.48		
2	2.33	1.98	2.21	2.21	1.98	2.33
3	0.24	0.24	0.24	0.24		

Table 10: Minimum product distances of all the projections S_C of the rotated 4D-hypercube signal set arranged according to the number of collisions k.

4 The Decoder

We show here how to perform efficient ML detection in the presence of collisions for the signal sets we have constructed. Let us consider the lattice $\Lambda = \mathcal{C} + M\mathbb{Z}_M^n$ obtained by applying Construction A to a $[n, 1]$ code over $\mathbb{Z}_M$ generated by the vector $\mathbf{v} = (1, v_2, \ldots v_n)$

$$\Lambda = \mathcal{C} + M\mathbb{Z}_M^n$$

This lattice possesses a generator matrix of the form

$$G = \begin{pmatrix} 1 & v_2 & \cdots & v_n \\ 0 & M & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & M \end{pmatrix}$$

Given the collision pattern C with k collisions, the resulting projected lattice Λ_C has a generator matrix G_C which is obtained from G by removing the k rows and the columns corresponding to the collided components.

We conclude that all the signal sets S_C are lattice constellations hence we are able to decode them efficiently by applying the Universal Lattice decoder proposed in [7] and [8]. The search radius for the decoder can be adjusted according to the number of collisions in order to increase the decoding speed.

k	C with k collisions					
0	105					
1	15	15	15	15		
2	3	3	4	4	3	3
3	1	1	1	1		

Table 11: Minimum product distances of all the projections S_C of the $\mathbb{Z}_{16}$ signal set arranged according to the number of collisions k.

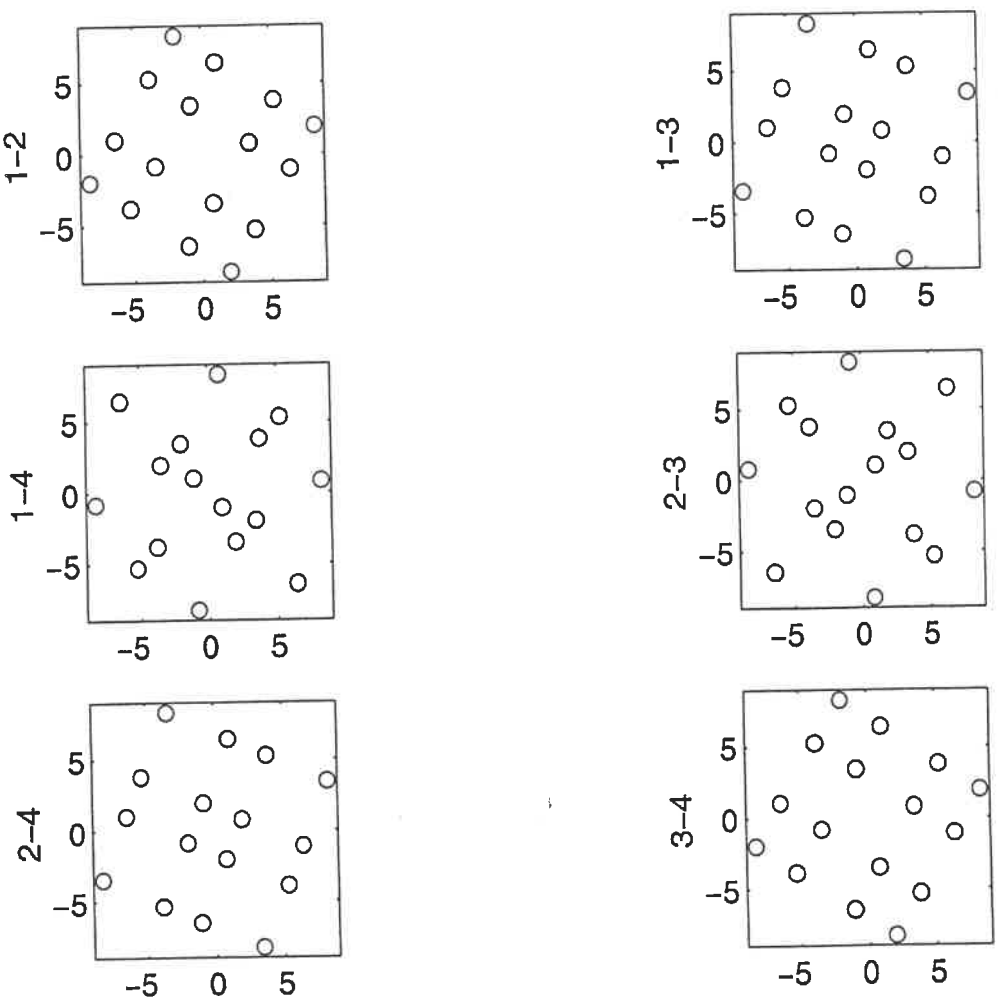


Figure 1: The 6 two-dimensional projections of the rotated 4D-hypercube signal set.

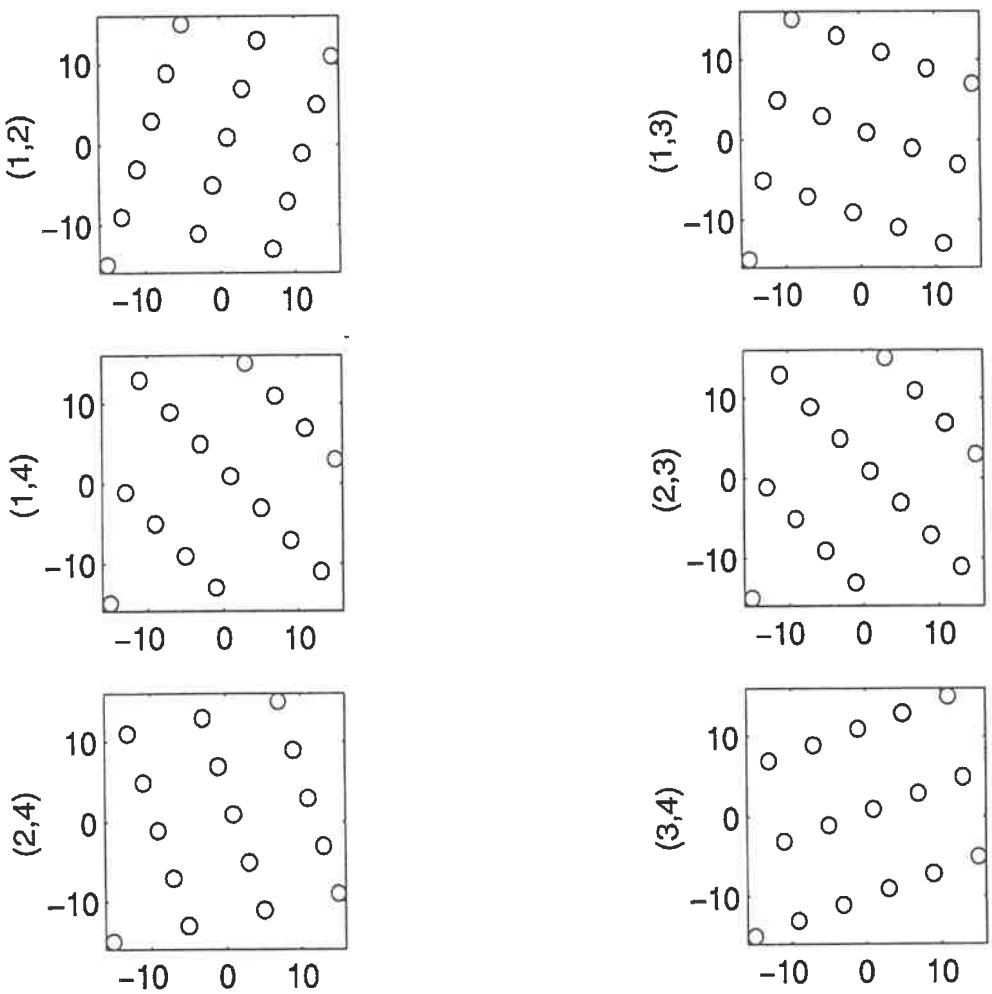


Figure 2: The 6 two-dimensional projections of the $\mathbb{Z}_{16}$ signal set.

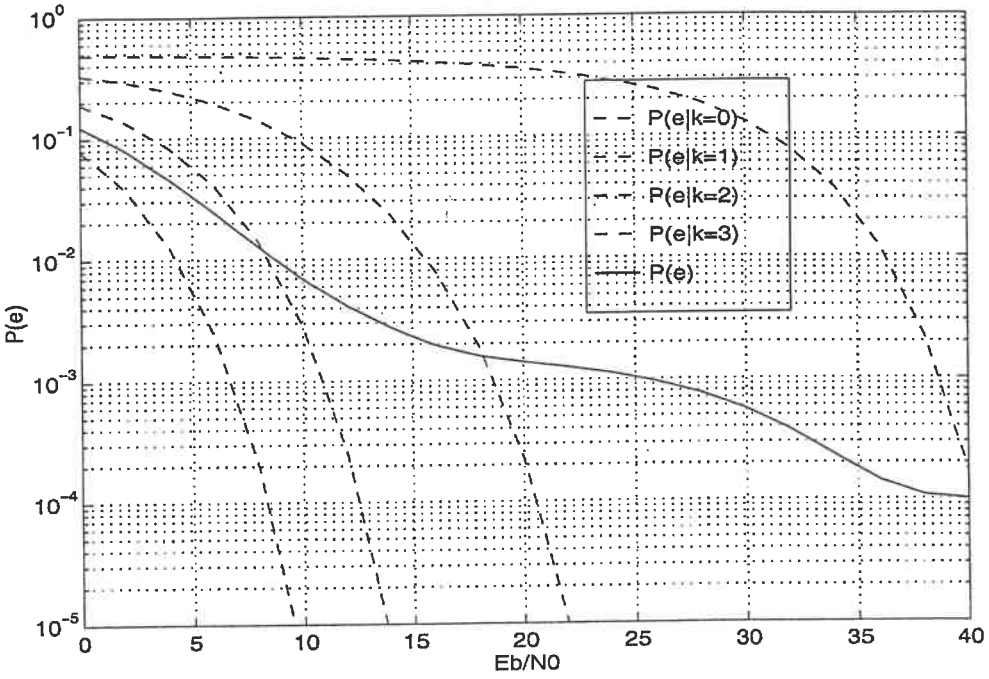


Figure 3: Rotated 4D-hypercube signal set $P_c = 0.1$.

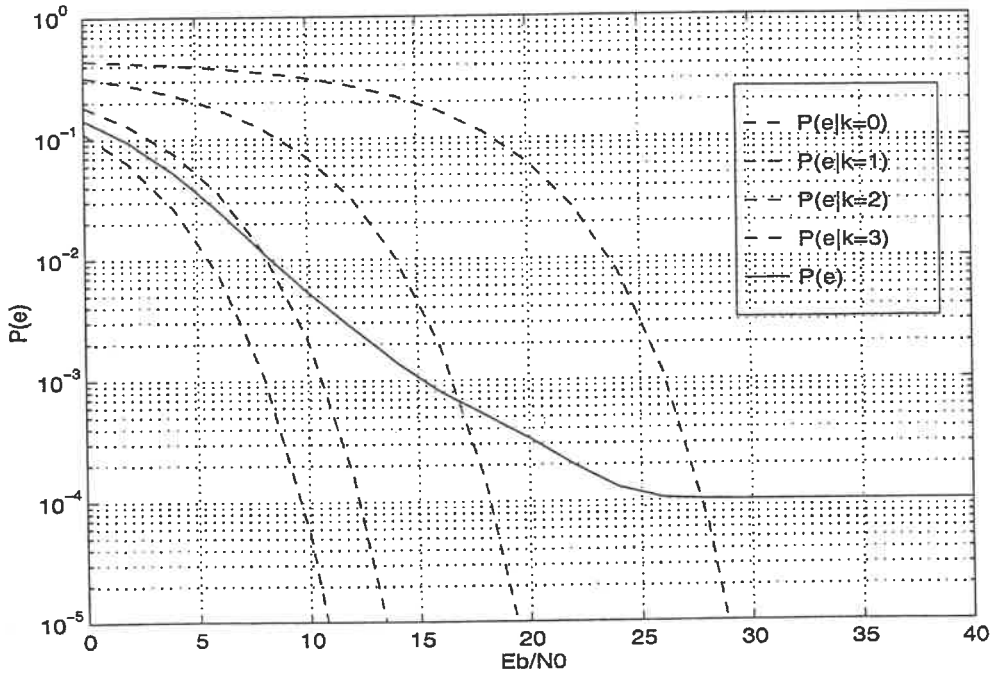


Figure 4: The $\mathbb{Z}_{16}$ signal set $P_c = 0.1$.

References

[1] J. Boutros, E. Viterbo, C. Rastello, and J. C. Belfiore, "Good lattice constellations for both Rayleigh fading and Gaussian channel," *IEEE Trans. on Information Theory*, pp. 502–518, vol. 42, no. 2, March 1996.

[2] J. Boutros and E. Viterbo, "Signal Space Diversity: a Power- and Bandwidth-Efficient Diversity Technique for the Rayleigh Fading Channel," *IEEE Transactions on Information Theory*, pp. 1453–1467, vol. 44, no. 4, July 1998.

[3] G. Caire, E. Leonardi and E. Viterbo, "Collision Resistant Modulation," *ICT '98*, Porto Carras, Greece, 22 – 25 June 1998.

[4] G. Caire, E. Leonardi and E. Viterbo, "Concatenated Coding for Packet Transmission over the Collision Channel," accepted for presentation at *ISIT '98*, Cambridge, MA Aug. 1998.

[5] J. H. Conway, N. J. Sloane: *Sphere packings, lattices and groups*, 2nd ed., 1993, Springer-Verlag, New York.

[6] K. F. Ireland, M. Rosen, *A Classical Introduction to Modern Number Theory (Graduate Texts in Mathematics, Vol 84)*, 2nd ed., Springer Verlag, 1991.

[7] E. Viterbo, E. Biglieri: "A universal lattice decoder", 14-ème Colloque GRETSI, Juan-les-Pins, Sept. 1993.

[8] E. Viterbo, J. Boutros: "A Universal Lattice Code Decoder for Fading Channels", submitted to *IEEE Transactions on Information Theory*, Apr. 1998.

Code Constructions for Block Coded Modulation Systems with Interblock Memory

Chia-Ning Peng,[1] Houshou Chen,[2] John T. Coffey,[2] and Richard G. C. Williams[3]

[1]Texas Instruments, San Jose, California. e-mail: cnpeng@ti.com. [2]EECS Department, University of Michigan, Ann Arbor, Michigan. e-mail: {houshou,scoffey}@eecs.umich.edu. [3]3-COM, San Diego, California. e-mail: Richard_Williams@3com.com

Abstract

This paper examines the performance gains achievable by adding interblock memory to block coded modulation systems. The channel noise considered is additive Gaussian, and the twin design goals are to maximize the asymptotic coding gain and to minimize the number of apparent nearest neighbors. In the case of the additive Gaussian noise channel, these goals translate into the design of block codes of a given weighted or 'normalized' distance whose rate is as high as possible, and whose number of codewords at minimum normalized distance is as low as possible.

It is shown that the effect of designing codes for normalized distance rather than Hamming distance is to ease the problem of determining the best codes for given parameters in the cases of greatest interest, and many such best codes are given.

I. Introduction

Block coded modulation systems have been extensively studied, and have much in common with generalized concatenated codes. A simple modification to these systems, the introduction of "interblock memory", transforms the code design problem into one in which the criterion of interest is a weighted or "normalized" Hamming distance. Compared to regular BCM systems, the systems with interblock memory support higher transmission rates with given probability of error. The only cost is a slightly more complicated structure, and hence more expensive decoding. This extended abstract outlines the relation between the motivating coded modulation problem and the normalized distance coding problem, and develops many optimal codes in this framework. It is remarkable that two extremely simple bounds turn out to be exact in very many of the cases of interest.

Previous work on block coded modulation with interblock memory (BCMIM) systems include the original paper of Lin [3] and subsequent papers by Yamaguchi and Imai [9] and by Lin *et al.* [4], [5] that give some codes, with associated trellis decoding structures and performance simulations, though not general bounds and constructions. This paper generalizes the performance metric and explores the limits of such schemes.

Details not given here and many further codes can be found in [7].

II. Definitions of BCM and BCMIM systems

We assume throughout that the modulation scheme is 2^x-ary, with bits labels assigned by a set partitioning scheme in which the intra-subset minimum distances rise in the ratio $1 : a_2 : \cdots : a_x$.

A baseline BCM 'codeword array' is constructed by taking x binary linear codes $C_1, \ldots, C_x$, each of length n. A sequence of transmitted symbols is obtained by filling the rows of an $x \times n$ array with codewords from these respective codes, then reading labels upwards, column by column.

The codeword array thus has the form:

This work was supported in part by the U.S. Army Research Office under Grant DAAH04-96-1-0377.

1st row:	LSLB's	a codeword of C_1
2nd row:	2nd LSLB's	a codeword of C_2
	...	
$(x-1)$-th row:	$(x-1)$-th LSLB's	a codeword of C_{x-1}
x-th row:	MSLB's	a codeword of C_x

where LSLB denotes the least significant label bits, and MSLB denotes the most significant label bits.

The minimum squared Euclidean distance between two sequences of n symbols in a baseline BCM system is well known to be

$$D_E^2 = \min_i d_i \cdot a_i \cdot E^2,$$

where d_i is the minimum Hamming distance of code C_i. (See, for example, Cusack [1]).

For a baseline BCM system using a length xn code and a 2^x-ary modulation scheme to have a distance gain D, we simply choose code C such that each row code C_i has minimum Hamming distance $d_{\min} \geq \frac{D}{a_i}$. The maximum rate of a baseline BCM system with distance gain D is then

$$\sum_{i=1}^{x} K\left(n, \frac{D}{a_i}\right),$$

where $K(n,d)$ is the maximum dimension of a length n code with minimum Hamming distance $d_{\min} \geq d$.

A. Block Coded Modulation with Interblock Memory

BCMIM systems differ from BCM systems in two ways: the generator matrix is allowed a more general structure, and the codewords are mapped to symbols in a different way.

A BCMIM system with interblock memory between the first i blocks is one with generator matrix of the form

$$\begin{bmatrix} G(C_{1,i}) & \begin{matrix}0\\ \cdots \\ 0\end{matrix} & \cdots & \begin{matrix}0\\ \cdots \\ 0\end{matrix} \\ \begin{matrix}0 & \cdots\end{matrix} & G(C_{i+1}) & 0 & \\ \begin{matrix}\cdots & \end{matrix} & \begin{matrix}0\end{matrix} & \cdots & \begin{matrix}0\end{matrix} \\ \begin{matrix}0 & \cdots\end{matrix} & \begin{matrix}\cdots\end{matrix} & \begin{matrix}0\end{matrix} & G(C_x) \end{bmatrix}$$

in which $G(C_{1,i})$ is the generator matrix of a binary linear code of length in.

The resulting codewords are mapped to symbols in the following staggered way:

Bits $1 \ldots n$	(Previous block)		
(Next block)	Bits $n+1, \ldots, 2n$	(Previous block)	
	(Next block)	...	
			Bits $, \ldots, xn$

,

where the symbols will be read and transmitted going from right to left. (Thus bit xn of the codeword affects the transmitted stream first.)

Thus the difference in generator matrix is that we allow interdependencies between the first i rows of the codeword array; and the difference in mapping is that we allow a codeword of length in to affect in transmitted symbols, rather than a block of i symbols as in regular BCM.

B. Asymptotic Performance and Normalized Weight

Definition: Normalized weight:
Given the basic block size n and the sequence $1 : a_2 : \cdots : a_x$ of increasing intra-subset squared Euclidean distances, the normalized weight of a word c of length xn is defined as

$$W_n(c) = w_1(c) + a_2 \cdot w_2(c) + \cdots + a_x \cdot w_x(c), \tag{1}$$

where $w_i(c)$ is the Hamming weight of the i-th length n basic block of c.

This quantity derives its significance from the fact that it represents the asymptotic (in signal-to-noise ratio) improvement in squared Euclidean distance of the BCMIM system over the uncoded system. Assuming that all past codewords have been decoded correctly, i.e., assuming that all blocks above the one to be decoded are known and correct, the squared Euclidean distance between two codewords c_1 and c_2 is at least

$$E^2 \cdot a_1 \cdot w\left(b_{i,1} - d_{i,1}\right) + \cdots + E^2 \cdot a_x \cdot w\left(b_{i,x} - d_{i,x}\right) = \sum_{1 \le k \le x} E^2 \cdot a_k \cdot w_k\left(c_1 - c_2\right) = E^2 w_n\left(c_1 - c_2\right),$$

i.e., the minimum squared Euclidean distance is higher than in the uncoded case by exactly the minimum normalized weight of the code, as claimed.

The probability of error is not exactly predicted by this minimum squared Euclidean distance due to the conditioning on previous decoding results; however at large enough signal to noise ratios the conditioning effect becomes negligible.

The rates of the BCM and BCMIM systems are lower than uncoded. We calculate asymptotic coding gain by comparing the coded system to an interpolation of uncoded QAM. Omitting details, the result is that the asymptotic coding gain of a system with rate R_2 and minimum squared Euclidean distance $D^2_{E,2}$ over interpolated uncoded QAM is

$$10 \log_{10} \frac{D^2_{E,2}}{D^2_{E,1}} - 10 \log_{10} \frac{2^{R_1} - 1}{2^{R_2} - 1} \text{ dB} \tag{2}$$

as the target probability of error approaches 0, where R_1 and $D^2_{E,1}$ are the rate and minimum squared Euclidean distance of the uncoded QAM system on which the coded system is defined.

C. Nearest neighbors

The asymptotic gain overestimates the coding gain at finite signal to noise ratios, and the difference is mostly attributable to the large number of "apparent nearest neighbors", i.e., the large multiplicity of the most likely error events. The exact difference varies according to target probability of error; a widely used rule of thumb proposed by Forney is that the signal to noise ratio increases by 0.2 dB for each doubling in the number of nearest neighbors.

Although we will not calculate exact coding gains in this discussion, we will take account of the effect of number of nearest neighbors by seeking codes that have as few codewords of minimum weight as possible.

D. Statement of Problem

We thus consider the following problem in 'classical' coding theory.

Given a nondecreasing sequence $1, a_2, \ldots, a_x$, find binary linear codes of length in, and basic block length n, of highest dimension for given *normalized* distance d_n, where the normalized weight of a

codeword is defined as $\sum_{j=1}^{i} a_j w_j$, and w_j is the ordinary Hamming weight of the codeword in bits $(j-1)n+1$ to jn.

Among those codes of highest dimension, find the ones with smallest number of codewords of normalized weight d_n.

We will consider only the case $a_i = 2^i$ in this paper: this is the case for QAM systems.

The notation for codes will be of the form $[n_1|n_2|\cdots|n_x, k, d_n]$ to indicate a code of length $n_1 + n_2 + \cdots + n_x$, dimension k, and normalized distance, i.e., $w_1 + a_2 w_2 + \cdots + a_x w_x$, at least d_n. Thus $[n|n, k, d_n]$ will indicate a code of length $2n$, dimension k, in which for every nonzero codeword the weight on the left plus twice the weight on the right is at least d_n.

III. Bounds and Constructions for Normalized Weight Codes

Many of the cases are accounted for by two very simple bounds: the extension bound for interblock memory between the first two basic blocks, and the 'full rate' lemma for interblock memory between three or more basic blocks. The extension bound is a lower bound, while the full rate lemma is an upper bound, so one must find appropriate upper bounds for the first case, and constructions for the second to demonstrate that the bounds are tight.

A. *Upper bounds*

Lemma 1 (Full dimension lemma) The maximum dimension gain obtainable in extending interblock memory from between the first $i-1$ basic blocks to between the first i blocks is upper bounded by

$$n - K\left(n, \frac{d_n}{a_i}\right)$$

bits.

This follows easily since a length in code with normalized distance d_n has dimension at most n higher than a length $(i-1)n$ code with the same normalized distance.

A.1 Modification of standard bounds

The Plotkin bound argument gives

$$M \leq \frac{d_n}{d_n - \frac{1}{2}(n_1 + a_2 n_2 + \cdots + a_r n_r)},$$

with equality if and only if every nonzero codeword has the minimum normalized weight d_n.

A Griesmer-type bound can also be derived. In the usual situation, we have $n = d_n$ and interblock memory between the first two basic blocks. If we have the codeword $(1_n|0_n)$, then forming the residual code with respect to this codeword gives us a $[0|n, k-1, \lceil d_n/2 \rceil]$ code, i.e., an $[n, k-1, \lceil \lceil d_n/2 \rceil / a_2 \rceil]$ binary linear code. Thus we must have

$$k \leq 1 + K(n, \lceil \lceil d_n/2 \rceil / a_2 \rceil)$$

if the code contains $(1_n|0_n)$.

From the MacWilliams Identities for split weight enumerators, we get constraints that can be used with linear programming. We seek the maximum of $\sum c(w_1, \ldots, w_x)$ subject to the linear constraints $c(0, \ldots, 0) = 1$, $c(w_1, \ldots, w_x) = 0$ for all $w_1, \ldots, w_x$ such that $1 \leq w_1 + a_2 w_2 + \cdots + a_x w_x < d_n$, and $\sum_{(w_1,\ldots,w_x)} c(w_1, \ldots, w_x) \prod_{j=1}^{x} P_{e_j}(w_j; p_j) \geq 0$ for all $w_1, \ldots, w_x$ and all $e_1, \ldots, e_x$ with $0 \leq w_j, e_j \leq p_j$, where $P_k(w, n) = \sum_{i=0}^{k} (-1)^k \binom{w}{i} \binom{n-w}{k-i}$ is the usual Krawtchouk coefficient.

B. Lower bounds on the maximum rate gain of introducing interblock memory to a baseline BCM system

B.1 Extension bound

This bound relates the normalized and Hamming distance problems directly. Assume that a_2 is an integer dividing both n and d_n. Then if an $[n + n/a_2, k, d/a_2]$ code exists, we can take any n/a_2 bits, replicate them a_2 times, and take the resulting n bits to form the first basic block, with the remaining n bits forming the second block. The normalized distance of this code is clearly d, so we have constructed an $[n|n, k, d]$ code. Note also that the normalized weight enumerator of this code is the same as the Hamming weight enumerator for the original code.

This bound gives the highest dimension possible for $n = 12, 14$, and 16 in the QAM problem. The bound falls one short in the cases $n = 6, 8$, and 10, and at least one short in the cases $n = 18$ and 20.

B.2 Other constructions

$|u + v|u|$ constructions: if C_1 and C_2 are codes of the same length, with parameters $[n, k_1, d_1]$ and $[n, k_2, d_2]$ respectively, then the code consisting of all words of the form $|u + v|u|$, with $u \in C_1, v \in C_2$, has parameters $[n|n, k_1 + k_2, d_n \geq \min\{d_2, (1 + a_2)d_1, (a_2 - 1)d_1 + d_2\}]$.

X and Reverse-X constructions: the X construction [6, p. 581] takes codes $C_1 \sim [n_1, k_1, d_1]$, $C_2 \sim [n_1, k_2, d_2]$, with $C_1 \subset C_2$ and $C_3 \sim [n_3, k_2 - k_1, d_3]$. Then the new code consists of the words $|x_i + u|y_i|$, where the x_i's are coset representatives of C_2 in C_1, $u \in C_2$, and $y_i \in C_3$. The x_i's and y_i's can be paired to make the resulting code linear.

This code has normalized parameters $[n_1|n_3, k_2, \geq \min\{d_1, d_2 + a_2 d_3\}]$.

In a reverse-X construction, we interchange the blocks to get a code with normalized parameters $[n_3|n_1, i\, k_2, \geq \min\{a_2 d_1, a_2 d_2 + d_3\}]$.

$X4$ construction: This construction takes four codes $C_i \sim [n_i, k_i, d_i]$ with C_2 a union of b disjoint cosets of C_1, with coset representatives $x_1, \ldots, x_b$, and C_4 a union of b disjoint cosets of C_3, with coset representatives $y_1, \ldots, y_b$. The new code consists of all words of the form $|x_i + u|y_i + v|$, where $u \in C_1$, $v \in C_3$. The normalized parameters are $[n_1|n_3, k_2 + k_3, \geq \min\{d_1, a_2 d_3, d_2 + a_2 d_4\}]$.

IV. Best BCMIM codes

In this section we examine some of the more interesting best codes for normalized distance. For reasons of space we omit the following very interesting codes: an $[11|11, 8, 12]$, a $[13|13, 9, 14]$, and a $[16|16|16, 28, 16]$ code, which the reader is invited to find. We concentrate on the natural case in which the normalized distance is equal to the basic block length.

A. Best codes for QAM-based BCMIM systems with basic block length 8

The codes in the baseline BCM system will be $[8, 1, 8]$, $[8, 4, 4]$, $[8, 7, 2]$ and $[8, 8, 1]$ codes, as given by Cusack. The case $n = d_n = 8$ is a main focus of the papers of Lin *et al.* Lin and Ma give an $[8|8, 8, 8]$ code [4] and Lin, Wang, and Ma follow by extending this to an $[8|8|8, 16, 8]$ code [5]. These papers give efficient trellis representations and performance simulations, but do not discuss constructions, bounds, or optimality.

A.1 Best $[8|8, k, 8]$ codes

If we allow interblock memory between the first two basic blocks, we are seeking the best $[8|8, k, 8]$ code. The baseline BCM case with no interblock memory is obtained by taking the direct sum of the $[8, 1, 8]$ code and the $[8, 4, 4]$ code, and so has dimension 5.

From the full dimension lemma, the gain in dimension when interblock memory is introduced is upper bounded by $n - K(n, d_n/a_2) = 8 - 4 = 4$. The first necessary condition for equality in this bound is that the past subcode has highest possible dimension, i.e., that $k_P = 1$. This implies that the codeword

$(1_8|0_8)$ would be in the code, so that we can apply the Griesmer argument, from which the maximum rate gain is upper bounded by $K(n, d_n/(2a_2)) - K(n, d_n/a_2) = K(8,2) - K(8,4) = 3$, i.e., we have $k \leq 8$ for an $[8|8, k, 8]$ code.

This upper bound can be achieved using the code discussed by Lin and Ma [4] with generator matrix $G_{8/2}$:

$$G_{8/2} = \left[\begin{array}{cccccccc|cccccccc}
1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
\hline
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{array}\right]$$

This code can be obtained from an $X4$ construction, with C_1 an $[8,1,8]$ code, C_2 and C_3 $[8,4,4]$ codes, and C_4 an $[8,7,2]$ code.

It can be shown that this code is (up to permutations) the unique $[16,8]$ code with normalized distance 8. (See [7] for details.)

We note that the best $[16,8]$ code in terms of ordinary Hamming distance, a $[16,8,5]$ shortened quadratic residue code, also unique [8], cannot by the above discussion be partitioned into two halves to get normalized distance 8 (though 7 is possible). This shows that we cannot solve the normalized distance problem in general by simply taking the best code for ordinary Hamming distance and finding the best partition for this code.

A.2 *Best* $[8|8|8, k, 8]$ *codes*

The baseline $[8|8|8, k, 8]$ code is obtained by a direct sum of an $[8|8, k_2, 8]$ code for the first two basic blocks and an $[8,7,2]$ code for the third basic block. Thus using the above code, we can achieve dimension 15 without extending the interblock memory to include the third basic block.

The full dimension lemma indicates that the dimension of any $[8|8|8, k_3, 8]$ code will be at most 8 more than the dimension of the best $[8|8, k_2, 8]$ code. Thus with interblock memory extended to the third basic block, we can potentially increase the dimension by at most 1.

This can in fact be achieved by the code considered by Lin *et al.* [5] in which the $[8|8,8,8]$ code for the first two basic blocks and the $[8,7,2]$ baseline code for the third basic block are glued by the word

$$[0\,0\,0\,0\,0\,0\,1\,1\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0\,1].$$

B. *Best codes for QAM-based BCMIM systems with basic block length 9*

The codes in a baseline BCM system are $[9,1,9]$, $[9,2,6]$, $[9,5,3]$ and $[9,8,2]$.

B.1 *Best* $[9|9, k, 9]$ *codes*

With interblock memory between the first two blocks, we seek the best $[9|9, k, 9]$ code. A direct sum of the $[9,1,9]$ and $[9,2,6]$ codes gives $k = 3$.

Applying split weight linear programming to this case indicates that $k \leq 7$. On fixing the number of codewords accordingly at 128 and using split weight linear programming to minimize the number of codewords with minimum normalized weight 9, we find that the minimum of the objective is 0. This suggests that we might be able to find a $[9|9,7,10]$ code. This is in fact possible, and we will give a couple of different constructions.

This case is also interesting in that it is the first case in which the Griesmer-type bound of Section III-A.1 is not tight.

Since the Griesmer argument indicates that $k \leq 6$ if $k_p = 1$, we must have $k_p = 0$, and so $k_f + k_g = 7$. The punctured future code C_{fg} is then a $[9, 7]$ code, and its dual $C_{fg}^{\perp}$ is a $[9, 2, \leq 6]$ code. Shortening the overall code in any set of m positions holding a codeword of $C_{fg}^{\perp}$ gives a code with $n_1 = 9$, $n_2 = 9 - m$, and $k_s \geq 7 - m + 1$ (this is construction Y1 from MacWilliams & Sloane [6, p. 592]). Suppose that $C_{fg}^{\perp}$ had minimum distance ≤ 5. Then applying Y1 with $m = 5$ gives a code with $n_1 = 9$, $n_2 = 4$, and $k \geq 3$. But the Plotkin bound for this case gives $M \leq 10/(10 - (9/2 + 4)) < 7$. Thus we conclude that $C_{fg}^{\perp}$ is the unique $[9, 2, 6]$ linear code.

We can take the generator matrix for C_{fg} to be

$$\begin{bmatrix} 1&1&1&1&1&1&1&1&1 \\ 1&1&0&0&0&0&0&0&0 \\ 1&0&1&0&0&0&0&0&0 \\ 0&0&0&1&1&0&0&0&0 \\ 0&0&0&1&0&1&0&0&0 \\ 0&0&0&0&0&0&1&1&0 \\ 0&0&0&0&0&0&1&0&1 \end{bmatrix}.$$

Now applying Y1 again, with $m = 6$, we get a code with $n_1 = 9$, $n_2 = 3$, and $k \geq 2$. Applying the Plotkin bound to this case, we find that $M \leq 10/2.5 = 4$; thus the shortened code would meet the Plotkin bound with equality. This in turn would imply that every nonzero codeword has the minimum nonzero normalized weight, as noted earlier. This means that in each of the rows of weight 2 on the right, we would have to have a row of weight exactly 6 on the left. This further narrows the possibilities to generator matrices where, with the above matrix on the right, rows 2 and 3 on the left generate a [9,2,6] linear code, as do rows 4 and 5, and rows 6 and 7.

There are at least two inequivalent codes that satisfy these constraints. The first is obtained by a version of Piret's construction [6, pp. 588–9] applied to the $[9, 6, 2]$ irreducible cyclic code. We first construct the code consisting of 0 and all words of the form

$$u_j = |\gamma(x)^j \theta_1(x) | \gamma(x)^{j+a} \theta_1(x)|$$

for $0 \leq j \leq 62$, where $\theta_1(x) = x^6 + x^3$ is the idempotent of the given irreducible code, and $\gamma(x) = x^8 + x^6 + x^4 + x^3$. A generator matrix for this code can be taken as having rows $u_j, 0 \leq j \leq 5$. The key property is that all nonzero codewords are either rows or permutations of rows of the generator matrix, and so we need only select the remaining parameter a to maximize the minimum distance of the rows of the generator matrix,

$$d' = \min_{0 \leq j \leq 5} (w_j + 2w_{j+a}),$$

where $w_j = \mathrm{wt}(\gamma(x)^j \theta_1(x))$. We have $(w_0, w_1, \ldots, w_6) = (2, 6, 6, 4, 6, 4, 4)$, and $w_{j+7} = w_j$ for all j, so the (unique) best choice for a in this case is 3, giving $d^{\perp} = 10$. (The factor 2 on the right of the expression for d' and the corresponding optimal choice of a are the only differences with the development in [6].) This results in a $[9|9, 6, 10]$ code, and adding the row $(0_9|1_9)$ to the generator matrix does not affect the minimum normalized distance, thus giving a $[9|9, 7, 10]$ code. There are 36 codewords of minimum normalized weight. A generator matrix for this code is

$$\left[\begin{array}{ccccccccc|ccccccccc} 0&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1&1 \\ 0&0&1&0&0&1&0&0&0&1&1&0&1&0&0&0&1&0 \\ 0&1&1&1&0&1&1&1&0&1&0&1&0&1&1&1&1&0 \\ 0&0&1&1&1&1&1&1&0&0&0&0&0&1&1&0&1&1 \\ 1&1&0&1&0&0&0&1&0&0&1&0&1&0&0&1&1&0 \\ 1&0&1&0&1&1&1&1&0&0&1&0&0&1&0&0&0&0 \\ 0&0&0&0&1&1&0&1&1&1&1&1&0&1&1&1&0&0 \end{array}\right].$$

A slightly smaller number of nearest neighbors can be obtained from an optimal $[18, 7, 7]$ code partitioned in an appropriate way. The generator matrix that results is

$$\left[\begin{array}{ccccccccc|ccccccccc} 1&1&0&0&1&1&1&0&1&1&1&1&1&1&1&1&1 \\ 1&1&1&1&1&1&0&0&0&1&1&0&0&0&0&0&0 \\ 1&1&1&0&0&0&1&1&1&1&0&1&0&0&0&0&0 \\ 1&1&0&1&1&0&1&1&0&0&0&0&1&1&0&0&0 \\ 1&0&1&1&0&1&1&0&1&0&0&0&1&0&1&0&0 \\ 1&1&0&1&0&1&0&1&1&0&0&0&0&0&0&1&1 \\ 0&1&1&1&1&0&1&0&1&0&0&0&0&0&0&1&0 \end{array}\right.\left.\begin{array}{c} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{array}\right]$$

which has 33 codewords of minimum normalized weight. This is obtained by shortening the $[24, 12]$ Golay code to get an $[18, 6, 8]$ code with Hamming weight enumerator $1+45x^8+18x^{12}$, then augmenting by the top row.

Split linear programming, constraining the code to have 128 codewords, normalized distance 10 and to have a $[9, 2, 6]$ code as dual of the punctured future code gives a lower bound of 31 codewords of minimum normalized weight. We remark that in the case of a $[9|9, 7, 10]$ code, there is a particularly large number of solutions to the linear programming constraints that are feasible candidates for codes, in the sense of having nonnegative integer weights in both code and dual code. Many are ruled out by the extra constraint that the punctured future code is the dual of the $[9, 2, 6]$ code. We do not know of any such code with 31 codewords of minimum normalized weight, though a feasible integer weight enumerator exists.

C. Best codes for QAM-based BCMIM systems with basic block length 16

The codes in the baseline BCM system will be $[16, 1, 16]$, $[16, 5, 8]$, $[16, 11, 4]$ and $[16, 15, 2]$.

C.1 Best $[16|16, k, 16]$ codes

Either split linear programming or the Y1/Plotkin argument show that $k \leq 12$. This can be achieved using the duplication construction, starting with the $[24, 12]$ Golay code, and forming the first block by duplicating an arbitrary subset of eight bits. This gives 759 nearest neighbors.

This number of nearest neighbors can be reduced to 503 as follows. We use an $X4$ construction with C_1 the $[16, 1, 16]$ code, C_3 the $[16, 5, 8]$ first order Reed-Muller code, and C_4 the extended Hamming $[16, 11, 4]$ code. This still leaves C_2 to be chosen. We choose C_2 to be the $[16, 7, 4]$ doubly even code formed by taking the $[16, 5, 8]$ first order Reed-Muller code, plus two linearly independent cosets; we choose the cosets so that they both have weight enumerators $4x^4 + 24x^8 + 4x^{12}$, and their 'sum' (i.e., the translate of one coset by any element of the other coset) has the same weight enumerator. This is possible if we choose one coset to be the one represented by the Boolean function v_1v_2 and the other to be the one represented by the Boolean function v_1v_3 [6, p. 418]. The 'sum' coset is represented by the Boolean function $v_1v_2 + v_1v_3$. Letting $f(\mathbf{v}) = v_1v_3$, we see that $v_1v_2 + v_1v_3 = f(A\mathbf{v})$, with $A = (1\ 1\ 00/0\ 1\ 0\ 0/0\ 0\ 1\ 0/0\ 0\ 0\ 1)$, from which we conclude (see [6, Thm. 14.4, p. 417]) that the cosets represented by v_1v_3 and $v_1v_2 + v_1v_3$ have the same weight distribution.

The code C_2 thus has weight enumerator $1 + 12x^4 + 102x^8 + 12x^{12} + x^{16}$. A straightforward application of the $X4$ bound gives the lower bound 12 on normalized distance with these codes; however, only codewords with weight distribution $4|4$ can have normalized weight this low, and all others have normalized weight at least 16. Since the number of codewords of weight 4 in C_2 is low, we can match each to words of weight greater than 4 on the right, with some trial and error.

One resulting code has normalized weight enumerator $1 + 503x^{16} + 512x^{20} + 2064x^{24} + 512x^{28} + 503x^{32} + x^{48}$. (Once we have chosen C_2, this number of nearest neighbors is minimum: split linear programming constraining the left code to have the weight enumerator of C_2 gives a minimum of exactly

503 for the number of nearest neighbors. Furthermore this number is the only feasible solution to the split linear programming problem when the past subcode is constrained to have the weight enumerator of C_2.)

Split linear programming gives an overall lower bound of 271 on the number of nearest neighbors.

V. Other Codes

The reader is invited to construct the very interesting codes $[11|11, 8, 12]$, $[13|13, 9, 14]$, and $[16|16|16, 28, 16]$.

VI. Conclusion

A promising twist on block coded modulation constructions has been shown to lead to a modified version of the classical coding problem. A number of interesting codes and code constructions arise in this new scenario.

References

[1] E. L. Cusack, "Error control codes for QAM signaling," *Elec. Letters*, vol. 20, no. 2, pp. 62–63, Jan. 1984.

[2] S. M. Dodunekov and S. B. Encheva, "Uniqueness of some linear subcodes of the extended binary Golay code," *Probl. Peredachi Inform.*, vol. 29, no. 1, pp. 45–51, Jan.–Mar. 1993. (In Russian. English translation in *Probl. Inform. Trans.*, vol. 29, no. 1, pp. 38–43, Jan.–Mar. 1993.).

[3] M.-C. Lin, "A coded modulation scheme with inter block memory," in *Proc. 1990 IEEE Intl. Symp. Inform. Theory (San Diego, California, 1990)*, p. 51.

[4] M.-C. Lin and S.-H. Ma, "A coded modulation scheme with inter block memory," *IEEE Trans. Commun.*, vol. 42, no. 2/3/4, pp. 911–916, Feb./Mar./Apr. 1994.

[5] M.-C. Lin, J.-Y. Wang, and S.-C. Ma, "On block-coded modulation with interblock memory," *IEEE Trans. Commun.*, vol. 45, no. 11, pp. 1401–1411, Nov. 1997.

[6] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*, North-Holland Publishing Company, New York, 1977.

[7] C.-N. Peng, H. Chen, J. T. Coffey, and R. G. C. Williams, "Rate gains in block coded modulation systems with interblock memory," submitted to *IEEE Trans. Inform. Theory*, October 1998.

[8] J. Simonis, "The [18, 9, 6] code is unique," *Discrete Mathematics*, vol. 106/107, pp. 439–448, Sep. 1992.

[9] K. Yamaguchi and H. Imai, "A new block coded modulation scheme and its soft decision decoding," in *Proc. 1993 IEEE Intl. Symp. Inform. Theory (San Antonio, Texas, 1993)*, p. 64.

EFFICIENT MULTIPLEX FOR BAND-LIMITED CHANNELS:
Galois-Field Division Multiple Access

H.M. de Oliveira, R.M. Campello de Souza and A.N. Kauffman
CODEC- Grupo de Pesquisas em Comunicações
Departamento de Eletrônica e Sistemas - CTG- UFPE
C.P. 7800, 50711-970, Recife-PE, Brazil
e-mail: {hmo,ricardo}@npd.ufpe.br

Abstract

A new "Efficient-bandwidth code-division-multiple-access (CDMA) for band-limited channels" is introduced which is based on finite field transforms. A* multilevel *code division multiplex exploits orthogonality properties of nonbinary sequences defined over a complex finite field. Galois-Fourier transforms contain some redundancy and only cyclotomic coefficients are needed to be transmitted yielding compact spectrum requirements. The primary advantage of such schemes regarding classical multiplex is their better spectral efficiency. This paper estimates the "bandwidth compactness factor" relatively to Time Division Multiple Access TDMA showing that it strongly depends on the alphabet extension. These multiplex schemes termed Galois-field Division Multiplex (GDM) are based on transforms for which there exists fast algorithms. They are also convenient from the hardware viewpoint since they can be implemented by a Digital Signal Processor.

Keywords- Digital multiplex, Code-divison multiple access, Hartley-Galois transform, Finite field transforms, Spread sequence design.

1. Introduction

The main title of this paper is, apart from the term multiplex, literally identical to a Forney, Gallager and coworkers paper issued more than one decade ago [FOR et al. 84], which analyzed the benefits of coded-modulation techniques. The large success achieved by Ungerboeck's coded-modulation came from the way redundancy was introduced in the encoder [UNG 82]. In classical channel coding, redundant signals are *appended* to information symbols in a way somewhat analogous to time division multiplex TDM (envelope interleaving). It was believed that introducing error-control ability would increase bandwidth. An efficient way of introducing such an ability without sacrificing rate nor requiring more bandwidth consists in adding redundancy by an alphabet expansion. This technique is particularly suitable for channels in the narrow-band region. A similar reasoning occurs in the multiplex framework where many people nowadays believe that mux must increase bandwidth requirements.

In the present work, the coded-modulation idea is adapted to multiplex: Information streaming from users are not combined by interleaving (like TDM) but rather by a *signal alphabet expansion.* The mux of users' sources over a Galois Field GF(p) deals with an expanded signal set having symbols from an extension field $GF(p^m)$, m>1. As a consequence, the multiplex of N band limited channels of identical maximal frequency B leads to *bandwidth requirements less than* N B, in contrast with TDMed or FDMed signals.

The design of such an "efficient bandwidth mux" is based upon Galois field Transforms (GFT) such as the Finite Field Fourier Transform (FFFT) introduced by Pollard [POL 71]. The FFFT been successfully applied to perform discrete convolution and image processing [REE et al. 77, REE&TRU 79], among many other applications. In this paper we are concerned with a new finite field version [CAM et al. 98] of the integral transform introduced by R.V.L. Hartley [HAR 42, BRI 92]. Alike classical Galois-Fourier transforms [BLA 79, CAM&FAR 85], Finite Field Hartley transforms (FFHT) defined on a Gaussian integer set $GI(p^m)$ [CAM et al. 98] contains some redundancy and only the cyclotomic coset leaders of the transform coefficients need to be transmitted. This yields a new "Efficient-bandwidth Code Division multiplex for band-limited channels". These multiplexes may present lower bandwidth requirements than TDM. Tradeoffs between the extension of the alphabet and the bandwidth are exploited in the sequel. This paper shows that the "bandwidth compactness factor" relatively to TDM depends on the length m, the alphabet extension.

Another point to mention is that the superiority of digital mux regarding analog mux is essentially due to the low complexity of TDM. The majority of current multiplex systems follows plesiochronous (PDH) or synchronous (SDH) hierarchy [SIL&SHA 96]. Besides presenting higher spectral efficiency (bits/s/Hz) than classical multiplex, the new code division multiplex (CDM) schemes introduced here are based on fast transforms, so they also seem to be attractive from the implementation viewpoint [HOU 87]. Although most mux systems today intended to optical fiber which are not yet bandlimited channels, multiplex has also been adopted on satellite channels. Applications of such a multiplex will be on transponders and more probably on *cellular mobile*

communications. Cable Television (CATV) can also benefit of such a technique.

2. A New Mux Scheme: Galois-Division-Multiplex

Digital multiplex normally alludes Time Division Multiplex (TDM). However, it can also be achieved by Coding Division Multiplex (CDM) which has recently been the focus of interest, specially after the IS-95 standardization of the CDMA system for cellular telephone [QUAL 92]. The CDMA is now becoming the most popular multiple access schemes for mobile communication. In this section we introduce a new class of mux schemes based upon finite-field transforms which can be implemented by fast transform algorithms. *Classical multiplex increases simultaneously the transmission rate and the bandwidth by the same factor*, keeping thus the spectral efficiency unchanged. In order to achieve (slight) better spectral efficiencies, classical CDMA uses waveforms presenting a nonzero but residual correlation. We introduce here a new and powerful issue on CDMA techniques.

Given a signal v over a finite field GF(p), we deal with the Galois domain considering the spectrum V over an extension field GF(p^m) which corresponds to the Finite Field Transform (Galois Transform) of the signal v [BLA 79, CAM et al. 98].

As an alternative and attractive implementation (figure 1), the multiplex can be carried out by a Galois Field Transform (FFFT/FFHT) so the DEMUX corresponds exactly to an *Inverse Finite-Field Transform* of length N | p^m-1.

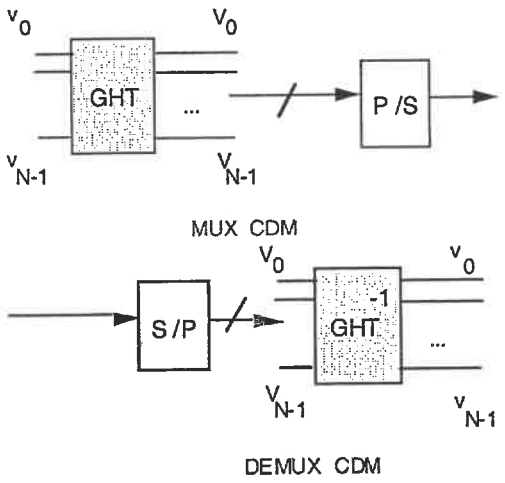
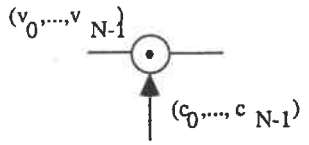


Figure 1. Implementation of Galois Field Transform (GFT) multiplex.

Each symbol in the ground field GF(p) has duration T seconds. An N-user mux can be designed on the extension field GF(p^m) where N | p^m-1. For the sake of simplicity, we begin with m=1 and consider a (p-1)-channel mux as follows. Typically, we can consider GF(3) corresponding to Alternate Mark Inversion (AMI) signaling.

Definition. A Galois modulator carries a pairwise multiplication between a signal $(v_0,v_1,...,v_{N-1})$, $v_i \in GF(p)$ and a carrier $(c_0,c_1,...,c_{N-1})$, with $c_i \in GI(p)$. ■

A pictorial representation of a Galois modulator:



A (p-1)-CDM considers digital carrier sequences per channel as versions of the cas function $\{cas_i k\}_{k=0}^{p-1}$ over the Galois (complex) field GI(p). The cas (cos and sin) function is defined in terms of finite field trigonometric functions [CAM et al. 98] according to $cas_i k := \cos_i k + \sin_i k$.

Carrier 0:
$\{cas_0 0 \; cas_0 1 \; cas_0 2 \; ... \; cas_0(N-1)\}$
Carrier 1:
$\{cas_1 0 \; cas_1 1 \; cas_1 2 \; ... \; cas_1(N-1)\}$
...
Carrier j:
$\{cas_j 0 \; cas_j 1 \; cas_j 2 \; ... \; cas_j(N-1)\}$
...
Carrier N-1:
$\{cas_{N-1} 0 \; cas_{N-1} 1 \; cas_{N-1} 2 ... \; cas_{N-1}(N-1)\}$.

The cyclic digital carrier has the same duration T of an input modulation symbol, so that it carries N slots per data symbol. The interval of each cas-symbol is T/N and therefore the bandwidth expansion factor when multiplexing N channels may be roughly N, *the same result as* FDM and TDM/PAM.

A first scheme of the multiplex is showed in the figure 2: The output corresponds exactly to the Galois-Hartley Transform of the "user"-vector $(v_0,v_1,...,v_{N-1})$. Therefore, it contains all the information about all channels. Each coefficient V_k of the spectrum has duration T/N.

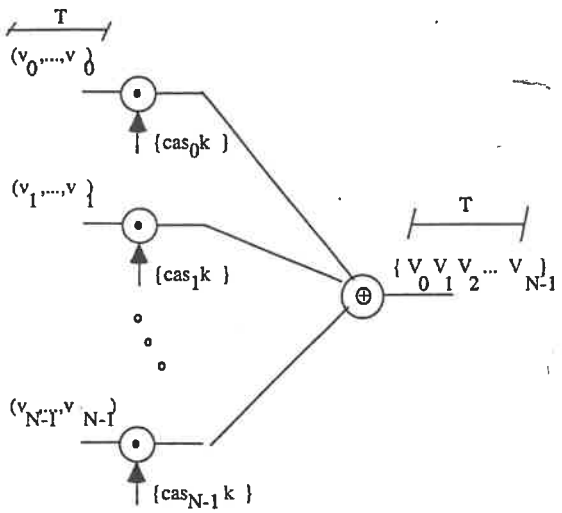


Figure 2. Galois-Field MUX: Spreading sequences.

These carriers can also be viewed as spreading waveforms [MAS 95]. An N-user mux has N spreading sequences, one per channel. The requirements to achieve Welch's lower bound according to Massey and Mittelholzer [MAS&MIT 91], are achieved by $\{cas_i k\}_{k=0}^{p-1}$ sequences. The matrix [{ $cas_i k$ }] presents both orthogonal rows and orthogonal columns having the same "energy".

A naive example is presented in order to illustrate such an approach (figure 3). A 4-channel mux over GF(5) can be easily implemented i=0,1,...,p-2=3. It is straightforward to see that such signals are not FDMed nor TDMed. GI(5)-valued cas(.) function is shown on table I assuming α equals to 2, an element of GF(5) of order 4.

TABLE I. Cas Function on GI(5) with α=2, an element of order 4.

$cas_0 0$=1+j0	$cas_0 1$=1	$cas_0 2$=1	$cas_0 3$=1
$cas_1 0$=1+j0	$cas_1 1$=j3	$cas_1 2$=4	$cas_1 3$ =2j
$cas_2 0$=1+j0	$cas_2 1$=4	$cas_2 2$=1	$cas_2 3$ =4
$cas_3 0$=1+j0	$cas_3 1$=2j	$cas_3 2$=4	$cas_3 3$ =j3

Putting these results as complex carriers, one has:

{cas0k}= {1, 1, 1, 1}
{cas1k}= {1, j3, 4, j2}
{cas2k}= {1, 4, 1, 4}
{cas3k}= {1, j2, 4, j3}.

Therefore, a 4-channel multiplex based on GHT is shown in figure 3.

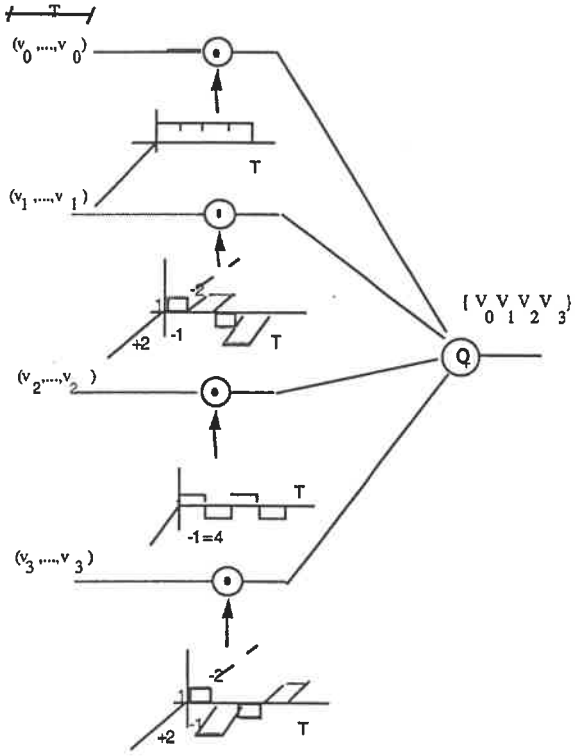


Figure 3. Interpreting Galois-Hartley Transform over GF(5) as spreading waveforms.

The digital carriers are defined on a complex Galois field GI(p) where the element $j = \sqrt{-1}$ may or not may belong to GF(p), although the original definition [CAM et al. 98] considers -1 as a quadratic non residue in GF(p). Two distinct cases are to be considered: p=4k+1 or p=4k+3, k integer. If p ≡ 1 (mod 4), then -1 is a quadratic residue. For instance, considering j∈GF(5) then $2^2 \equiv -1 \pmod 5$ so $j = \sqrt{-1} \equiv 2 \pmod 5$. Two-dimensional digital $\{cas_i k\}_{k=0}^{p-1}$ carriers then degenerated to one-dimensional carriers.

Considering the above example, carriers are reduced to Walsh carriers!

{cas0k}= {1, 1, 1, 1} = {1,1, 1, 1}
{cas1k}= {1, 1, 4, 4} = {1,1,-1,-1}
{cas2k}= {1, 4, 1, 4} = {1,-1,1, -1}
{cas3k}= {1, 4, 4, 1} = {1,-1,-1, 1}

$$\begin{vmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{vmatrix} \Leftrightarrow \begin{vmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{vmatrix} = \begin{vmatrix} \Delta & \Delta \\ \Delta & -\Delta \end{vmatrix} =$$

$$[\mathrm{WAL}(k,i)].$$

In the absence of noise, there is no cross-talk from any user to any other one, which corresponds to orthogonal carrier case.

If channels number 1, 2, 3, and 4 are transmitting {4, 0, 1, 2} respectively, the mux output will be (2, 3+4j, 3, 3+j), which corresponds to

$\{4,0,1,2\} \otimes \{1,1,1,1\} \equiv 2 \mod 5$
$\{4,0,1,2\} \otimes \{1,j3,4,j2\} \equiv 3+4j \mod 5$
$\{4,0,1,2\} \otimes \{1,4,1,4\} \equiv 3 \mod 5$
$\{4,0,1,2\} \otimes \{1,j2,4,j3\} \equiv 3+j \mod 5$

There is *no gain* when the transform is taken without alphabet extension. However, we have a nice interpretation of CDM based on finite field transforms.

3. A New CDM scheme based on Galois-Hartley Transforms

So far we have essentially considered Finite Field Transforms from GF(p) to GF(p). Extension fields can be used and results are much more interesting: The Galois-Field Division Multiple Access schemes. The advantage of the new scheme named GDMA over FDMA / TDMA regards its higher spectral efficiency.

The new multiplex is carried out over the Galois domain instead the Frequency or time domain. Figure 4 exhibits a block diagram of transform-based multiplexes. First, the Galois spectrum of N-user GF(p) signals is evaluated. The spectral compression is achieved by eliminating the redundancy: only the leaders of cyclotomic cosets are transmitted. The demultiplex is carried out (after signal regeneration) first recovering the complete spectrum by "filling" missing components from the received coset leaders. Then, the inverse finite field transform is computed so as to obtain the demux signals. Another additional feature is that GDM implementations can be made more efficient if fast algorithms for computing the transforms involved are used.

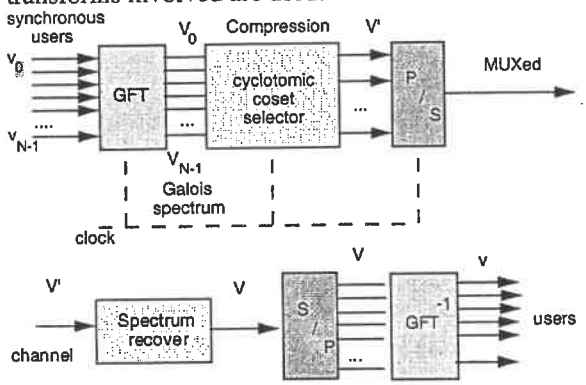


Figure 4. Multiplex based on Finite Field Transforms.

Suppose that users data are p-ary symbols transmitted at a speed $B_1:=1/T$ bauds. Let us consider the problem of multiplexing N users. Traditionally the bandwidth requirements will increase proportionally with the number N of channels, i.e., $B_N=NB_1$ Hz.

Thereafter the number of cyclotomic cosets associated with a Galois-Fourier (or Galois-Hartley) finite field spectrum is denoted ν_F (respectively ν_H). The clock driving GHT symbols is N/ν times faster than the input baud rate.

Definition. The bandwidth compactness parameter γ_{cc} is defined as $\gamma_{cc}:=N/\nu$. ■
It plays a role somewhat similar to the coding asymptotic gain γ_c on coded modulation [UNG 82].

Transform-multiplex, i.e., mux based on finite field transforms are very attractive compared with FDM/ TDM due to their better spectral efficiency as it can be seen in table II (appendix).

Another point that should be stressed is that instead of compressing spectra (eliminating redundancy), it is possible to use all the coefficients to introduce some error-correction ability. The valid spectrum sequences generate a multilevel block code.

Lemma 1. For an N-user GDMA system over $GI(p^m)$ with $N \mid p^m-1$, only a number $\nu = \gamma_{cc}^{-1}N$ (see below) of finite-field transform coefficients are required to be transmitted.
proof. According to Mœbius' inversion formula, $I_k(q)=\frac{1}{k}\sum_{d|k}\mu(d)q^{k/d}$ gives the number of distinct irreducible polynomials of degree k over GF(q), where μ is the Mœbius function [McE 87]. Therefore, the number ν_F of cyclotomic sets on the Galois-Fourier transform $(V_0,V_1,...,V_{N-1})$ is given by

$$\nu_F = \sum_{k|m} I_k(p) - 1.$$

Since each pair of cosets containing reciprocal roots is clustered, then

$$\nu_H = \frac{\nu_F - (N \bmod 2)}{2} + 1. \quad ■$$

As a rule-of-thumb, the number of cosets (in fact the gain γ_{cc}) when $N=p^m-1$ is roughly given by $\nu_F \approx \left\lceil \frac{N}{m} \right\rceil$ and $\nu_H \approx \left\lceil \frac{1}{2}\left\lceil \frac{N}{m} \right\rceil + 1 \right\rceil$, where $\lceil x \rceil$ is the ceiling function (the smallest integer greater than or equal to x). A simple example over $GF(3) \rightarrow GI(3^3)$ is presented below: Factoring $x^{26}-1$ one obtains $\nu_F=10$ and $\nu_H=6$. For the FFHT, $V_k^3 = V_{26-k3}$ (indexes modulo 26) according to [CAM et al. 98, Lemma 1].

FFFT cosets	FFHT cosets
C0=(0)	C0=(0)

C1=(1,3,9)
C2=(2,6,18)
C4=(4,12,10)
C5=(5,15,19)
C7=(7,21,11)
C8=(8,24,20)
C13=(13)
C14=(14,16,22)
C17=(17,25,23).

C1=(1,23,9,25,3,17)
C2=(2,6,18,8,24,20)
C4=(4,14,10,22,12,16)
C5=(5,11,19,21,15,7)
C13=(13).

Another interesting possibility is multiplexing without the cyclotomic coset compression. Although such a GDM presents the same spectral efficiency as TDM or FDM it introduces some error-correcting ability yielding a better performance.

By way of interpretation, Hartley transforms can be seen as some kind of Digital Single Side Band since the number of cyclotomic cosets of the FFHT is roughly half that of the FFFT. We can therefore say that "*GDM/FFFT is to FDM/AM as GDM/FFHT is to FDM/SSB.*"

Gain of GDM. The gain on the number of channels GDMed regarding to TDM/FDM over the same bandwidth is N-ν, which corresponds to $g_{\%}=100(1-\gamma_{cc}^{-1})\%$.
proof. The bandwidth gain is $g_{band}=B_{TDM}/B_{GDM}=\gamma_{CC}$ and the saved Bandwidth is given by $B_{TDM}-B_{GDM}$. Calculating how many additional B_1-channels (users) can be introduced:

$$(B_{TDM}-B_{GDM})/B_1=(1-\frac{1}{\gamma_{cc}})N. \quad \blacksquare$$

In the previous exemple, a 26-user GDM furnishes a 20-channels gain (≈77%) regarding to TDM.

Indeed, a more formal treatment of spectra should be tried. Power Spectral density calculations must be evaluated by using cyclic Autocorrelation Functions (ACF) of the carriers. As usual, it is assumed that users' data sequences are independent.

Lemma 2. Users' signal sequences can be viewed as wide-sense stationary random processes in both time and Galois-domain, by assuming an uniform probability distribution on GF(p) symbols.
proof.
time domain data stream:
$\ldots(v_0^{(-1)},v_1^{(-1)},\ldots,v_{N-1}^{(-1)})\ (v_0^{(0)},v_1^{(0)},\ldots,v_{N-1}^{(0)})\ (v_0^{(1)},v_1^{(1)},\ldots,v_{N-1}^{(1)})\ \ldots$
Supposing $v_i\in\{0,\pm1,\pm2,\ldots\pm(p-1)/2\}$ equally likely, the mean and the ACF of the discrete process are given respectively by (**E** denotes the expected value):

$$\mathbf{E}\{v_i^{(m)}\}=\mathbf{E}\{v_i\}=0 \quad (\forall i), \text{ and}$$

$$R_v(j)=E\{v_i^{(m)}v_{i-j}^{(m)*}\}=E\{v_i\,v_{i-j}^{*}\}=0,\ i\neq j,$$

$$R_v(0)=p^{-1}\left(0+1^2+2^2+3^2+\ldots+(\frac{p-1}{2})^2\right):=\mathbf{P}.$$

Galois domain data stream:
$\ldots(V_0^{(-1)},V_1^{(-1)},\ldots,V_{N-1}^{(-1)})\ (V_0^{(0)},V_1^{(0)},\ldots,V_{N-1}^{(0)})\ (V_0^{(1)},V_1^{(1)},\ldots,V_{N-1}^{(1)})\ \ldots$

From $V_k=\sum_{i=0}^{N-1}v_i cas_k i \quad (\forall k)$, it follows that:

$$E\{V_k^{(m)}\}=E\{V_k\}=\sum_{i=0}^{N-1}E(v_i)cas_k i=0 \quad (\forall k).$$

The ACF of the random process (Galois spectrum sequence) is:

$$R_V(j)=E(V_k V_{k-j}^{*})=$$

$$=\sum_{i=0}^{N-1}\sum_{\Delta i=0}^{N-1}E(v_i v_{i-\Delta i}^{*})cas_k i\ cas_{k-j}^{*}(i-\Delta i)$$

$$=\sum_{i=0}^{N-1}\sum_{\Delta i=0}^{N-1}R_v(\Delta i)cas_k i\ cas_{k-j}^{*}(i-\Delta i)$$

$$=R_v(0)\sum_{i=0}^{N-1}cas_k i\ cas_{k-j}^{*}i.$$

From orthogonality properties of cas_k function [CAM et al. 98], it follows that

$$R_V(j)=0,\ j\neq 0 \text{ and } R_V(0)=R_v(0). \quad \blacksquare$$

This result can be used to show that the multiplex based upon the finite field Hartley transform does not shape the signal power spectrum. Denote by $S_b(t)$ the complex envelope of the generalized QAM (G-QAM) signal s(t), i.e.,

$$s(t)=\Re e\{s_b(t)\exp(j2\pi f_c t)\} \text{ where}$$

$$s_b(t)=\sum_{m=-\infty}^{+\infty}\sum_{k=0}^{N-1}V_k^{(m)}u_{k,m}(t),$$

$$u_{k,m}(t)=u(t-kT-mNT).$$

Theorem 3. The Power Spectral Density of GDM signals is given by $|U(f)|^2$ which depends just on the shape of the spectrum of the shaping filter u(t).
proof. It can be shown that $s_b(t)$ is a cyclostationary process [GAD&FRA 75] but we treat it as a wide-sense stationary one by introducing a random phase uniformly distributed over one block. Therefore we consider a related signal

$$\tilde{s}_b(t)=\sum_{m=-\infty}^{+\infty}\sum_{k=0}^{N-1}V_k^{(m)}u_{k,m}(t;\theta),$$

where $u_{k,m}(t,\theta)=u_{k,m}(t-\theta)$, θ being uniformly distributed in the interval (0,NT). The autocorrelation function (ACF) $R_{\tilde{s}_b}(t,t-\tau)=\mathbf{E}\{\tilde{s}_b(t)\tilde{s}_b^{*}(t-\tau)\}$ of the complex envelope is

$$R_{\tilde{s}_b}(t,t-\tau)=\sum_{m=-\infty}^{+\infty}\sum_{\Delta m=-\infty}^{+\infty}\sum_{k=0}^{N-1}\sum_{\Delta k=0}^{N-1}\mathbf{E}V_k^{(m)}V_{k-\Delta k}^{(m-\Delta m)*}\frac{1}{NT}\int_0^{NT}u_{k,m}(t,\theta)u_{k-\Delta k,m-\Delta m}^{*}(t-\tau,\theta)d\theta$$

where indexes k-Δk are taken mod N.

We first remark that spectra (blocks) are transmitted independently so that complex symbols on different N-vectors are uncorrelated, yielding

$$\sum_{\Delta k=0}^{N-1}\mathbf{E}V_k^{(m)}V_{k-\Delta k}^{(m-\Delta m)*}=0,\quad \Delta m\neq 0.$$

Furthermore, we suppose the sequence of N-dimensional signals is wide-sense stationary (lemma 2) and that there exist a block ACF $R_V(j)$, j=0,1,2,...N-1. The ACF of the complex envelope is therefore

$$R_{\tilde{s}_b}(t,t-\tau)=$$

$$\frac{1}{NT}\sum_{k=0}^{N-1}\sum_{\substack{j=0\\ j\neq k}}^{N-1}R_V(j)\sum_{m=-\infty}^{+\infty}\int_0^{NT}u_{k,m}(t,\theta)u_{k-j,m}^{*}(t-\tau,\theta)d\theta$$

By an appropriate change of variables, we obtain

$$R_{\tilde{s}_b}(t,t-\tau)=$$

$$\frac{1}{NT}\sum_{k=0}^{N-1}\sum_{\substack{j=0\\ j\neq k}}^{N-1}\mathbf{E}V_k^{(m)}V_{k-j}^{(m)*}\int_{-\infty}^{+\infty}u(\alpha+\tau-kT)u^{*}(\alpha)d\alpha$$

Putting the above equation in a simpler notation results in:

$$R_{\tilde{s}_b}(\tau)=R_{\tilde{s}_b}(t,t-\tau)=\frac{1}{NT}\sum_{j=0}^{N-1}R_V(j)v(\tau-jT)$$

where $v(\tau-jT)=\int_{-\infty}^{+\infty}u(\alpha+\tau-jT)u^{*}(\alpha)d\alpha$.

Indeed, $R_{\tilde{s}_b}(\tau)=\frac{1}{NT}R_V(0)v(\tau)$. Taking now the Fourier transform of the ACF, we finally have by the Wiener-Kinchine relation

$S_{\tilde{s}_b}(f)=\frac{\mathbf{P}}{NT}|U(f)|^2$, so the power spectrum of the multiplexed signal follows directly from the modulation theorem. ■

What can we say about the alphabet extension? A simple upper-bound on the bandwidth compactness factor can be easily derived. The greatest extension that can be used depends on the signal-to-noise ratio, since the total rate cannot exceed Shannon Capacity over the Gaussian channel. Therefore,

$$B_{GDM}\gamma_{cc}\log_2 p\leq B_{GDM}\log_2\left(1+\frac{S}{N}\right)\text{ bps, or}$$

$$\gamma_{cc}\leq\log_p\left(1+\frac{S}{N}\right).$$

4. Conclusions

Finite field transforms are offered as a new tool of spreading sequence design. New digital multiplex schemes based on such transforms have been introduced which are *multilevel* Code Division Multiplex. They are attractive due to their better spectral efficiency regarding to classical TDM/CDM which require a bandwidth expansion roughly proportional to the number of channels to be multiplexed. This new approach is promising for cellular mobile communications and channels supporting a high signal-to-noise ratio. Moreover, the Galois-Field Division (GDM) implementation can be easily carried out by a Digital Signal Processor (DSP). Combined multiplex and error-correcting ability should be investigated. Another nice payoff of GDM is that when Hartley Finite Field transforms are used, the mux and demux hardware are exactly the same. It is proved that GDM based on Finite Field Hartley Transform does not shape the signal Power Spectrum. They can directly be applied in multiple access digital schemes.

ACKNOWLEDGMENTS

This first author expresses his deep indebtness to Professor Gérard Battail whose philosophy had a decisive influence on his way of looking to coding and multiplex.

REFERENCES

[BLA 79] R.E. Blahut, Transform Techniques for Error Control Codes, *IBM J. Res. Develop*, **23**, n.3, pp. 299-314, May, 1979.

[BRI 92] J. Brittain, Scanning the past: Ralph V.L. Hartley, *Proc. IEEE*, vol.**80**, p. 463, 1992.

[CAM et al. 98] R.M. Campello de Souza, H.M. de Oliveira, A.N. Kauffman and A.J.A. Paschoal, "Trigonometry in Finite Fields and a new Hartley Transform", IEEE *International Symposium on Information Theory*, ISIT, MIT Cambridge,MA, THB4: Finite Fields and Appl., p. 293, 1998. (see http://lids.mit.edu/ISIT98)

[CAM&FAR 85] R.M. Campello de Souza and P.G. Farrel, Finite Field Transforms and Symmetry Groups, *Discrete Mathematics*, **56**, pp. 111-116, Elsevier pub., 1985.

[FOR et al. 84] G.D. Forney Jr, R.G. Gallager, G.R. Lang, F.M. Longstaff and S.U. Qureshi, Efficient modulation for Band-limited channels, *IEEE J. Select. Areas Commun.*, SAC **2**, Sept., pp. 632-646, 1984

[GAD&FRA 75] W.A. Gadner and L.E. Franks, Characterization of cyclostationary random process, *IEEE Trans. Info. Theory*, **21**, Jan., pp. 4-15, 1975.

[HAR 42] R.V.L. Hartley, A more symmetrical Fourier analysis applied to transmission problems, *proc. IRE*, vol **30**, pp. 144-150, 1942.

[HON&VET 93] J.J. Hong and M. Vetterli, Hartley Transforms over Finite Fields, *IEEE Trans. Info. Theory*, **39**, n.5, pp.1628-1638,Sept.,1993. Also Computing m DFT's over GF(q) with one DFT over GF(q^m), *IEEE Trans. Info. Theory*, **29**, n.1, pp. 271-274, Jan., 1993.

[HOU 87] H.S. Hou, The fast Hartley transform algorithm, *IEEE Trans. Comput.*, vol. **36**, pp. 147-156, 1987.

[MAS 95] J.L. Massey, Towards an Information Theory of Spread-Spectrum Systems, in: *Code Division Multiple Access Communications*, Eds S.G. Glisic and P.A. Leppänen, Boston, Dordrecht and London, Kluwer, pp. 29-46, 1995.

[MAS&MIT 91] J.L. Massey and T. Mittelholzer, Welch's bound and sequence sets for Code-Division-Multiple-Access Systems, *Proc. of Sequences*'91, Springer-Verlag, 1991.

[McE 87] R.J. McEliece, Finite Fields for Computer Scientist and Engineers, Kluwer Ac. Pub., 1987.

[POL 71] J.M. Pollard, The Fast Fourier Transform in a Finite Field, *Math. Comput.*, **25**, pp. 365-374, Apr., 1971.

[QUAL 92] Qualcomm, The CDMA Network Engineering Handbook, Qualcomm Inc., San Diego, CA, 1992.

[REE et al. 77] I.S. Reed, T.K. Truong, V.S. Kwoh and E.L. Hall, Image Processing by Transforms over a Finite Field, *IEEE Trans. Comput.*, **26**, pp. 874-881, Sep., 1977.

[REE&TRU 79] I.S. Reed and T.K. Truong, Use of Finite Field to Compute Convolution, *IEEE Trans. Info. Theory*, pp. 208-213, Mar., 1979.

[SIL&SHA 96] C.A. Siller and M. Shafi, Eds., SONET/SDH: A Sourcebook of Synchronous Networking, IEEE press, 1996.

[UNG 82] G. Ungerboeck, Channel Coding with multilevel/phase signals, *IEEE Trans. Info. Theory*, IT **28**, pp. 55-67, Jan., 1982.

APPENDIX

TABLE II. A Spectral Efficiency Comparison for Multiplex Systems.

	one-user	N-users TDMed or FDMed	GDMed
Transmission rate	$R_{i-user} = \frac{\log_2 p}{T}$	$R = \sum_i R_{i-user} = N\frac{\log_2 p}{T}$ bps	$R = \sum_i R_{i-user} = N\frac{\log_2 p}{T}$ bps
Bandwidth requirements	$B_1 = \frac{1}{T}$ Hz	$B_N = \frac{1}{T/N} = NB_1$ Hz	$B_{GDM} = \frac{1}{T/(\gamma_{cc}^{-1}N)} \frac{1}{\gamma_{cc}}(NB_1)$ Hz
Spectral efficiency	$\eta_{i-user} = \log_2 p$ bits/s/Hz	$\eta_{mux} = \log_2 p$ bits/s/Hz	$\eta_{GDM} = \gamma_{cc} \log_2 p$ bits/s/Hz

Perfect Codes and Balanced Generalized Weighing Matrices

Dieter Jungnickel
Lehrstuhl für Diskrete Mathematik, Optimierung
und Operations Research
Universität Augsburg
D–86135 Augsburg, Germany
Jungnickel@Math.uni-augsburg.de

and

Vladimir D. Tonchev*
Department of Mathematical Sciences
Michigan Technological University
Houghton, Michigan 49931, USA
tonchev@mtu.edu

December 9, 1998

Abstract

It is proved that any set of representatives of the distinct 1-dimensional subspaces in the dual code of the unique linear perfect single-error-correcting code of length $\frac{q^d-1}{q-1}$ over $GF(q)$ is a balanced generalized weighing matrix over the multiplicative group of $GF(q)$. Moreover, this matrix is characterized as the unique (up to equivalence) weighing matrix for the given parameters with minimum q-rank. The classical, more involved construction for this type of BGW-matrices is discussed for comparison, and a few monomially inequivalent examples are included.

*Research partially supported by a research grant of the Alexander von Humboldt Foundation.

1 Introduction

We assume familiarity with some basic facts and notions from coding theory and combinatorial design theory ([2], [6], [7]). In particular, we require the following definitions.

Definition 1.1 A ***balanced generalized weighing matrix*** $BGW(m, k, \mu)$ over a group G is an $m \times m$ matrix $W = (g_{ij})$ with entries from $\overline{G} := G \cup \{0\}$ such that each row of W contains exactly k nonzero entries, and for every $a, b \in \{1, \ldots, m\}$, $a \neq b$, the multiset $\{g_{ai}g_{bi}^{-1} : 1 \leq i \leq m, g_{ai}, g_{bi} \neq 0\}$ contains exactly $\mu/|G|$ copies of each element of G.

Definition 1.2 Two matrices over $GF(q)$ are said to be *monomially equivalent* if one is obtainable from the other by permutations of rows and columns and multiplying rows and columns by nonzero elements from $GF(q)$.

There are many monomially inequivalent balanced generalized weighing matrices with the same parameters that are distinguishable by their rank over $GF(q)$. Some small examples are listed in the Appendix. These examples, as well as many other ones are obtained by decomposing difference sets with the "classical" parameters $(2^{d+1} - 1, 2^d - 1, 2^{d-1} - 1)$ with respect to a subdesign with classical parameters [4].

It is the aim of this note to give a simple coding-theoretical construction of the balanced generalized weighing matrices with parameters $(\frac{q^d-1}{q-1}, q^{d-1}, q^{d-1} - q^{d-1})$ of minimum q-rank, and to characterize these matrices as the unique (up to monomial equivalence) matrices of minimum q-rank.

2 A class of weighing matrices from the simplex code

The q-ary *simplex code* $S_d(q)$ of length $\frac{q^d-1}{q-1}$, where $d \geq 2$ and q is a prime power, is defined as a linear code over $GF(q)$ with a generator matrix having as columns representatives of all distinct 1-dimensional subspaces of the d-dimensional vector space $GF(q)^d$. In other words, S_d is the dual code of the unique linear perfect single-error-correcting code of length $\frac{q^d-1}{q-1}$ over $GF(q)$, that is, the q-ary analogue of the Hamming code.

Lemma 2.1 *(i) The Hamming weight enumerator of $S_d(q)$ is given by*

$$1 + (q^d - 1)X^{q^{d-1}}.$$

(ii) The supports of all nonzero vectors in $S_d(q)$ are the blocks of a symmetric 2-$(\frac{q^d-1}{q-1}, q^{d-1}, q^{d-1} - q^{d-2})$ design isomorphic to the design with blocks the complements of hyperplanes in $PG(d-1, q)$.

The statement (i) is a folklore fact. For a proof of (ii), see [8].

Theorem 2.2 *Any* $\frac{q^d-1}{q-1} \times \frac{q^d-1}{q-1}$ *matrix* M *with rows a set of representatives of the* $\frac{q^d-1}{q-1}$ *distinct 1-dimensional subspaces of* $S_d(q)$ *is a balanced generalized weighing matrix with parameters*

$$m = \frac{q^d-1}{q-1},\ k = q^{d-1},\ \mu = q^{d-1} - q^{d-2}$$

over the multiplicative group $GF(q)^*$ *of* $GF(q)$.

Proof. Let $x = (x_1, x_2, \ldots, x_m)$ be a row of M. By 2.1, if $y = (y_1, y_2, \ldots, y_m)$ is any other row of M, there are exactly $q^{d-1} - q^{d-2}$ indices i such that $x_i \neq 0$ and $y_i \neq 0$. We want to show that the multiset

$$S = \{x_i \cdot y_i^{-1} \mid y_i \neq 0\}$$

contains every nonzero element of $GF(q)$ equally frequently, that is, exactly q^{d-2} times. Note that multiplication of any column or a row of M by a nonzero element of $GF(q)$ preserves the set of frequencies of the elements in S. Therefore, multiplying the ith column of M by y_i^{-1} for all i such that $y_i \neq 0$, transforms M into a matrix M' in which x is transformed into $x' = (\ldots x_i y_i^{-1} \ldots)$ and y becomes a $(0,1)$-vector y'. By Lemma 2.1,

$$D_H(x', y') = D_H(x, y) = q^{d-1},$$

where D_H denotes the Hamming distance. Let

$$I = \{i \mid x'_i \neq 0 \text{ and } y'_i \neq 0\}.$$

In order for M' (and therefore M) to be a balanced generalized weighing matrix, the multiset $S' = \{x'_i \mid i \in I\}$ has to contain each of the $q-1$ nonzero elements of $GF(q)$ the same number of times, that is, q^{d-2} times. Note that $q^{d-2} = (q^{d-1} - q^{d-2})/(q-1)$ is the average frequency of an element in S'. If there is some $\beta \in GF(q)$, $\beta \neq 0$ that occurs more than q^{d-2} times in S' then multiplying the vector x' by β^{-1} gives a vector x'' such that $D_H(x'', y') < q^{d-1}$; but another application of Lemma 2.1 shows $D_H(x'', y') = q^{d-1}$, a contradiction. □

Theorem 2.3 *Let* M *be any balanced generalized weighing matrix with parameters* $(\frac{q^d-1}{q-1}, q^{d-1}, q^{d-1} - q^{d-2})$ *over* $GF(q)^*$. *Then*

$$rank_q M \geq d.$$

Moreover, the equality $rank_q M = d$ *holds if and only if* M *is monomially equivalent to a matrix obtained by the construction of Theorem 2.2.*

Proof. Since the rows of M and their nonzero multiples constitute $q^d - 1$ distinct nonzero vectors in the row space C of M over $GF(q)$, the rank of M over $GF(q)$, $rank_q(M)$, is at least d. If $rank_q(M) = d$ then C consists of the zero vector and all nonzero multiples of the rows of M. Since the supports of the rows, as well as the columns of M are the blocks of a symmetric design with parameters 2-$(\frac{q^d-1}{q-1}, q^{d-1}, q^{d-1} - q^{d-2})$, that is, a 2-design with $k > \lambda$, any two columns of M are linearly independent over $GF(q)$. Consequently, the orthogonal subspace (or dual code) $C^\perp$ has minimum Hamming distance at least 3. Thus, $C^\perp$ is a linear single-error-correcting code of length $\frac{q^d-1}{q-1}$ and dimension $\frac{q^d-1}{q-1} - d$ that meets the Hamming (sphere packing) bound, hence $C^\perp$ must be monomially equivalent to the unique linear perfect code with these parameters, namely, the q-ary Hamming code. Any basis of C formed by rows of M consists of d linearly independent rows. By the remarks about $C^\perp$, the set of columns of B is a set of distinct representatives of all 1-dimensional subspaces of the d-dimensional vector space $GF(q)^d$. Consequently, the matrix B is unique up to monomial equivalence over $GF(q)$. □

3 A comparison with the classical construction

There is a "classical" construction for balanced generalized weighing matrices with parameters

$$m = \frac{q^{d+1} - 1}{q - 1}, \ k = q^d, \ \mu = q^d - q^{d-1}$$

over the multiplicative group $GF(q)^*$ of $GF(q)$ which we will now recall; see [3] and [5] for background. We warn the reader that the notation used by us differs from that in [3], where we used $\lambda = \mu/n$ instead of μ as the third parameter of a BGW-matrix.

Let R be the set of elements of $GF(q^{d+1})$ of trace 1 relative to $GF(q)$. Then R is a classical relative difference set with parameters $(\frac{q^{d+1}-1}{q-1}, q-1, q^d, q^{d-1})$ in $GF(q^{d+1})^*$ relative to $N = GF(q)^*$. Let β be a primitive element of $GF(q^{d+1})$ and define a $(\frac{q^{d+1}-1}{q-1} \times \frac{q^{d+1}-1}{q-1})$-matrix $W = (w_{ij})$ with entries in $GF(q)$ as follows. If there is a (necessarily unique) element r of $R\beta^i$ in the coset $N\beta^j$, then set $w_{ij} = \beta^{-j}r$, and otherwise set $w_{ij} = 0$. Then W is the desired BGW-matrix. Actually, this construction gives BGW-matrices of a special form, namely ω-circulant matrices, where $\omega = \beta^{-1}$. Recall that an ***ω-circulant*** matrix is defined by the following property: each row of W is obtained from the preceding row by shifting every entry but the one in the final column one position to the right, whereas the entry in the final column is first multiplied by ω and then the result is put in the first position of the shifted row. Formally, we have

$$w_{i,j} = w_{i+1,j+1} \text{ for } j = 1, \ldots, m-1 \text{ and } w_{i+1,1} = \omega w_{i,m}.$$

By a result of [3], ω-circulant *BGW*-matrices over a cyclic group and cyclic relative difference sets are actually equivalent concepts:

Result 3.1 *Let N be a cyclic group of order n, and let ω be a generator for N. Then the existence of a ω-circulant BGW-matrix with parameters (m, k, μ) over N is equivalent to the existence of an (m, n, k, λ)-difference set in the cyclic group G of order $v = mn$ relative to the unique subgroup of order n (which may, of course, be identified with N), where $\lambda = \mu/n$.* □

It is also known that the classical BGW-matrices can be put into circulant form whenever $(q-1, \frac{q^{d+1}-1}{q-1}) = 1$. This is an easy consequence of the following analogue of Result 3.1, see [3]. Here a matrix $A = (a_{g,h})$ whose rows and columns are indexed by the elements of a group H is called ***H-invariant*** provided that

$$a_{g,h} = a_{g+k,h+k} \qquad \text{for all } g, h, k \in H.$$

In particular, A is circulant if and only if it is H-invariant for a cyclic group H.

Result 3.2 *Let H and N be groups of orders m and n, respectively, and let $G = H \times N$. Then the existence of an H-invariant BGW-matrix with parameters (m, k, μ) over N is equivalent to the existence of an (m, n, k, λ)-difference set in G relative to N, where $\lambda = \mu/n$.* □

For obvious reasons, relative difference sets of the form described in Result 3.2 are called ***splitting***. We now have the following result.

Proposition 3.3 *The matrices constructed in Theorem 2.2 can be put into ω-circulant form. They can also be put circulant form whenever $(q-1, \frac{q^{d+1}-1}{q-1}) = 1$.*

Proof. The second assertion follows from the well-known fact that the q-ary Hamming code (and hence its dual, the simplex code) is a cyclic code in these cases. In general, the matrices of Theorem 2.2 can be put into ω-circulant form, since the q-ary Hamming code always is a constacyclic code; see, for instance, [1], p. 303. □

It is an open problem whether or not our construction gives the same matrices (up to monomial equivalence) as the classical construction outlined in this section, though the few small examples we have checked out suggest this to be the case. Although the p-ranks of the classical affine difference sets are known, this does not seem to imply a simple formula for the rank of the corresponding BGW-matrices over $GF(q)$. In any case, the construction presented here is obviously much simpler to implement than the classical one.

4 Appendix

In Table 4.1 we use the following notation: if α is a primitive root of $GF(q)$ then i denotes α^{i-1} for $1 \leq i \leq q-1$, and 0 is the zero in $GF(q)$.

Table 4.1 *Some inequivalent BGW-matrices*

No.	(m, k, μ)	q	q-rank	first row	ω
1	$(9, 8, 7)$	8	2	1 6 1 2 6 6 5 1 0	7
2	$(9, 8, 7)$	8	4	5 1 7 7 4 5 7 5 0	7
3	$(73, 64, 56)$	8	3	0 0 4 0 4 5 4 0 3 5 3 7 4 5 2 0 7 3 7 7 5 3 6 4 3 5 2 7 7 1 3 0 5 4 5 3 0 4 4 4 7 7 4 3 2 2 1 5 7 3 2 7 1 1 0 4 2 4 3 6 2 3 6 0 3 7 3 5 7 7 2 3 4	7
4	$(73, 64, 56)$	8	9	0 0 4 0 5 5 6 0 4 7 5 7 2 2 4 0 6 5 7 4 2 7 6 4 5 1 3 1 5 5 1 0 1 2 2 7 0 4 1 5 4 1 3 4 5 2 3 5 1 7 1 6 6 3 0 6 4 7 1 7 6 6 5 0 4 6 5 1 4 1 4 4 4	7
5	$(85, 64, 48)$	4	4	1 1 1 1 3 1 0 1 2 2 3 1 0 0 3 1 0 3 0 3 2 2 0 1 1 0 0 0 2 2 1 1 3 0 1 2 1 0 3 2 1 3 3 3 1 0 2 1 0 1 0 0 2 0 3 0 2 3 1 3 3 1 3 1 3 2 0 0 2 1 2 3 1 1 1 0 2 2 3 3 2 1 2 2 0	3
6	$(85, 64, 48)$	4	16	3 2 3 3 2 2 0 2 3 3 1 3 0 0 2 3 0 2 0 2 3 1 0 2 3 0 0 0 3 3 2 2 2 0 2 3 2 0 1 3 3 2 1 1 2 0 1 3 0 2 0 0 1 0 1 0 1 2 2 2 2 3 1 3 2 3 0 0 3 3 3 2 2 3 2 0 1 1 2 2 3 2 1 3 0	3

Acknowledgment The second author wishes to thank the university of Augsburg, Germany, for the hospitality during his visit as a research fellow of the Alexander von Humboldt Foundation.

References

[1] E.R. Berlekamp, *Algebraic Coding Theory*, McGraw-Hill Book Company, New York 1968.

[2] T. Beth, D. Jungnickel and H. Lenz: *Design theory (2nd edition)*. Cambridge University Press (in press).

[3] D. Jungnickel: On automorphism groups of divisible designs. *Canadian J. Math.* **34** (1982), 257–297.

[4] D. Jungnickel and V.D. Tonchev, Decompositions of difference sets *J. Algebra* (to appear).

[5] A. Pott: *Finite geometry and character theory*. Lecture Notes in Mathematics **1601**, Springer, Berlin (1995).

[6] V.D. Tonchev: *Combinatorial Configurations.* Longman- Wiley, New York (1988).

[7] V.D. Tonchev: Codes. In: *The CRC Handbook of Combinatorial Designs* (Eds. C.J. Colbourn and J.H. Dinitz). CRC Press, Boca Raton (1996), pp. 517-543.

[8] V.D. Tonchev: Linear perfect codes and a characterization of the classical designs. *Designs, Codes and Cryptography* (to appear).

Higher Order Covering Radii

P. Solé*
CNRS-I3S,
ESSI, BP 145,
Route des Colles,
06 903 Sophia-Antipolis,
France

Abstract

The $m-$**covering radius** of a code is introduced here as the smallest radius such that for all $m-$sets of vectors in ambient space, one of them or one of their paiwise sums lies in a ball of that radius about some codeword. This parameter is different from the multicovering radius and easier to deal with. An application to the construction of good long nonlinear error-correcting codes as union of cosets of linear codes is given. When the code is linear this parameter is the $m-$diameter of the coset graph. Therefrom upper bounds based on the dual weight distribution are derived. **Quasi-random codes**, a new class of codes comprising the BCH and the duals of many interesting cyclic codes used in CDMA is introduced in this note. These bounds show that long quasi-random codes have bounded higher covering radii.

Keywords: Covering Radius, Coset Graph, Weight Distribution, Quasi-random codes.

*sole@essi.fr

1 Introduction

Motivated by the study of streamciphers [5], Klapper introduced in [6] a generalization of the concept of covering radius called multi-covering radius. We introduce here for the first time an alternative parameter the $m-$**covering radius** which might be easier to study and is natural to consider when constructing dense nonlinear codes from union of cosets of a linear code. The parameters and properties of the Kerdock codes, for instance, give an upper bound on a $m-$covering radius of $RM(1,2s)$. We provide spectral bounds based on the connection with $m-$diameters of graphs a new combinatorial parameter introduced in [1]. We introduce here for the first time the new class of **quasi-random codes** which are defined by spectral conditions and show that long quasi-random codes have a bounded covering radius. Example of such codes abound in the literature of cyclic codes due the constructions of low-correlation sequences [8] by application of the Weil Riemann Hypothesis or direct computations.

2 Definitions

2.1 Codes

We denote by $\mathbf{F}$ the finite field of order 2. For a subset $X \subseteq \mathbf{F}^n$, we denote by X^* the set $X \setminus 0$. The Hamming distance between x and y with $x, y \in \mathbf{F}^n$ is denoted by $d(x,y)$. The distance $d(X,Y)$ between two sets of vectors X and Y is $\min\{d(x,y)|x \in X,\ y \in Y\}$. The $m-$covering radius $R_m(C)$ of a code C of length n is the smallest integer r such that for all $m-$sets of vectors in $\mathbf{F}^n$, *one of them or one of their pairwise sums* at least is at distance at most r of some codeword. Formally

$$R_m(C) := \max\{d(S \bigcup (S+S)^*, C) |\, S \subseteq \mathbf{F}^n \& |S| = m\}.$$

The $m-$multicovering radius $t_m(C)$ of a code C of length n is the smallest integer r such that for all $m-$tuples of vectors in $\mathbf{F}^n$, *all of them* are at distance at most r of some codeword. Clearly $t_m(C) \geq R_m(C)$. Since it is known [6] that $t_m(C) \geq n/2$ and is shown below (Proposition 1) that $R_m(C) \leq R_1(C)$, a quantity very often $\leq n/2$ it seems that the two parameters are unrelated.

Recall some classical definitions. The entropy function H is defined for $x \in (0,1)$ as

$$H(x) := -x\log_2(x) - (1-x)\log_2(1-x).$$

The Reed-Muller code of order r and length 2^s is denoted by $RM(r,s)$. The usual covering radius is denoted by R. Plainly $R_1(C) = R(C)$.

2.2 Graphs

We consider finite undirected graphs with no loops and no multiple edges and view them as metric spaces for the shortest path distance. A $k-$code in a graph is a $k-$subset of vertices. Its distance is the minimum of its pairwise distance between distinct elements. The *k-diameter* of Γ say $D_k(\Gamma)$ is the largest possible distance a k-code in Γ can have. Note that D_2 is the standard diameter. Denote by T the diagonal matrix indexed by V such that $T_{x,x}$ is the degree of $x \in V$, by A the adjacency matrix of Γ and let $L = T - A$. The *Laplace operator* is then defined as

$$\mathcal{L} := T^{-1/2} L T^{-1/2}.$$

Let

$$\lambda_0 = 0 \leq \lambda_1 \leq \ldots \leq \lambda_{v-1}$$

be the eigenvalues of the Laplace operator arranged in increasing order. It is not hard to check that the whole *spectrum* fits into $[0,2]$. In particular, if the graph is Δ-regular then $\lambda_i = 1 - \mu_i/\Delta$, where μ_i is the i^{th} eigenvalue (decreasing order) of the adjacency matrix.

3 Basic Properties

The following is immediate from the definition.

Proposition 1 *The function $m \mapsto R_m(C)$ is non increasing. In particular for $m \geq 1$ we have $R_m(C) \leq R_1(C)$.*

Some special values are easy to determine. Let E_n denote the even weight code and E_n' its dual the repetition code. For instance,

$$R_m(\mathbf{F}^n) = 0$$

for all m.

$$R_m(E_n) = 1,$$

for $m \leq 2^{n-1}$, and 0 otherwise. Define the **jumps** of C as the integers $m > 1$ where $R_m(C) < R_{m-1}(C)$. For instance E_n admits the jump $2^{n-1}+1$. Call $J(C)$ the set of jumps of C. Call $V(n,r)$ the volume of the ball of radius r.

Proposition 2 *Let C be a binary code of packing radius e. If C is not perfect then*

$$\min(J(C)) \leq 1 + 2^n - |C|V(n,e).$$

Proof: Pick m vectors with $m > 2^n - |C|V(n,e)$, then one of them at least falls within distance e of the code. By hypothesis and the preceding inequality $m > 1$. □

More generally let $N(r,C)$ denote the number of vectors at distance exactly r from C.

Proposition 3 *If C is a binary code then*

$$\min(J(C)) \leq 1 + N(R_1(C), C).$$

For instance, the first jump of $RM(1,2s)$ gives a lower bound on the total number of bent functions. We leave as an open problem $R_m(E_n')$ and $R_m(0)$. In fact determining $R_m(0)$ for all $m \leq 2^n$ is closely related to finding $A(n,d)$ for all $d \leq n$.

Proposition 4 *For all $d \leq n$ and $m \leq 2^n$ we obtain*

- *If $m+1 < A(n,d)$ then $R_m(0) > d$*
- *If $m+1 = A(n,d)$ then $R_m(0) = d$*
- *If $R_m(0) \geq d$ then $A(n,d) \geq m+1$*

4 Graphical Approach

Let $G(C)$ denote the Cayley graph on the cosets of a binary linear $[n,k,d]$ code, with generators the cosets of weight one. The spectrum of that graph is well-known [2].

Lemma 1 *If C is a linear $[n,k,d]$ code with dual weights (resp. frequencies) w_i' (resp. A_i') then $G(C)$ is a multigraph on 2^{n-k} vertices with Laplacian eigenvalues $\lambda_i = 2w_i'/n$ and multiplicities A_i'. If, furthermore, $d > 2$, this graph has no multiple edges and is regular of degree n.*

Recall from [2] that the usual covering radius $R_1(C)$ is no one else than the diameter of $G(C)$. This generalizes as follows for $m > 1$.

Proposition 5 *The $(m+1)-$diameter of $G(C)$ is equal to the $m-$covering radius of C.*

Proof: Given $m+1$ cosets with pairwise distance at least $D_{m+1}(G(C))$ being attained for some pair, we can always assume, by translation, that they comprise the trivial coset C. Therefore

$$R_m(C) \geq D_{m+1}(G(C)).$$

In the other direction, to a set S of m vectors of $\mathbf{F}^n$ corresponds an $m+1$-code of $G(C)$ consisting of C and the m cosets $s+C$ for s running over S. Assume S realizes $R_m(C)$. The minimum distance of that graphical code of size $m+1$ is then $R_m(C)$ by definition. Therefore

$$R_m(C) \leq D_{m+1}(G(C)).$$

□

Some easy values follow

Corollary 1 *If C is the Hamming code then $R_m = 1$ for all m. If C is the extended Hamming code $RM(s-2,s)$ then $R_m = 2$ for $m \leq 2^s$ and 1 otherwise.*

Proof: In the first case $G(C)$ is the complete graph. In the second case $G(C)$ is the bipartite complete graph $K_{2^s,2^s}$. □

We derive an anlogue of the sphere covering bound for linear codes. This yields a lower bound on the jump of a linear code.

Corollary 2 *If C is a linear code of length n and dimension k then*

$$2^{n-k} \leq (m+1)V(n, R_m(C)-1).$$

This entails

$$\min(J(C)) \geq 2^{n-k}/V(n, R(C)-1),$$

and, furthermore

$$N(R,C)+1 \geq 2^{n-k}/V(n,R(C)-1).$$

Proof: Apply the standard Varshamov-Gilbert argument to a code of size $m+1$ and minimum distance $R_m(C)$ in the coset graph $G(C)$. Combine with Proposition 3 to obtain the last assertion. □

For instance if $C = RM(1,2s)$ knowing that R_1 is $n/2+O(\sqrt{n})$ is enough to assert that the first jump is at least of the order of 2^s, for large s. A better bound will be derived in §6.

5 Spectral Bounds

We are now in a position to use the following result [1, Cor.3].

Theorem 1 *If G is a graph of size v distinct from the complete graph its $k-$diameter is bounded above as*

$$D_k(G) \leq \left\lfloor \frac{\cosh^{-1} v}{\cosh^{-1} \frac{2+\lambda_k}{2-\lambda_k}} \right\rfloor$$

if $\lambda_k \neq \lambda_{v-1}$. If furthermore $2 \geq \lambda_k + \lambda_{v-1}$ the bound can be sharpened to

$$D_k(G) \leq \left\lfloor \frac{\cosh^{-1} v}{\cosh^{-1} \frac{1}{1-\lambda_k}} \right\rfloor$$

This last result, applied to $G(C)$ yields.

Corollary 3 *If C is a linear $[n,k]$ code with dual weights (resp. frequencies) w'_i (resp. A'_i) then let the cumulative weight frequencies be*

$$f_j := \sum_{i=1}^{j} A'_i.$$

With these notations, provided C is not the perfect Hamming code its $m-$covering radius is bounded above for $f_{j-1} < m \leq f_j$ as

$$R_m(C) \leq \lceil \cosh^{-1}(2^{n-k})/\cosh^{-1}(\frac{n+w'_j}{n-w'_j}) \rceil \leq \lceil (n-k+1)/\log_2(\frac{n+w'_j}{n-w'_j}) \rceil.$$

If, furthermore the weights $< n$ are symmetric wrt $n/2$ then

$$R_m(C) \leq \lceil \cosh^{-1}(2^{n-k})/\cosh^{-1}(\frac{n}{n-2w'_j}) \rceil \leq \lceil (n-k+1)/\log_2(\frac{n}{n-2w'_j}) \rceil.$$

Proof: Apply Theorem 1 to the coset graph of a quasi-random or $m-$quasi-random code, using Lemma 1 as a dictionary to translate into graphical terms the coding-theoretic hypotheses. □

Consider the double-error correcting BCH code of length $n = 2^s - 1$ for s odd and codimension $n - k = 2s$. By [7, p.451] we know that the dual distance is $2^{s-1} - 2^{(s-1)/2}$ with frequency $n(2^{s-2} + 2^{(s-3)/2})$. For large s the preceding bound entails

$$R_m(C) \leq 5.$$

Define a family of binary codes of unbounded length n and codimension $\kappa \log_2(n)$ to be **quasi-random** if their dual weights different from zero and n lie in the range $[n/2 - c\sqrt{n}, n/2 + c\sqrt{n}]$, for some constant c. Such codes are abundant in the litterature due to the use of arithmetic geometry bounds like the Carlitz-Uchiyama bound [7, p.280]. The first examples are the $t-$error correcting BCH codes for fixed t [7, p.280]. More examples can be found by taking duals of the cyclic codes used in direct sequence CDMA to generate low correlation sequences [8]. See the example sections of [2, 9].

More generally, we shall say that a code is $m-$quasi-random if the preceding condition on the dual weights is replaced by the same range condition bearing on the dual weights indexed in the range $[m, n-m]$. Clearly, a quasi-random code is $m-$quasirandom for all m in the range $[1, n/2]$. The preceding bounds entail the following result.

Theorem 2 *Let C be quasi-random $[n, k]$ code. For length large enough*

$$R_m(C) \leq R_1(C) \leq 2\kappa + 1.$$

If, more generally, C is $m-$quasi-random then

$$R_m(C) \leq 2\kappa + 1.$$

Proof: By definition for quasi-random codes $n - k \sim \kappa \log_2(n)$ for large n, and for $w \in [n/2 - c\sqrt{n}, n/2 - c\sqrt{n}]$ we can write

$$\log_2((n+w)/(n-w)) \geq \log_2(c\sqrt{n}) \sim 0.5 \log_2(n).$$

Plugging these estimates into Corollary 3 yields the first assertion. The proof of the second assertion is similar. □

In the special case of BCH codes, the first statement is proved in [2, §6.3.1] and constitutes an alternative to the character sum approach of [3, Corollary 3.3]. The second statement is a generalization to higher covering radius.

6 Union of cosets

Many good nonlinear codes are unions of high weight cosets of a fixed linear code. The chief examples of these are the Kerdock and Preparata codes, not to mention the Delsarte-Goethals and Goethals-Delsarte codes of [4].

Proposition 6 *If C is linear then there are m cosets $x_i + C$ such that the code*

$$D := C \bigcup \bigcup_{i=1}^{m} (x_i + C)$$

satisfies $d(D) = \min(d(C), R_m(C))$.

As a consequence we obtain the following bound, which, while stated here for the first time is certainly implicit in, e.g. the Preparata section of [7].

Corollary 4 *If $R_m(C) \leq d(C)$ then*

$$(m+1)|C| \leq A(n, R_m(C)).$$

The celebrated Kerdock code is a union of 2^s cosets of $RM(1,s)$, for even $s > 1$. Therefrom we obtain

$$R_{2^s-1}(RM(1,s)) \geq 2^{s-1} - 2^{s/2}.$$

Is this bound tight? It is the case for $s = 4$ by the fact [7] that $A(16,6) = 64$. However, it is true for all such s by Proposition 1 and $R_1(RM(1,s) = 2^{s-1} - 2^{s/2}$. This shows that $\min(J(RM(1,s))$ for even s is at least 2^s.

Similarly, the $[2^s - 1, 2^s - s - 1, 3]$ Hamming code for even $s > 2$ is a union of 2^s translates of the shortened Preparata code $P(s)^*$ [7, p.475]. This implies

$$R_{2^s-1}(P(s)^*) \geq 3.$$

This is actually equal in view of the optimality of the Hamming code. The fact that the double-error correcting BCH code of the preceding section is contained in the perfect Hamming code yields

$$R_m(C) \geq 3$$

for $m \leq 2^s - 1$. Alternatively, equality also holds by Proposition 1 and $R_1(P(s)^*) = 3$.

7 Acknowledgement

We thank P. Langevin and S. Perennes for helpful discussions and the anonymous referees for helpful remarks.

References

[1] Chung, F.R.K, Delorme, C. and Solé, P., Multidiameters and Multiplicities, submitted to the special issue of European J. of Combinatorics in memoriam F. Jaeger (1999).

[2] Delorme, C. and Solé, P., Diameter, covering number, covering radius and eigenvalues, European J. Combinatorics 12 (1991), 95-108.

[3] T. Helleseth, On the covering radius of cyclic linear codes and arithmetic codes, Discrete Appl. Math 11 (1985) 157-173.

[4] F. Hergert, On the Delsarte-Goethals codes and their formal duals, Discrete Math 83 (1990) 249-263.

[5] A. Klapper, On the existence of secure feedback shift registers, Proc. Eurocrypt '96, Springer Lect. Notes in Comp. Sc. 1070 (1996) 256-267.

[6] A. Klapper, The multicovering radii of codes, IEEE Trans. on Information Theory, 43 (1997) 1372-1377.

[7] F.J. MacWilliams, N.J. A. Sloane, *The Theory of Error-Correcting Codes*, North-Holland (1977).

[8] M.B. Pursley, D. Sarwate, Crosscorrelation properties of pseudorandom and related sequences, Proc. of the IEEE, May 1980.

[9] P. Solé, Limit law on the weight distribution of binary codes, IEEE Trans. on Information Theory IT-36 (1990) 229-231.

Strengthening the Gilbert-Varshamov bound

Alexander Barg*
Bell Labs, Lucent Technologies
600 Mountain avenue 2C-375
Murray Hill, NJ 07974-0636, USA

Sugi Guritman† and Juriaan Simonis‡
Delft University of Technology
Faculty of Information Technology and Systems
P.O. Box 5031
2600 GA Delft, the Netherlands

December 1, 1998

Abstract

The paper discusses some ways to strengthen (nonasymptotically) the Gilbert-Varshamov bound for linear codes. The unifying idea is to study a certain graph constructed on vectors of low weight in the cosets of the code, which we call the Varshamov graph. Various simple estimates of the number of its connected components account for better lower bounds on the minimum distance of codes, some of them known in the literature.

1 Introduction

Let C be a q-ary linear code of length n, dimension k and minimum distance d, in short an $[n,k,d]_q$-code. The Varshamov bound [13] guarantees, for any given q,n,k, the existence of a linear $[n,k,d]_q$ code with a certain relation between the parameters n,k,d,q (see Prop. 12 below). Moreover, [13] suggests a greedy procedure of constructing a parity-check matrix for a code whose parameters meet the bound. Gilbert [6] suggested a similar greedy algorithm that produces (not necessarily linear) codes whose parameters satisfy a similar relation. Asymptotically, both bounds give the same function; therefore, it became common to join them into the "Varshamov-Gilbert bound."

To improve the Varshamov-Gilbert bound asymptotically is a notoriously difficult task [11]. However, for any small values of n,k,d,q, the best codes that we know are usually better than this bound. Therefore, the question whether better nonasymptotic bounds are possible seems to be a natural one. In Section 2 we introduce a graph on the standard array of the code and relate its parameters to those of the code. Simple estimates on the number of connected components

*email: abarg@research.bell-labs.com
†email: S.Guritman@twi.tudelft.nl
‡email: J.Simonis@twi.tudelft.nl

of the graph lead to improvements of the Varshamov-Gilbert bound given in Proposition 10, Proposition 14 [7], Proposition 15, and Corollary 21 [4].

2 The Varshamov graph

Definition 1 *The code $\mathcal{C}$ is said to be* maximal *if it cannot be obtained by shortening an $[n+1, k+1, d]_q$-code.*

The following is a useful characterization of maximal codes.

Proposition 2 *The code $\mathcal{C}$ is maximal if and only if its covering radius $\rho(\mathcal{C})$ does not exceed $d-2$.*

Proof. If $\mathbf{x} \in \mathbb{F}_q^n$ has distance $\geq d-1$ to $\mathcal{C}$, then the code $\mathcal{C}'$ spanned by $(\mathbf{x}, 1)$ and $\{(\mathbf{c}, 0) \mid \mathbf{c} \in \mathcal{C}\}$ has the parameters $[n+1, k+1, d]_q$, and shortening $\mathcal{C}'$ with respect to the last coordinate position gives $\mathcal{C}$. Conversely, if $\mathcal{C}$ is obtained by shortening an $[n+1, k+1, d]_q$-code $\mathcal{C}'$, then any word in $\mathcal{C}'$ which is nonzero in the shortening position yields a vector $\mathbf{x} \in \mathbb{F}_q^n$ at distance $\geq d-1$ from $\mathcal{C}$. ■

So an $[n, k, d]_q$-code $\mathcal{C}$ with $\rho(\mathcal{C}) > d-2$ is not maximal. The following proposition, a generalization of a result by Elia [5], extends this observation to codes with arbitrary covering radius. The proof is completely analogous to that of the preceding proposition.

Proposition 3 *An $[n, k, d]_q$-code $\mathcal{C}$ with $\rho(\mathcal{C}) > \alpha$, $\alpha < d$, can be extended to an $[n+d-\alpha-1, k+1, d]_q$-code.*

Let $\mathcal{C}$ be an $[n, k, d]_q$-code.

Definition 4 *The undirected graph with the vertex set*

$$V_\alpha := \{\mathbf{x} \mid \mathbf{x} \in \mathbb{F}_q^n \text{ and } \mathrm{wt}(\mathbf{x}) \leq \alpha\}$$

and the edge set

$$E_\alpha := \{\{\mathbf{a}, \mathbf{b}\} \mid \mathbf{a}, \mathbf{b} \in V_\alpha \text{ and } \mathbf{a} - \mathbf{b} \in \mathcal{C} \setminus \{0\}\}$$

is denoted by $G_\alpha(\mathcal{C})$. The graph $G(\mathcal{C}) := G_{d-2}(\mathcal{C})$ is called the Varshamov graph *of $\mathcal{C}$.*

Obviously, the number of vertices in G_α is equal to

$$|V_\alpha| = \sum_{i=0}^{\alpha} (q-1)^i \binom{n}{i}.$$

The number of edges will turn out to be a function of the weight distribution $(A_i(\mathcal{C}))_{i=0,1,\ldots,n}$ of $\mathcal{C}$.

Let $\mathbf{x}, \mathbf{y} \in \mathbb{F}_q^n$ be any two vectors with $d(\mathbf{x}, \mathbf{y}) = w$. Then the integers

$$p_{i,j}^w := |\{\mathbf{z} \in \mathbb{F}_q^n \mid d(\mathbf{x}, \mathbf{z}) = i \text{ and } d(\mathbf{y}, \mathbf{z}) = j\}| \tag{1}$$

are known to be independent of the choice of $\mathbf{x}$ and $\mathbf{y}$. They are the so-called *intersection numbers* of the Hamming scheme $H(n, q)$. Sometimes the $p_{i,j}^w$ are

also called the *linearization coefficients* of the Hamming scheme (cf. Remark 6). See [3], Chapter 21 of [9] or Chapter 30 of [8] for a detailed description of the Hamming scheme and other association schemes. Finally, the numbers $p_{i,j}^w$ arise naturally in estimating the error probability of bounded distance decoding on the q-ary symmetric channel [1].

In the sequel we need an explicit formula for the $p_{i,j}^w$.

Proposition 5 ([1])

$$p_{i,j}^w = \sum_{\delta=0}^{\lfloor \frac{i+j-w}{2} \rfloor} (q-2)^{i+j-w-2\delta}(q-1)^\delta \binom{w}{j-\delta}\binom{j-\delta}{w-i+\delta}\binom{n-w}{\delta}. \qquad (2)$$

For $q=2$, this reduces to

$$p_{i,j}^w = \begin{cases} 0 & \text{if } i+j-w \text{ is odd,} \\ \binom{w}{(w+i-j)/2}\binom{n-w}{(i+j-w)/2} & \text{if } i+j-w \text{ is even.} \end{cases}$$

Proof. In (1), we may assume that $\mathbf{x}=0$ and $\mathrm{wt}(\mathbf{y})=w$. So $p_{i,j}^w$ counts the number of $\mathbf{z}$ with $\mathrm{wt}(\mathbf{z})=i$ and $\mathrm{wt}(\mathbf{z}-\mathbf{y})=j$. Put

$$\begin{aligned} \alpha &:= |\{u \mid z_u = y_u, z_u \neq 0\}|, \\ \beta &:= |\{u \mid z_u \neq y_u, z_u \neq 0, y_u \neq 0\}|, \\ \gamma &:= |\{u \mid z_u = 0, y_u \neq 0\}|, \\ \delta &:= |\{u \mid z_u \neq 0, y_u = 0\}|. \end{aligned}$$

Then

$$p_{i,j}^w = \sum (q-2)^\beta (q-1)^\delta \binom{w}{\alpha}\binom{w-\alpha}{\gamma}\binom{n-w}{\delta}, \qquad (3)$$

where the sum is taken over all nonnegative integer solutions of the system

$$\begin{cases} w &= \alpha+\beta+\gamma \\ i &= \alpha+\beta+\delta \\ j &= \beta+\gamma+\delta \end{cases}$$

Solve for α, β and γ, and substitute in (3). ∎

Remark 6 *Another formula for $p_{i,j}^w$ is*

$$p_{i,j}^w = q^{-n} \sum_{u=0}^{n} K_i(u) K_j(u) K_u(w),$$

with

$$K_x(y) := \sum_{m=0}^{x} (-1)^m (q-1)^{x-m} \binom{y}{m}\binom{n-y}{x-m}.$$

The $K_x(y)$ are polynomials of degree x in y, the so-called Krawtchouk polynomials. *Again, we refer to the relevant sections of [3], [9], and [8].*

Proposition 7 *The size of the edge set E_α of $G_\alpha(\mathcal{C})$ is equal to*

$$\frac{1}{2}\sum_{w=d}^{2\alpha} A_w(\mathcal{C}) \sum_{i,j=0}^{\alpha} p_{i,j}^w.$$

A more explicit formula in the binary *case is*

$$|E_\alpha| = \frac{1}{2}\sum_{w=d}^{2\alpha} A_w(\mathcal{C}) \sum_{v=0}^{2\alpha-w} \sum_{u=0}^{\lfloor v/2 \rfloor} \binom{w}{\alpha+u-v}\binom{n-w}{u}).$$

Proof. If $\mathbf{c} \in \mathcal{C}$ is a codeword of weight $w > 0$, then the set

$$X_{\mathbf{c}} := \{\{\mathbf{a}, \mathbf{b}\} \mid \mathbf{a}, \mathbf{b} \in V_\alpha \text{ and } \mathbf{a} - \mathbf{b} = \mathbf{c}\}$$

has size $\sum_{i,j=0}^{\alpha} p_{i,j}^w$ or $\frac{1}{2}\sum_{i,j=0}^{\alpha} p_{i,j}^w$, depending on whether q is odd or even. By definition, E_α is equal to

$$\bigcup_{\mathbf{c}\in\mathcal{C}\setminus\{\mathbf{o}\}} X_{\mathbf{c}}.$$

Now observe that $X_{\mathbf{c}} = X_{-\mathbf{c}}$ and that $X_{\mathbf{c}} \cap X_{\mathbf{c}'} = \emptyset$ if $\mathbf{c} \neq \pm\mathbf{c}'$. ∎

The graph $G_\alpha(\mathcal{C})$ has a very simple structure: its components are complete graphs whose vertices are the intersections of V_α with the cosets of weight $\leq \alpha$ of $\mathcal{C}$. Let $c_\alpha(\mathcal{C})$ be the number of components of $G_\alpha(\mathcal{C})$. It is also the size of the largest coclique in G. Applying Turán's theorem ([12] or [10]) we get a relation between c_α and E_α.

Proposition 8 *If $c_\alpha \leq K$, then*

$$E_\alpha \geq \left\lfloor \frac{V_\alpha}{K} \right\rfloor V_\alpha - \frac{1}{2}\left\lfloor \frac{V_\alpha}{K} \right\rfloor (\left\lfloor \frac{V_\alpha}{K} \right\rfloor + 1)K.$$

Hence the integer

$$\min\{K \mid E_\alpha \geq \left\lfloor \frac{V_\alpha}{K} \right\rfloor V_\alpha - \frac{1}{2}\left\lfloor \frac{V_\alpha}{K} \right\rfloor (\left\lfloor \frac{V_\alpha}{K} \right\rfloor + 1)K\}$$

is a lower bound for c_α.

Another useful invariant of the graph $G_\alpha(\mathcal{C})$ is $\mu_\alpha(\mathcal{C})$, the size of its largest component (i.e. clique). Turán's theorem for the complementary graph $\overline{G}_\alpha$ yields the following lower bound for $\mu_\alpha(\mathcal{C})$.

Proposition 9 *If $\mu_\alpha(\mathcal{C}) \leq M$, then*

$$E_\alpha \leq \binom{V_\alpha}{2} - \left\lfloor \frac{V_\alpha}{M} \right\rfloor V_\alpha + \frac{1}{2}\left\lfloor \frac{V_\alpha}{M} \right\rfloor (\left\lfloor \frac{V_\alpha}{M} \right\rfloor + 1)M.$$

Hence

$$\min\{M \mid E_\alpha \leq \binom{V_\alpha}{2} - \left\lfloor \frac{V_\alpha}{M} \right\rfloor V_\alpha + \frac{1}{2}\left\lfloor \frac{V_\alpha}{M} \right\rfloor (\left\lfloor \frac{V_\alpha}{M} \right\rfloor + 1)M\}$$

is a lower bound for μ_α.

However, the next proposition shows that good upper bounds for $\mu(\mathcal{C}) := \mu_{d-2}(\mathcal{C})$ are much more useful. What we really need are upper bounds for the number of words of weight up to $d-2$ in the cosets of $\mathcal{C}$.

Proposition 10 *An $[n,k,d]_q$-code $\mathcal{C}$ with*

$$\sum_{i=0}^{d-2}(q-1)^i\binom{n}{i} - 2\frac{|E(\mathcal{C})|}{\mu(\mathcal{C})} < q^{n-k}$$

is not maximal.

Proof. Consider the Varshamov graph $G(\mathcal{C})$. For $i = 1,2,\ldots,\mu := \mu(\mathcal{C})$, let ν_i denote the number of components of size i. Then

$$\begin{aligned} c(\mathcal{C}) &= \sum_{i=1}^{\mu}\nu_i, \\ |V| &= \sum_{i=1}^{\mu} i\nu_i, \text{ and} \\ |E| &= \sum_{i=1}^{\mu}\binom{i}{2}\nu_i. \end{aligned}$$

Hence $c(\mathcal{C})$ is upperbounded by the maximal value of $\sum_{i=1}^{\mu} x_i$ under the constraints

$$\begin{aligned} \sum_{i=1}^{\mu} i x_i &= |V|, \\ \sum_{i=1}^{\mu}\binom{i}{2}x_i &= |E|, \\ x_i &\geq 0, \quad 1 \leq i \leq \mu. \end{aligned}$$

We claim that

$$x_\mu = \frac{|E|}{\binom{\mu}{2}},\ x_1 = |V| - \mu x_\mu, x_i = 0 \text{ otherwise},$$

is an optimal solution. So the maximal value of $\sum_{i=1}^{\mu} x_i$ is equal to

$$|V| - \mu\frac{|E|}{\binom{\mu}{2}} + \frac{|E|}{\binom{\mu}{2}} = |V| - 2\frac{|E|}{\mu}.$$

Indeed, the dual linear program

$$\begin{aligned} |V|\,z_1 + |E|\,z_2 &\rightarrow \min \\ iz_1 + \binom{i}{2}z_2 &\geq 0, \quad 1 \leq i \leq \mu, \\ z_1, z_2 &\gneqq 0 \end{aligned}$$

has a feasible solution,

$$z_1 = 1, z_2 = -\frac{2}{\mu},$$

that produces the same value of the objective function. ■

Remark 11 *Following the idea of Proposition 3, we can generalize this result: An $[n,k,d]_q$-code $\mathcal{C}$ with*

$$\sum_{i=0}^{\alpha}(q-1)^i\binom{n}{i} - 2\frac{|E_\alpha(\mathcal{C})|}{\mu_\alpha(\mathcal{C})} < q^{n-k}$$

for some $\alpha \leq d-1$ can be extended to an $[n+d-\alpha-1, k+1, d]_q$-code.

3 Varshamov–Gilbert type results

The number $c_\alpha(\mathcal{C})$ of components of $G_\alpha(\mathcal{C})$ cannot exceed q^{n-k}, the total number of cosets. Obviously, $\mathcal{C}$ is maximal if and only if the number of components $c(\mathcal{C})$ of the Varshamov graph $G(\mathcal{C}) := G_{d-2}(\mathcal{C})$ equals q^{n-k}.

Our goal is to find upper bounds on $c(\mathcal{C})$. For if such an upper bound is smaller then q^{n-k}, then $\mathcal{C}$ is not maximal. The simplest upper bound is

$$c(\mathcal{C}) \leq |V| = \sum_{i=0}^{d-2}(q-1)^i\binom{n}{i},$$

which immediately gives the classical Varshamov-Gilbert bound.

Proposition 12 (Varshamov [13].) *If*

$$\sum_{i=0}^{d-2}(q-1)^i\binom{n}{i} < q^{n-k},$$

then no $[n,k,d]_q$-code is maximal.

Remark 13 *By Proposition 3, we can generalize this:*
If

$$\sum_{i=0}^{\alpha}(q-1)^i\binom{n}{i} < q^{n-k},$$

for some $\alpha \leq d-1$, then any $[n,k,d]_q$-code can be extended to an $[n+d-\alpha-1, k+1, d]_q$-code.
For $\alpha = d-3$ this reduces to Elia's result [5].

A general approach to find Varshamov-Gilbert type bounds would be to estimate the number of components of specific subgraphs of the Varshamov graph. We discuss two examples, basically due to Hashim [7] and Edel [4] respectively.

The first idea to consider a *forest* F in G. If $F' \supseteq F$ is a spanning forest of G, then

$$c(\mathcal{C}) = |V| - |E(F')| \leq |V| - |E(F)|, \tag{4}$$

So if we can find a forest in G with many edges, we have a good upper bound for $c(\mathcal{C})$.

An interesting example was found by Hashim. Put $t := \lfloor \frac{d-1}{2} \rfloor$.

Proposition 14 (Hashim [7].)*An $[n,k,d]_q$-code $\mathcal{C}$ with*

$$\sum_{w=d}^{d-2+t}\sum_{i=w-d+2}^{t}\binom{w}{i}A_w(\mathcal{C}) > \sum_{i=0}^{d-2}(q-1)^i\binom{n}{i} - q^{n-k}$$

is not maximal.

Proof. Consider the two disjoint subsets

$$X_1 := \{\mathbf{a} \mid 2 \le \mathrm{wt}(\mathbf{a}) \le t\}, X_2 := \{\mathbf{b} \mid t < \mathrm{wt}(\mathbf{b}) \le d-2\}$$

of $\mathbb{F}_q^n$. The bipartite graph on $\{X_1, X_2\}$ with the edge set

$$E' := \{\{\mathbf{a},\mathbf{b}\} \mid \mathbf{a}-\mathbf{b} \in \mathcal{C} \text{ and } \operatorname{supp}\mathbf{a} \cap \operatorname{supp}\mathbf{b} = \emptyset\}$$

is a forest in G because all its vertices in X_2 have degree ≤ 1. Indeed, if $\{\mathbf{a},\mathbf{b}\}, \{\mathbf{a}',\mathbf{b}\} \in E'$, then $(\mathbf{a}-\mathbf{b})-(\mathbf{a}'-\mathbf{b}) = \mathbf{a}-\mathbf{a}' \in \mathcal{C}$ and $\mathrm{wt}(\mathbf{a}-\mathbf{a}') \le 2t < d$, whence $\mathbf{a} = \mathbf{a}'$. Each word of weight w in $\mathcal{C}$ contributes $\sum_{i=w-d+2}^{t}\binom{w}{i}$ to E'. Hence

$$|E'| = \sum_{w=d}^{d-2+t}\sum_{i=w-d+2}^{t}\binom{w}{i}A_w(\mathcal{C})$$

and, by (4),

$$c(\mathcal{C}) \le \sum_{i=0}^{d-2}(q-1)^i\binom{n}{i} - \sum_{w=d}^{d-2+t}\sum_{i=w-d+2}^{t}\binom{w}{i}A_w(\mathcal{C}).$$

■

Hashim's result admits a simple improvement.

Proposition 15 *An $[n,k,d]_q$-code $\mathcal{C}$ with*

$$\sum_{w=d}^{d-2+t}\sum_{i=w-d+2}^{t}\sum_{j=w-i}^{d-2}p_{i,j}^{w}A_w(\mathcal{C}) > \sum_{i=0}^{d-2}(q-1)^i\binom{n}{i} - q^{n-k}$$

is not maximal.

Proof. Now consider the bipartite graph with the same vertex sets X_1, X_2 as in the preceding proposition, but with edge set

$$E'' := \{\{\mathbf{a},\mathbf{b}\} \mid \mathbf{a} \in X_1, \mathbf{b} \in X_2 \text{ and } \mathbf{a}-\mathbf{b} \in \mathcal{C}\}.$$

By the same reasoning, this bipartite graph is seen to be a forest. Its number of edges is

$$\sum_{w=d}^{d-2+t}\sum_{i=w-d+2}^{t}\sum_{j=w-i}^{d-2}p_{i,j}^{w}A_w(\mathcal{C}). \tag{5}$$

Again we apply (4). ■

Remark 16 *In the* binary *case, Expression (5) takes the form*

$$\sum_{\delta=0}^{t-2}\sum_{v=0}^{t-2-\delta}\sum_{u=0}^{\lfloor v/2\rfloor}\binom{d+\delta}{d-2+u-v}\binom{n-d-\delta}{u}A_{d+\delta}(\mathcal{C}). \tag{6}$$

Remark 17 *Proposition 3 enables us to generalize Proposition 15 in the following way:*
An $[n,k,d]_q$-code $\mathcal{C}$ with

$$\sum_{w=d}^{\alpha+t}\sum_{i=w-\alpha}^{t}\sum_{j=w-i}^{\alpha}p_{i,j}^{w}A_w(\mathcal{C}) > \sum_{i=0}^{\alpha}(q-1)^i\binom{n}{i}-q^{n-k}$$

for some $\alpha \le d-1$ can be extended to an $[n+d-\alpha-1,k+1,d]_q$-code.

Now we come to the second idea. First we fix some notation. Let T be a subset of the coordinate index set $\{1,2,\dots,n\}$. The *projection* of an $\mathbf{x}\in\mathbb{F}_q^n$ to T is denoted by $\mathbf{x}_T$ and the code obtained by $\mathcal{C}$ through *shortening* with respect to T by $\mathcal{C}^{\overline{T}}$. (Here $\overline{T}$ denotes the complement of T in $\{1,2,\dots,n\}$.) We define $s_\alpha(\mathcal{C}) := c_\alpha(\mathcal{C}) - c_{\alpha-1}(\mathcal{C})$. Note that

$$s_\alpha(\mathcal{C}) \le (q-1)^\alpha\binom{n}{\alpha}, \tag{7}$$

with equality for $\alpha \le t = \lfloor\frac{d-1}{2}\rfloor$.
We need two obvious lemmas.

Lemma 18 *If $\mathcal{C} := \mathcal{C}_1 \oplus \mathcal{C}_2$, then*

$$c_\alpha(\mathcal{C}) = \sum_{j=0}^{\alpha}s_j(\mathcal{C}_1)c_{\alpha-j}(\mathcal{C}_2).$$

Lemma 19 *If $\mathcal{D}\subseteq\mathcal{C}$, then $c_\alpha(\mathcal{D}) \ge c_\alpha(\mathcal{C})$.*

The following relation between the values of c_α for $\mathcal{C}$, $\mathcal{C}^T$ and $\mathcal{C}^{\overline{T}}$ creates a possibility of induction.

Proposition 20

$$c_\alpha(\mathcal{C}) \le \sum_{j=0}^{\min(\alpha,m)}s_j(\mathcal{C}^T)c_{\alpha-j}(\mathcal{C}^{\overline{T}}). \tag{8}$$

Proof. Note that $\mathcal{C}^T\oplus\mathcal{C}^{\overline{T}}$ is a subcode of $\mathcal{C}$ and apply the preceding lemmas. In fact, the right hand side counts the components of the subgraph G' of $G(\mathcal{C})$ with the same vertex set V, but with the edge set

$$E' := \{\{\mathbf{a},\mathbf{b}\} \mid \mathbf{a}_T - \mathbf{b}_T \in \mathcal{C}^T \wedge \mathbf{a}_{\overline{T}} - \mathbf{b}_{\overline{T}} \in \mathcal{C}^{\overline{T}}\}.$$

■

Corollary 21 (Edel [4]) *From (7) we infer that*

$$c_\alpha(\mathcal{C}) \leq \sum_{j=0}^{\min(\alpha,m)} (q-1)^j \binom{n}{j} c_{\alpha-j}(\mathcal{C}^{\overline{T}}).$$

We can embed any $[n,k,d]_q$-code $\mathcal{C}$ in an $[n+1,k,d]_q$-code $\mathcal{C}'$ by adding one zero coordinate to each codeword. Let us call this construction *trivial lengthening*. Corollary 21, with $m=1$, immediately gives the bound

$$c_\alpha(\mathcal{C}') \leq c_\alpha(\mathcal{C}) + (q-1)c_{\alpha-1}(\mathcal{C}). \tag{9}$$

If an $[n,k,d]_q$-code $\mathcal{C}$ is not maximal, we can embed it in an $[n+1,k+1,d]_q$-code $\mathcal{C}'$. Let us call this a *Varshamov step*. The component sizes $c_\alpha(\mathcal{C}')$ of the new code $\mathcal{C}'$ satisfy the bounds

$$c_\alpha(\mathcal{C}') \leq c_\alpha(\mathcal{C}) + (q-1)c_{\alpha-1}(\mathcal{C}). \tag{10}$$

Indeed, let $n+1$ be the extra coordinate index in $\mathcal{C}'$. We can split the vertex set $V_\alpha(\mathcal{C}')$ of $\mathcal{C}'$ into the q subsets

$$W_i := \{\mathbf{u} \in V_\alpha(\mathcal{C}') \mid u_{n+1} = i\}.$$

Then the restriction of $G_\alpha(\mathcal{C}')$ to W_i is isomorphic to $G_\alpha(\mathcal{C})$ if $i=0$, and isomorphic to $G_{\alpha-1}(\mathcal{C})$ if $i \neq 0$.

Now Edel's idea in [4] is as follows. Start with an $[n_i,k_i,d]_q$-code $\mathcal{C}_0$ and build a sequence of $[n_i,k_i,d]_q$-codes $\mathcal{C}_i$, $i=1,2,\ldots$, of increasing length by taking Varshamov steps when our information on $c_{d-2}(\mathcal{C}_i)$ tells us that this possible. If not, apply trivial lengthening until a Varshamov step again is possible. At each step, estimate the $c_\alpha(\mathcal{C}_i)$ using (9), (10) and the trivial bound $c_\alpha(\mathcal{C}_i) \leq q^{n_i-k_i}$. By this simple method, Edel improved quite a few lower bounds in Brouwer's tables [2] on bounds for optimal linear ternary and quaternary linear codes. Without doubt, the method will work for larger alphabets as well.

References

[1] R. E. Blahut, *Theory and Practice of Error Control Codes,* Addison-Wesley Publishing Co., Reading, Mass., 1983.

[2] A. E. Brouwer, Bounds on the size of linear codes. To appear in *Handbook of coding theory* (V. Pless and W. Cary Huffman, Eds.)., Elsevier Science, 1998. Online version of the tables: `http://www.win.tue.nl/math/dw/voorlincod.html`.

[3] P. Delsarte, An Algebraic Approach to the Association Schemes of Coding Theory, Philips Res. Rep. Suppl. 10, 1973.

[4] Y. Edel, Eine Verallgemeinerung von BCH-Codes. PhD Thesis, Univ. Heidelberg, 1996.

[5] M. Elia, Some results on the existence of binary linear codes. *IEEE Trans. Inform. Theory* 29: 933–934 (1983).

[6] E. N. Gilbert, A comparison of signalling alphabets. *Bell Syst. Tech. J.* 31: 504–522 (1952).

[7] A. A. Hashim, Improvement on Varshamov-Gilbert lower bound on minimum Hamming distance of linear codes. *Proc. Inst. Elec. Engrs.* 125: 104–106 (1978).

[8] J. H. van Lint, R. M. Wilson, *A Course in Combinatorics*, Cambridge University Press, Cambridge, 1992.

[9] F. J. MacWilliams, N. J. A. Sloane, *The Theory of Error-Correcting Codes*, 2nd reprint, North-Holland Mathematical Library, Vol. 16, North-Holland Publishing Co., Amsterdam - New York - Oxford, 1983.

[10] O. Ore, *Theory of Graphs*, Third printing, with corrections, American Mathematical Society Colloquium Publications, Vol. XXXVIII American Mathematical Society, Providence, R.I., 1967.

[11] M. A. Tsfasman, S. G. Vladut, *Algebraic-Geometric Codes*, Translated from the Russian by the authors. Mathematics and its Applications (Soviet Series), 58. Kluwer Academic Publishers Group, Dordrecht, 1991.

[12] P. Turán, Eine Extremalaufgabe aus der Graphentheorie. (Hungarian. German summary.) *Mat. Fiz. Lapok* 48: 436–452 (1941).

[13] R.R. Varshamov, Estimate of the number of signals in error correcting codes. *Dokl. Acad. Nauk SSSR,* 117: 739–741 (1957).

Recursive MDS-codes

Elena Couselo, Santos Gonzalez,
Victor Markov, Alexandr Nechaev
University of Oviedo (Spain)
Center of New Informational Technologies
of Moscow State University *
e-mail: nechaev@cnit.chem.msu.su

Introduction

A code $\mathcal{K} \subset \Omega^n$ in an alphabet Ω of q elements is called *k-recursive*, $1 \leq k < n$, if there exists a function $f : \Omega^k \to \Omega$ such that $\mathcal{K}$ consists of all the rows $u(\overline{0, n-1}) = (u(0), \ldots, u(n-1)) \in \Omega^n$ with the property

$$u(i+k) = f(u(\overline{i, i+k-1})), \quad i \in \overline{0, n-k-1}.$$

In other words $\mathcal{K}$ is the set of all output n-sequences of a feedback shift register with a feedback function f. We denote $\mathcal{K} = \mathcal{K}(n, f)$ and investigate the existence problem for MDS-codes of such type, i.e. recursive $[n, k, n-k+1]_q$-codes. In connection with this, the following set of parameters is interesting.

$n(k, q)$ – maximum of lengths of MDS codes $\mathcal{K}$ of *(combinatorial) dimension k*

($|\mathcal{K}| = q^k$) in alphabet Ω of cardinality q.

$n^r(k, q)$ – maximum of lengths of k-recursive MDS codes of the same type.

$l(k, q)$ – maximum of lengths of MDS codes $\mathcal{K}$ of (combinatorial) dimension k which are linear over an abelian group $(\Omega, +)$ for some operation $+$

*The last two authors thank the University of Oviedo for the hospitality and support. Their work was partially supported also by RFBR grants 96-01-00-627 and 96-01-00-931

(i.e. $\mathcal{K}$ is a subgroup of an Abelian group Ω^n and $|\mathcal{K}| = q^k$, where $q = |\Omega|$ and $n \in \mathbb{N}$). We also shall call such codes *linear in broad sense.*

$l^r(k,q)$ – the analog of $l(k,q)$ for recursive codes.

For the primary (the power of a prime) numbers q we also study

$m(k,q)$ – the analog of $l(k,q)$ only for the codes which are linear over the field $\mathbb{F}_q$ (*linear in narrow sense*).

$m^r(k,q)$ – the analog of $m(k,q)$ for the recursive codes.

Moreover, we shall call the above function $f(\mathbf{x})$ *idempotent* if it satisfies the identity $f(x,\dots,x) = x$. The last property is equivalent to the condition that all "constants" $(a,\dots,a)$ belong to $\mathcal{K}(n,f)$. So in addition to the above six we can introduce three new parameters $n^{ir}(k,q), l^{ir}(k,q)$ and $m^{ir}(k,q)$ (only for primary q) with evident definitions. ("ir" means "idempotent recursive").

So we have the following matrix of parameters

$$M(k,q) = \begin{pmatrix} [m^{\mathrm{ir}}(k,q) & m^{\mathrm{r}}(k,q) & m(k,q)] \\ l^{\mathrm{ir}}(k,q) & l^{\mathrm{r}}(k,q) & l(k,q) \\ n^{\mathrm{ir}}(k,q) & n^{\mathrm{r}}(k,q) & n(k,q) \end{pmatrix}$$

whose entries do not decrease from left to right and from up to down. Naturally the first row of this matrix (shown in brackets) is present only when q is primary. It is interesting to estimate and to compare the entries of $M(k,q)$ for various values of k and q. In what follows the equality $x^y(k,q) = k$ means that the corresponding code does not exist. A standard argument gives the following source of estimations for the entries of $M(k,q)$:

0.1. Proposition. *If* $x \in \{l,n\}$, $y \in \{\emptyset, r, ir\}$ *and* $k, q_1, q_2 \in \overline{2,\infty}$ *then*

$$x^y(k, q_1q_2) \geq min\{x^y(k,q_1), x^y(k,q_2)\}.$$

1 Results for $k = 2$

It is well-known that $n(2,q) = 2 + N(q)$ where $N(q)$ is the maximal number of mutually orthogonal Latin $q \times q$-squares. The detailed reviews of the results on $N(q)$ are given in [3, 4, 5]. We cite here only the following general conclusion:

1.1. Theorem. *Let* $q \in \mathbb{N}, q > 1$. *Then:*
(a) $N(q) \leq q - 1$;

(b) if q is primary, then $N(q) = q - 1$;
(c) $N(q_1 q_2) \geq \min\{N(q_1), N(q_2)\}$, in particular, if $q = q_1 \cdot \ldots \cdot q_t$ is a canonical factorization of q then $N(q) \geq \min\{q_1 - 1, \ldots, q_t - 1\}$;
(d) $N(q) \geq 2$ if and only if $q \notin \{2, 6\}$;
(e) $N(q) \geq 3$ if $q \notin \{2, 3, 6, 10\}$.
(f) $N(q) \geq q^{\frac{10}{143}} - 2$,

So the inequality $n(2,q) \geq 4$ is equivalent to the existence of a pair of orthogonal Latin squares of order q. The inequality $n^r(2,q) \geq 4$ means that, in addition, the second one must be recursive derivative of the first[1].

For small values of q we have

$$M(2,2) = \begin{pmatrix} 2 & 3 & 3 \\ 2 & 3 & 3 \\ 2 & 3 & 3 \end{pmatrix}, \; M(2,3) = \begin{pmatrix} 3 & 4 & 4 \\ 3 & 4 & 4 \\ 3 & 4 & 4 \end{pmatrix}, \; M(2,4) = \begin{pmatrix} 3 & 5 & 5 \\ 3 & 5 & 5 \\ 3 & 5 & 5 \end{pmatrix},$$

$$M(2,5) = \begin{pmatrix} 5 & 6 & 6 \\ 5 & 6 & 6 \\ 5 & 6 & 6 \end{pmatrix}, \quad M(2,7) = \begin{pmatrix} 7 & 8 & 8 \\ 7 & 8 & 8 \\ ? & 8 & 8 \end{pmatrix},$$

and of course (G.Tarry's solution [7] of L.Euler's problem of 36 officers)

$$M(2,6) = \begin{pmatrix} 2 & 3 & 3 \\ 3 & 3 & 3 \end{pmatrix}$$

1.2. Proposition. *If q is primary then*

$$m^r(2,q) = n^r(2,q) = n(2,q) = q + 1;$$

$$m^{ir}(2,q) = l^{ir}(2,q) = \begin{cases} q & \text{if } q \text{ is a prime;} \\ q-1 & \text{if } q \text{ is not a prime} \\ & \text{(V.Abashin (private communication)).} \end{cases}$$

The following two theorems are the main results of this section.

1.3. Theorem. *For arbitrary $q > 2$, except $q = 6$ and possibly $q \in \{14, 18, 26, 42\}$, $n^r(2,q) \geq 4$.*

Really, the last inequality may be sharpened for many values of q. Some of the stronger estimations are easily deduced from Propositions 0.1, 1.2. We call them *standard*. Other estimations we have obtained are presented in

1.4. Theorem. *The following nonstandard lower estimations are valid:*
$n^r(2,q) \geq 8$ *for* $q = 80$.
$n^r(2,q) \geq 7$ *for* $q \in \{50, 57, 58, 65, 70, 78, 84, 85, 86, 92, 94, 95, 96, 98\}$.
$n^r(2,q) \geq 6$ *for* $q \in \{54, 62, 66, 68, 69, 74, 75, 76, 82, 87, 90, 93\}$.
$n^r(2,q) \geq 5$ *for* $q \in \{21, 24, 39, 60\}$.
$n^{ir}(2,q) \geq 7$ *for* $q \in \{50, 57, 58, 65, 70, 78, 80, 84, 85, 86, 92, 94, 95, 96, 98\}$.
$n^{ir}(2,q) \geq 5$ *for* $q \in \{21, 54, 62, 66, 68, 69, 74, 75, 76, 82, 87, 90, 93\}$.

We present here the Table 1 which allows to compare the lower bounds

	0	1	2	3	4	5	6	7	8	9
0		∞ ∞	3 * 3	4 * 4	5 * 5	6 * 6	3 3	8 * 8	9 * 9	10 * 10
1	$\underline{4}$ 4	12 * 12	4 7	14 * 14	? 5	4 6	17 * 17	18 * 18	? 5	20 * 20
2	5 6	$\underline{5}$ 6	$\underline{4}$ 5	24 * 24	$\underline{5}$ 6	26 * 26	? 5	28 * 28	5 5	30 * 30
3	$\underline{4}$ 5	32 * 32	33 * 33	4 6	$\underline{4}$ 5	6 6	$\underline{4}$ 6	38 * 38	$\underline{4}$ 5	5 6
4	$\underline{4}$ 6	42 * 42	? 5	44 * 44	5 5	$\underline{4}$ 5	$\underline{4}$ 6	48 * 48	$\underline{4}$ 6	50 * 50
5	$\underline{7}$ 8	4 6	5 5	54 * 54	$\underline{6}$ 6	6 7	8 9	$\underline{7}$ 9	$\underline{7}$ 7	60 * 60
6	$\underline{5}$ 6	62 * 62	$\underline{6}$ 6	8 8	65 * 65	$\underline{7}$ 9	$\underline{6}$ 7	68 * 68	$\underline{6}$ 7	$\underline{6}$ 8
7	$\underline{7}$ 8	72 * 72	9 9	74 * 74	$\underline{6}$ 7	$\underline{6}$ 7	$\underline{6}$ 7	8 8	$\underline{7}$ 8	80 * 80
8	$\underline{8}$ 9	82 * 82	$\underline{6}$ 10	84 * 84	$\underline{7}$ 8	$\underline{7}$ 8	$\underline{7}$ 8	$\underline{6}$ 8	9 9	90 * 90
9	$\underline{6}$ 6	$\underline{8}$ 8	$\underline{7}$ 7	$\underline{6}$ 6	$\underline{7}$ 7	$\underline{7}$ 7	$\underline{7}$ 7	98 * 98	$\underline{7}$ 7	10 * 10

Table 1: Table of values of $n^r(2,q)$ and $n(2,q)$ for $q < 100$.

of $n^r(2,q)$ obtained by us (value in the upper left corner of each cell) and the known lower bounds $n(2,q)$ (value in the lower right corner cell) taken from [3, Figure 7.1, Chapter 5]. The cell in row i and column j corresponds to the value $q = 10i + j$. The nonstandard estimations of $n^r(2,q)$ are underlined. The cells marked in the upper right corner by $*$ correspond to the primary numbers.

2 Results for $k > 2$

The following estimations are known.

2.1. Theorem. [6, 5]
(a) If $q \leq k$ *then* $n(k,q) = k+1$.
(b) If $k \leq q$ *and* q *is even then* $n(k,q) \leq q+k-1$.
(c) If $3 \leq k \leq q$ *and* q *is odd then* $n(k,q) \leq q+k-2$.
(d) If $k \in \overline{1,k+1}$ *and* q *is primary (i.e. a power of a prime) then* $m^r(k,q) \geq q+1$.
(e) $n(3,q) = q+1$ *for primary odd* q.
(f) $n(3,q) = n(q-1,q) = q+2$, *for primary even* q.

We have the following "recursive version" of these results.

2.2. Proposition. *If* $q \leq k$ *then* $l^r(k,q) = n(k,q) = k+1$ *and, for primary* q, $m^r(k,q) = k+1$.

The well-known **conjecture of McWilliams and Sloane** [6] states that for any primary q,

$$m(k,q) = \begin{cases} q+1 & \text{for } 2 \leq k \leq q \\ k+1 & \text{for } q < k \end{cases}$$

except the case

$$m(3,q) = m(q-1,q) = q+2 \text{ for even } q.$$

For $k = 3$ we have (using PC calculation):

2.3. Proposition. *For every primary* $q \in \overline{4,128}$, *the following equality takes place:*

$$m^r(3,q) = q+1.$$

In each of these cases, the number of linear recursive $[q+1,3,q-1]$*-codes is equal to*

$$\frac{1}{2}\varphi(q+1)(q-1).$$

The last statement means that any code obtained in Proposition 2.3 can be constructed from some fixed linear cyclic code indicated in [6, ch.11, Theorem 9] by substitution of each word $(\alpha_0,\dots,\alpha_q)$ with the word $(\beta_0,\dots,\beta_q)$, where $\beta_i = a^i\alpha_{i\delta}$ for some fixed $a \in \mathbb{F}_q$ and $\delta \in \overline{1,q}$, $(\delta,q)=1$.

So the following conjecture is a "recursive analog" of McWilliams and Sloane's conjecture.

2.4. Conjecture. *Proposition 2.3 is valid for any primary* $q \geq 4$.

For the case $k=3$, $q=4$ we have the following full description of parameters $m^r(3,4), l^r(3,4), n^r(3,4)$.

2.5. Proposition.

$$m^r(3,4) = 5 < l^r(3,4) = n^r(3,4) = m(3,4) = n(3,4) = 6.$$

□ In view of 2.3 it is enough to prove only that $l^r(3,4)=6$. The computer search gives exactly 24 recursive $[6,3,4]_4$-codes $\mathcal{K}(6, f(x_1,x_2,x_3))$ over the field $\mathbb{F}_4 = \{0,1,\alpha,\alpha+1\}$. One of them is linear in the broad sense with recursion loo

$$f(x_1,x_2,x_3) = \alpha x_1^2 + \alpha x_2 + x_3^2.$$

We call it the *Asturian code.*□

The example of the Asturian code gives some important theoretical corollaries: (1) there exist linear in the broad sense recursive codes that are better than any linear in the classical sense recursive code, and (2) for some of the best known linear in classical sense but non recursive codes, there exist linear in the broad sense recursive codes with the same parameters. However, the Asturian code may be an exclusive example because the PC calculations give

$$l^r(3,q) = q+1 \quad \text{for} \quad q \in \{8,16\}.$$

References

[1] Couselo E., Gonzalez S., Markov V., Nechaev A. "Recursive MDS-codes and recursively differentiable quasigroups", Diskr. Math. and Appl., V.8, No. 3, 217–247, VSP, 1998.

[2] Couselo E., Gonzalez S., Markov V., Nechaev A. "Recursive MDS-codes and recursively differentiable quasigroups-II", Diskr. Math. and Appl., *to appear.*

[3] J.Dénes & A.D.Keedwell, "Latin Squares and their Applications". Akadémiai Kiadó, Budapest; Academic Press, New York; English Universities Press, London, 1974.

[4] J.Dénes & A.D.Keedwell, "Latin squares. New developments in the theory and applications". Annals of Discrete Mathematics, 46. North-Holland, Amsterdam, 1991.

[5] Heise W. & Quattrocci P. "Informations- und codierungstheorie", Springer, 1995, Berlin–Heidelberg.

[6] MacWilliams F.J. & Sloane N.J.A. "The theory of Error-Correcting Codes". Elsevier Science Publishers, B.V., 1988. North Holland Mathematical Library, Vol. 16.

[7] Tarry G. Le problème des 36 officiers, C.r. Assoc. France Av. Sci., P. 2, V. 29, 170–203, 1900.

A Construction of Systematic Authentication Codes based on Error-Correcting Codes

Sheng-bo Xu Henk van Tilborg
Department of Mathematics and Computing Science
Eindhoven University of Technology, P.O. Box 513
5600 MB Eindhoven, the Netherlands
E-mail: sbxu@win.tue.nl; henkvt@win.tue.nl

Abstract

A new method is presented to construct systematic authentication codes from error-correcting codes. It makes use of certain affine transformations. The basic parameters of the new authentication codes are analyzed and compared with existing constructions.

Keywords: Error-Correcting Codes, Authentication Codes

1 Introduction

The general theory of unconditional authentication has been developed by Simmons [9]. In this model, there are three participants: a sender, a receiver and an opponent. This model is usually called the non-arbitrator authentication model and will be explained briefly here. These authentication codes are called A-codes for short.

The sender wants to send some information, which is usually called a *source state* s, to the receiver by means of a public communications channel. The opponent tries to fool the receiver into accepting a fraudulent message as a genuine one. To defeat the attacks from the opponent, the transmitter

encodes s into a message m according to a secret *encoding rule* e shared with the receiver.

Here, we only discuss *systematic* (also called Carthesian) A-codes: the transmitted message consists of a concatenation of the source state s with a so-called *authenticator* t.

Let $\mathcal{S}$ be a finite set of source states and let $\mathcal{T}$ be a set of authentication values or authenticators. Further, let $\mathcal{E}$ be a finite set of encoding rules $e : \mathcal{S} \to \mathcal{T}$. This set will also be called the key space of the system. In the context of systematic A-codes, the set $\mathcal{M}$ of possible messages consists of all $m = (s, e(s))$. It is a subset of $\mathcal{S} \times \mathcal{T}$, but will mostly be equal to it. The tail $e(s)$ acts as authenticator t for s.

The receiver verifies the authenticity of a received message (s, t) by checking that $t = e(s)$, where e is the (secret) encoding rule shared with the presumed sender.

Suppose that the opponent has the ability to introduce new messages into the channel and/or to modify messages transmitted by the sender. Also, we shall assume that the opponent knows all the details of the authentication scheme except for the secret encoding rule.

When the opponent transmits a message $m' = (s', t') \in \mathcal{S} \times \mathcal{T}$ over the channel in an attempt to impersonate the sender, this is called an *impersonation attack*. When the opponent intercepts a transmitted message $m = (s, t)$ from the legitimate sender and replaces it by another message $m' = (s', t')$, with $s' \neq s$, this is called a *substitution attack*. In either case, it is the goal of the opponent to have m' accepted as an authentic message by the receiver. That is, if e is the encoding rule being shared by sender and receiver (which is not known to the opponent), then the opponent is hoping that $t' = e(s')$.

An A-code is completely described by the triple $(\mathcal{S}, \mathcal{M}, \mathcal{E})$. The code is said to have parameters $(|\mathcal{S}|, |\mathcal{M}|, |\mathcal{E}|)$. In addition, the probabilities of impersonation and substitution attacks are important parameters of A-codes.

Let P_I and P_S denote the maximum probability of a successful impersonation resp. substitution and let $P_D = \max\{P_I, P_S\}$ denote the probability of successful deception. More precisely:

$$P_I = \max_{(s,t)\in\mathcal{S}\times\mathcal{T}} \frac{|\{e \in \mathcal{E} \mid e(s) = t\}|}{|\mathcal{E}|} \tag{1}$$

$$P_S = \max_{(s,t),(s',t')\in\mathcal{S}\times\mathcal{T}, s\neq s'} \frac{|\{e \in \mathcal{E} \mid e(s) = t, e(s') = t'\}|}{|\{e \in \mathcal{E} \mid e(s) = t\}|} \tag{2}$$

Let $H(X|Y)$ and $I(X;Y)$ denote the usual (conditional) entropy function resp. the mutual information function (see [7] for definitions). Then the following inequalities are known [1, 8, 9] :

$$P_I \geq 2^{-I(M;E)}, \tag{3}$$
$$P_S \geq 2^{-H(E|M)}, \tag{4}$$
$$P_D \geq \frac{1}{\sqrt{|\mathcal{E}|}}, \tag{5}$$

where M and E denote random variables defined on $\mathcal{M}$ and $\mathcal{E}$.
An A-code satisfying

$$P_I = P_S = \frac{1}{\sqrt{|\mathcal{E}|}} \tag{6}$$

is commonly called a *perfect* A-code (see [8]).

¿From [2] we quote the following theorem (see also [4]).

Theorem 1.1 *A necessary condition for an authentication code to be perfect is that*

$$|\mathcal{S}| \leq \sqrt{|\mathcal{E}|} + 1. \tag{7}$$

In other words, a sender who wants to transmit from a large source space by means of a perfect A-code has to share a key with the receiver from a much larger key space. Hence, constructions of perfect A-codes are often not very practical. It is very important to find A-codes which contain as many source states as possible, given the size of the key space.

The paper is organized as followed. Section 2 gives an overview of some A-codes constructed by error-correcting codes (from now on shortened to EC-codes). Section 3 introduces constructions of systematic A-codes based on affine transformations. Section 4 presents a new construction method of A-codes, which is based on affine transformations and EC-codes. It also analyzes the basic parameters of these code. Finally, in Section 5, we will compare our construction with the q-twisted construction described in [3, 5].

2 Constructions of A-Codes from EC-Codes

M.N.Wegman and J.L.Carter [16] observed that by relaxing requirement slightly (by not requiring that the probability of deception is at its theoretical minimum), one can often reduce the number of encoding rules significantly, at least in the case where $|\mathcal{S}| \gg |\mathcal{T}|$. D.R.Stinson studied implementations of the above idea by means of universal hash functions [12]. In the following, we will give a small discussion of some works by means of EC-codes.

In [8], G.J.Simmons pointed out that *coding theory and authentication theory are dual theories: one is concerned with clustering the most likely alternations as closely about the original code as possible and the other with spreading the optimal (to the opponent) alterations as uniformly as possibly over M.* But the essence of this duality has not been revealed.

In the light of the above statement, many researchers began to investigate possible connections between A-codes and EC-codes. In [3, 5], Kabatianskii, e.o., established a particular kind of connection between A-codes and EC-codes. Through this connection, they showed that some systematic A-codes can be constructed from EC-codes and vice versa. The former is used to show that if P_S exceeds P_I by an arbitrarily small positive amount, the number of source states grows exponentially with the number of the keys but if $P_S = P_I$ it will grow only linearly. Also, they derived some other valuable results about authentication codes.

They introduced a particular subclass $\mathcal{C}^{(1)}$ of q-ary EC-codes, namely block codes satisfying the additional property:

$$\underline{c} \in C \Rightarrow \underline{c} + \lambda \underline{1} \in C, \quad \forall \lambda \in GF(q). \tag{8}$$

Through the so-called q-twisted construction, they can map a q-ary EC-code $C \in \mathcal{C}^{(1)}$ with parameters $(n, |C|, d)$ into a systematic A-code with parameters $(|C|/q, |C|, nq)$, in which $P_I = 1/q$ and $P_S = 1 - d/n$. Note that in such a A-code, the number of source states is $1/q$-th of the number of codewords in C. Also, C must satisfy property (8).

Safavi-Naini and Seberry [10] constructed authentication codes by making using the encoding rules of EC-codes directly as encoding rules for A-codes.

In the following, we will present a construction of systematic A-codes based on EC-codes and affine transformations.

3 A Construction of Systematic A-Codes Based on Affine Transformations

We firstly introduce a construction for systematic A-codes by means of affine transformations. It does not make use of EC-codes, but will serve as an introduction to our later constructions. This construction is an extension of *Example* 10.1 in [13] and it is similar to the modified GMS-scheme in [14], which is constructed from projective plane $PG(2, F_q)$.

Construction 3.1 *Let $GF(q)$ be a finite field with q elements. Let $\mathcal{S} = \mathcal{T} = GF(q)$. As set of encoding rules $\mathcal{E}$ take the set F of all mappings $f_{a,b} : GF(q) \to GF(q)$, $a, b \in GF(q)$, defined by*

$$f_{a,b}(s) = as + b, \ s \in \mathcal{S}. \tag{9}$$

The message set $\mathcal{M}$, as always, consists of all pairs (s, t) with $t = f_{a,b}(s)$. So, $\mathcal{M} = (GF(q))^2$.

The A-code constructed above is systematic and has parameters (q, q^2, q^2).

Let us now calculate the probabilities of impersonation and substitution.

First, we consider P_I. Independent of the choice of s and t in $GF(q)$ we have

$$|\{e \in \mathcal{E} \mid e(s) = t\}| \ = \ |\{(a, b) \in (GF(q))^2 \mid as + b = t\}| \ = \ q. \tag{10}$$

Indeed, for any choice of a there is a unique value of b, satisfying $as+b = t$. Since $|\mathcal{E}| = q^2$, we get

$$P_I = 1/q. \tag{11}$$

The computation of P_S is slightly more involved. Suppose that the opponent has observed message (s, t), and he want to replace it by (s', t') with $s' \neq s$. Then

$$\begin{aligned} |\{e \in \mathcal{E} \mid e(s) = z, e(s') = z'\}| = \ & |\{(a, b) \in (GF(q))^2 \mid as + b = z, \\ & as' + b = z')\}| \\ = 1. & \end{aligned}$$

So, by (10) we get

$$P_S = 1/q. \tag{12}$$

The above means that A-codes from Construction 3.1 satisfy (6), so they are perfect.

4 A Construction of Systematic A-Codes Based on Affine Transformations and EC-codes

The q-twisted construction described in [3, 5] makes use of q-ary EC-codes satisfying property (8). As a result of the construction, the number of source states in the resulting A-code is $1/q$-th of the cardinality of the original EC-code.

In the next construction no extra assumption will be made on the EC-code being used. Let $\pi_i : (GF(q))^n \rightarrow GF(q)$ denote the projection of a q-ary vector of length n onto its i-th coordinate, $1 \leq i \leq n$. Also, let $N = \{1, 2, \ldots, n\}$.

Construction 4.1 *Let C be a q-ary EC-code with parameters $(n, |C|, d)$. To define an A-code, take $\mathcal{S} = C$ as set of source states and take $\mathcal{T} = GF(q)$. So, the message set $\mathcal{M}$ is contained $(GF(q))^{n+1}$.*

The set $\mathcal{E}$ of encoding rules consists of all composite mappings $f_{a,b} \circ \pi_i$, $1 \leq i \leq n$, $a, b \in GF(q)$ (see (9) for the definition of $f_{a,b}$), so

$$(f_{a,b} \circ \pi_i)(\underline{c}) \;=\; f_{a,b}(\pi_i(\underline{c})) \;=\; ac_i + b.$$

Theorem 4.2 *Given any q-ary EC-code with parameters $(n, |C|, d)$, Construction 4.1 will yield a systematic A-codes which has $|C|$ source states, nq^2 encoding rules, and $|C|q$ messages.*

Also, the probability of success for an impersonation attack is given by $P_I = 1/q$, and the probability of success for substitution attack is given by $P_S \leq 1 - (q-1)d/qn$.

Proof: The parameters of the constructed A-codes can easily be verified. In the following, we will discuss the probability of success for an impersonation attack and a substitution attack.

To succeed in an impersonation attack, the opponent must guess the correct authentication symbol t corresponding to a source state $\underline{c}$ chosen by him. Again the probability of success turns out to be independent of the choice of $\underline{c}$ and t. Indeed,

$$\begin{aligned} |\{e \in \mathcal{E} \mid e(f(\underline{c})) = t\}| &= |\{(a,b,i) \in (GF(q))^2 \times N \mid f_{a,b}(\pi_i(\underline{c})) = t\}| \\ &= |\{(a,b,i) \in (GF(q))^2 \times N \mid ac_i + b = t\}| \\ &= nq. \end{aligned} \tag{13}$$

Indeed, for any choice of $i \in N$ and $a \in GF(q)$ there is a unique $b \in GF(q)$ such that $ac_i + b = t$. It follows from Equation (1) that $P_I = nq/nq^2 = 1/q$.

To calculate the probability of success for a substitution attack, we consider an intercepted message $(\underline{c}, t)$ that the opponent wants to replace by $(\underline{c}', t')$ with $\underline{c} \neq \underline{c}'$. Following (2), we calculate

$$\begin{aligned} |\{e \in \mathcal{E} \mid e(\underline{c}) = t, e(\underline{c}') = t'\}| = &|\{(a,b,i) \in (GF(q))^2 \times N \mid f_{a,b}(\pi_i(\underline{c})) = t, \\ &\qquad f_{a,b}(\pi_i(\underline{c}')) = t'\}| \\ = &|\{(a,b,i) \in (GF(q))^2 \times N \mid ac_i + b = t, \\ &\qquad ac_i' + b = t'\}|. \end{aligned}$$

Let us look more closely at the set of simultaneous equations $ac_i + b = t$ and $ac_i' + b = t'$. We distinguish two cases, namely $t = t'$ and $t \neq t'$. Let us first consider the case that $t = t'$. One has q solutions (a, b) on those coordinates i where $\underline{c}$ and $\underline{c}'$ agree (choose a arbitrary and solve for b). On the coordinates where $\underline{c}$ and $\underline{c}'$ differ, there is a unique solution (a, b), namely $a = 0$ and $b = t = t'$. Therefore, if $t = t'$, the right hand side above is equal to $qn - (q-1)d_H(\underline{c}, \underline{c}')$.

If $t \neq t'$, we get no solutions on the coordinates where $\underline{c}$ and $\underline{c}'$ agree and one solution on each coordinate where $\underline{c}$ and $\underline{c}'$ disagree. So, in this case, the right hand side above is equal to $d_H(\underline{c}, \underline{c}')$.

In the definition of P_S, we need to consider the maximum over all pairs (s, t). Since $d_H(\underline{c}, \underline{c}') \leq n$ we have $qn - (q-1)d_H(\underline{c}, \underline{c}') \geq d_H(\underline{c}, \underline{c}')$, we may conclude that in both cases:

$$\begin{aligned} |\{e \in \mathcal{E} \mid e(\underline{c}) = t,\ e(\underline{c}') = t'\}| &\leq qn - (q-1)d_H(\underline{c}, \underline{c}')) \\ &\leq qn - (q-1)d, \end{aligned}$$

since $\underline{c}$ and $\underline{c}'$ are different and so must different in at least d coordinates.

Suppose that the encoding rules $(a, b, i)s$ are chosen equiprobably. It follows from (2) and (13) that

$$P_S \leq (qn - (q-1)d)/qn = 1 - (q-1)d/qn. \tag{14}$$

Example 4.3 *Let $GF(q)$ be finite field of size q and let C be a (generalized) Reed-Solomon code over $GF(q)$. Then C is a linear code, say with parameters $[n, k, d]$, so $|C| = q^k$. The dimension k can take on any value in between 0 and n. The minimum distance is given by $d = n - k + 1$.*

Construction 4.1 now yields a systematic A-code with parameters (q^k, q^{k+1}, nq^2), with $P_I = 1/q$ and $P_S \leq 1 - (q-1)(n-k+1)/qn$.

In practice one likes to use A-codes with many source states, not too many encoding rules and a probability of success for deception that is as small as possible. In view of Construction 4.1 we need EC-codes with as many codewords and as large a minimum Hamming distance as possible. Concatenation codes may be convenient to use to this end.

5 Comparisons

Looking at Sections 2 and 4, it follows that Construction 4.1 has the following advantages and disadvantages over the q-twisted construction [3, 5].

- The lower bound on $1 - P_S$ in Construction 4.1 is a factor $(q-1)/q$ lower than in the q-twisted construction. For large field sizes this factor is negligible.
- Construction 4.1 can be applied to any type of EC-code. Property (8) is no longer required.

- Let $A_q(n,d)$ denote the maximum cardinality of a q-ary EC-code of length n and minimum distance d. Let $A_q^{(1)}(n,d)$ be defined similarly for EC-codes satisfying (8). It may happen that for some parameters $A_q^{(1)}(n,d)$ will be strictly smaller than $A_q(n,d)$.

- Given a code C satisfying (8). Then the A-code in Construction 4.1 contains q times as many source states as the A-code in the q-twisted construction. On the other hand, it uses q times as many encoding rules. In view of Theorem 1.1 this can be viewed as a good trade-off.

We note that the number of source states in both the q-twisted construction and Construction 4.1 increases exponentially in the length of the code.

6 Conclusions

We introduced and analyzed a new construction method for systematic authentication codes. It makes use of affine transformations and error-correcting codes. The main difference with [3] is that we do not need to impose any restrictions on the EC-codes that are used. As a result the A-code contains as many source states as the EC-code contains codewords.

References

[1] E.F.Brickell, *A few results in message authentication*, Congressus Numerantium, Vol.43(1984), pp.141-154.

[2] E.N. Gilbert, F.J. MacWilliams, and N.J.A. Sloane, *Codes which detect deception*, The Bell System Technical Journal, Vol. 53, No.3, pp. 405-424, March 1974.

[3] T. Johannsson, G.A. Kabatianskii, and B. Smeets, *On the relation of A-codes and codes correcting independent errors*, Advances in Cryptology, EUROCRYPT'93, pp.1-11, 1993.

[4] T.Johansson, *Contributions to unconditionally secure authentication*, KF Sigma, Lund, 1994.

[5] G.A. Kabatianskii, B. Smeets, and T. Johannsson, *On the cardinality of systematic A-codes via error-correcting codes*, IEEE Transactions on Information Theory, Vol. IT-42, No.2, pp. 566-578, March 1996.

[6] F.J. MacWilliams and N.J. Sloane, *The theory of error-correcting codes*, North-Holland Publishing Company, New York 1978.

[7] R.J.McEliece, *The theory of information and coding*, Encyclopedia of Math. and its Applications, Vol. 3, Addison-Wesley Publ. Comp., Reading, Mass., 1977.

[8] G.J. Simmons, *Authentication theory and coding theory*, Advances in Cryptology, Proc. CRYPTO-84, pp. 411-431, Springer-verlag, 1985.

[9] G.J. Simmons, *A survey of information authentication*,in: Contemporary Cryptology, The Science of information integrity (ed. G.J.Simmons), IEEE Press, New York, 1992.

[10] R.S. Safavi-Naini and J.R. Seberry, *Error-Correcting codes for authentication and subliminal channels*, IEEE Transactions Information Theory, vol. IT-37, No.1, pp.13–17, January 1991.

[11] D.R. Stinson, *The combinatorics of authentication and secrecy codes*, Journal of Cryptology, Vol.2, No.1, pp. 23–49, 1990.

[12] D.R. Stinson, *Universal hashing and A-codes*, Designs, Codes and Cryptography, 1994, No.4, pp. 369–380.

[13] D.R. Stinson, *Cryptography–Theory and Practice*, CRC Press, Inc., March 1995.

[14] B.Smeets, P.Vanroose, and Z.-X.Wan, *On the construction codes with secrecy and codes withstanding spoofing attacks of order* $L \geq 2$, in: Cryptography and Coding (ed. C. Boyd), Lecture Notes in Computer Science 1025, Springer Verlag, New York etc., pp. 169–183, 1995.

[15] H.C.A. van Tilborg, *Authentication codes: an area where coding and cryptography meet*, in: Cryptography and Coding (ed. C. Boyd), Lecture Notes in Computer Science 1025, Springer Verlag, New York etc., pp. 169–183, 1995.

[16] M.N. Wegman and J.L. Carter, *New hash functions and their use in authentication and set equality*, J. Comput. System Sci. 22, pp. 265–279, 1981.

Arithmetic Coding And Data Integrity

Xian Liu, Patrick G. Farrell*, and Colin A. Boyd"

Communications Research Group, School of Engineering, University of Manchester, Manchester M13 9PL, UK
liu@snow-goose.ee.man.ac.uk
*Communications Research Centre, Lancaster University, Lancaster LA1 4YR, UK
p.g.farrell@lancaster.ac.uk
"Information Security Research Centre, Queensland University of Technology, Brisbane Q4001, Australia
boyd@fit.qut.edu.au

Abstract: We have proposed a novel scheme based on arithmetic coding, an optimal data compression algorithm in the sense of shortest length coding. The scheme can provide data compression, encryption, and data integrity all together at the same time in a one pass operation. The key size used is 248 bits. The scheme can resist existing attacks on arithmetic coding encryption algorithms. A general approach to attack this scheme either on data secrecy or on data integrity is difficult. The statistical properties of this scheme are very good and the scheme is easily manageable in software. The compression ratio is only 2% worse than the original arithmetic coding algorithm.

1 Introduction

Data integrity or authenticity has two meanings. Firstly, it means that the data is not changed or altered during transmissions or storage. Secondly, it means that the data is originated from the authentic sender. Data integrity can be achieved by adopting either digital signature or message authentication code (MAC) algorithms. Digital signature, which was introduced by Diffie and Hellman in 1976, is superior to MACs in that it can provide unforgable and undeniable identification of the sender as well. Digital signature algorithms are in practice always based on a public key cryptographic system which has the advantage over a symmetric counterpart in key management. Even so, almost all of practical digital signature schemes, such as RSA and DSA, depend on a hard number theoretic problem. Due to the implementation technical problems, for some applications, digital signature algorithms require comparatively too much storage and computation. Therefore, for a lot of applications, conventional MAC algorithms are still used to achieve data integrity. MACs, such as MD5 and CBC-MAC, are usually used in a symmetric system in which two communication parties must trust each other and that means they cannot provide non-repudiation of origin. The shared symmetric keys make the key management harder and more costly. However, MACs are around 100 to 1000 times faster in the speed than digital signatures so they are preferable for some specific applications. One feature for both digital signature and MAC algorithms is that they are unable to provide data secrecy on their own. If data secrecy is required a separate cipher must be applied or for public key systems a two pass operation must be adopted.

Arithmetic coding [1] is an optimal data compression algorithm. It can achieve the theoretical compression ratio bound provided that the prediction of the symbol probabilities in the source is accurate enough. The first practical implementation for arithmetic coding with either a fixed

model or a first-order adaptive model was published in 1987 by Witten et al [9] (which we call the WNC implementation). Since then there have been several different implementations for arithmetic coding with adaptive models in which the symbol probabilities are evolving on the fly. Of course, the purpose of data compression is to reduce the redundancy in the message. On the other hand, redundancy contained in the output of a cryptosystem is usually one of the main resources to be used by the cryptanalyst. Based on these facts Witten et al [8] suggested that a higher-order adaptive arithmetic coding algorithm may provide high level security. They also indicated that to use an adaptive modelling compression algorithm as an encryption algorithm it was enough to transmit the initial state in the model as a key over a secure channel. In 1993 [3] Bergen and Hogan suggested a chosen plaintext attack on first-order adaptive arithmetic coding. Although this attack is not feasible to break an arithmetic coding encryption algorithm with an advanced adaptive model or the prediction by partial mach model, it succeeded in attacking the first-order adaptive arithmetic coding algorithm. In [10] we proposed a novel encryption algorithm based on first-order adaptive arithmetic coding whose strength is good enough to resist Bergen-Hogan attack. In this paper we will reduce the key size of our scheme to 248 bits and also we will provide a modified scheme to achieve data integrity as well so that data compression, encryption, and data integrity can be provided all together at the same time by the same algorithm in a one pass operation.

2 Arithmetic Coding

Arithmetic coding is based on the fact that the cumulative probability of a sequence of statistically independent source symbols equals the product of the source symbol probabilities. In arithmetic coding each symbol in the message is assigned a distinct subinterval of the unit interval of length equal to its probability. This is the encoding interval for that symbol. As encoding proceeds, a nest of subintervals is defined. Each successive subinterval is defined by reducing the previous subinterval in proportion to the current symbol' s probability. When the message becomes longer, the subinterval needed to represent it becomes smaller, and the number of bits needed to indicate that subinterval grows. The more likely symbols reduce the subinterval by less than the unlikely symbols and thus add fewer bits to the message. This results in data compression. When all symbols have been encoded, the final interval has length equal to the product of all the symbol probabilities and can be transmitted by sending any number belonging to the final interval. That means if the probability of the occurrence of a message is p, arithmetic coding can encode that message in $-\log_2 p$ bits, which is optimal in the sense of the shortest length encoding. The WNC implementation for arithmetic coding [9] is the first practical algorithm and is widely accepted. The algorithm is provided with either a static model or a first-order adaptive model. The algorithm realises integer arithmetic and incremental transmission. The arithmetic precision is 16-bit. In their first-order adaptive model, all the frequencies are initialised to 1. If the current model exceeds the maximum cumulative frequency, the model reduces all frequencies by half and recalculates cumulative frequencies. If necessary the model reorders the symbols to always put the current one in its correct rank in the frequency ordering. Adaptation is performed by incrementing the proper frequency count and adjusting cumulative frequencies accordingly. Due to the limit of the first-order adaptation, the compression ratio is 50% to 70% according to the size and type of the file. However it can be greatly improved by using a higher-order adaptive model.

3 Our Basic Scheme

3.1 Witten-Cleary Proposal

In 1988 [8] Witten and Cleary suggested two ways to insert the key into arithmetic coding:

Method 1: The initial model is used as the key in which an array of single-character frequencies in the range of 1-10 would do.

Method 2: A constant initial model is used and before transmission begins both the encoder and decoder assimilate a short secret message into the model.

Aiming at Method 1 in Witten-Cleary proposal with WNC adaptive implementation, Bergen and Hogan suggested a chosen plaintext attack on first-order adaptive arithmetic coding in 1993 [3]. Instead of trying to recover the initial model Bergen-Hogan attack tries to take control of the model and reduce it to a manageable form. If the encoder does not initialise its model, the attacker can decrypt any message transmitted after the attack is done. To be successful, in the Bergen-Hogan attack an associate as well as an attacker are necessary. The associate needs to send 2^{18} symbols and the attacker needs to try decoding the test string 2^{14} times. Up until now the Bergen-Hogan attack is the only feasible attack on the adaptive arithmetic coding encryption algorithm. It is an open question whether or not a modified Bergen-Hogan attack will succeed in breaking the arithmetic coding algorithm with an advanced adaptive model, such as higher-order adaptive models as well as the Prediction by Partial Match Model (PPM) [1].

3.2 Our Modified Proposal

In [10] we gave our initial proposal. Here is the revised version to reduce the total key size to 248 bits without impacting the strength significantly.

1. Select an initial frequency count for every symbol randomly, which act as the initial state in the model. The only restriction is that these numbers are all larger than 0.

2. Select the initial interval randomly within the full range but the length of the initial interval should not be less than $(2^{16}-1)/4$.

3. Select a secret 16-bit substitution with the key size 16 bits, which is used to substitute the first 16-bit output of the encoding.

4. Choose two secret parameter pairs $(\varepsilon_l{}^0, \varepsilon_h{}^0)$ and $(\varepsilon_l{}^1, \varepsilon_h{}^1)$ which are used to shrink the current interval controlled by a random 64-bit string cyclically, where the four parameters are different.

3.3 The Key Size

Firstly, 96 bits can be used to indicate the initial state. There are 96 symbols with exact meaning in extended ASCII set in text compression, so 96 bits are enough to indicate the initial state. If the bit is 0 set the count of the symbol to 2, otherwise to 3. Set the counts of the other

160 symbols to be 1. Secondly, 32 bits can be used to indicate the initial interval. The initial interval can be indicated by determining *low* and *high*, or *low* and *range*, so 16 bits are used to indicate one of them and 32 bits for both. Thirdly, 40 bits can indicate the shrinking parameters. The unknown part in every parameter ranges from 0 to 999, so 10 bits are necessary to indicate each of them. And then, we use 64 bits for the random control string. In this case, the polynomials (in Sec. 4) for *low* and *range* are of degree 256. And then, 16 bits are used for the 16-bit substitution key size. A secret 16-bit substitution with the key size 16 bits is preferable. So the total key size for the scheme is 248 bits.

3.4 The Strength

3.4.1 Resisting Bergen-Hogan Attack

In Bergen-Hogan attack the attacker knows he matches the keying materials only when he successfully decodes the test string. To use Bergen-Hogan attack on our proposal, the associate' s strategy is the same as that to attack with the Witten-Cleary proposal, but the attacker' s work will be increased dramatically. In order to decode the test string the attacker has to find the first symbol' s frequency count in the standard form, the initial interval, the substitution, the pairs $(\varepsilon_l{}^0, \varepsilon_h{}^0)$ and $(\varepsilon_l{}^1, \varepsilon_h{}^1)$, and the 64-bit control string all together, instead of just trying the first symbol' s frequency count in the standard form 2^{14} times in breaking with the Witten-Cleary proposal. The attacker has to try to decode the test string $2^{14} \times 2^{30} \times 2^{16} \times 2^{19} \times 2^{19} = 2^{168}$ times. Partially finding the keying materials is also very difficult. The reasonably simplest way for the attacker would be to firstly find the first symbol' s frequency count in the standard form together with the initial interval. For this purpose the attacker only needs to decode the first symbol in the test string, but he has to try decoding $2^{14} \times 2^{30} \times 2^{16} \times 2^{19} = 2^{79}$ times.

It has been shown in [10] that a general approach to attack our scheme is difficult and our scheme can resist other related attacks to arithmetic coding encryption algorithms.

3.4.2 Other Related Results

It has been shown in [10] that compared with WNC first order adaptive implementation, the compression ratio of our scheme is only 2% worse and the running time is about double of that of WNC implementation. The results of compression ratio and running time are the same for our modified scheme. The encoded files with our modified scheme have very good randomness. Changing any number of bits in the file to be encoded results in the fact that from the position in the encoded file the first changed bit corresponds to, then in the subsequent output, if this encoded file is compared with the encoded file resulted from the totally unchanged original file to be encoded, the changed bits and unchanged bits take the probabilities 0.5, and distribute uniformly and randomly. Furthermore, the outputs of our modified scheme passed the frequency test, the binary derivative test, the change point test, the poker test, the runs test, the sequence complexity test, the linear complexity test, and Maurer's universal test (the statistical test software Crypt-X [11] is from the Information Security Centre at the Queensland University of Technology), and also there is no statistical difference between the output from our modified scheme and that from the DES. So good plaintext diffusion and ciphertext avalanche are

achieved. Also, in the modified scheme, the keying materials have very good effects on balance, diffusion, completeness, and avalanche.

4 A Scheme to Achieve Data Integrity

Compared with original arithmetic coding we have introduced 2% redundancy into our scheme. The main steps to introduce redundancy into our scheme are the two stages of shrinking in the current interval after encoding (decoding) a symbol. The fact that 2% redundancy is introduced into our scheme and the scheme is originally from arithmetic coding results in the fact that the scheme is highly sensitive to any changes in the compressed file. From a number of months experiments, we have found that, without any exception, if we change the bits in the encoded file randomly, no matter how many bits in the encoded file have been changed, in the decoded file from the position the first changed bit in the encoded file corresponding to, the subsequent decoded file is totally rubbish but if only the last bit in the encoded file is changed there is not any affect on the decoding. This fact suggests a method to achieve data integrity: before encoding the file with our scheme, we add a sequence number together with fifty zeros to the end of the file. The decoder only needs to check if the sequence number is correct (to detect replay) and if 50 zeros are presented at the end of the decoded file. This method has hardly any affect on the compression ratio.

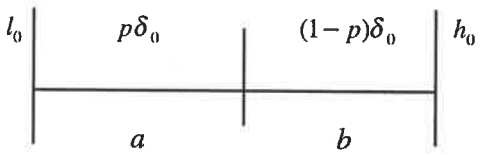


Figure 1. A binary fixed model with initial interval not unit.

Now we analyse how difficult it would be for the attacker to succeed in breaking the method without knowing the key. The binary fixed model in Figure 1 is the simplest one to attack which is the reason to analyse it, while in practice a higher order adaptive model is used with our scheme, which provides much better compression and also makes cryptanalysis much harder. First, we use the binary fixed model as in Figure 1 with our scheme. We assume that the substitution is ignored and the arithmetic precision is infinite and also the encoding is with our scheme in which the data integrity method is included. Let the size of the file to be encoded be n (>64) and the size of the sequence number together with the 50 zeros be m, so the size of the whole file to be encoded is $n+m$. When finishing encoding the output can be determined by:

$$\delta_{n+m} = \prod_{i=1}^{n}[a_i p+(1-a_i)(1-p)][\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_i][1-\varepsilon_l{}^0+(\varepsilon_l{}^0-\varepsilon_l{}^1)x_i]\delta_0 \times$$
$$\times \prod_{j=n+1}^{n+m}[a_j p+(1-a_j)(1-p)][\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_j][1-\varepsilon_l{}^0+(\varepsilon_l{}^0-\varepsilon_l{}^1)x_j]$$
$$l_{n+m} = l_0+\delta_0(1-a_1)p$$
$$+\sum_{i=2}^{n}(1-a_i)p\prod_{j=1}^{i-1}[a_j p+(1-a_j)(1-p)][\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_j][1-\varepsilon_l{}^0+(\varepsilon_l{}^0-\varepsilon_l{}^1)x_j]\delta_0+$$
$$+[a_1 p+(1-a_1)(1-p)]\delta_0[\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_1][\varepsilon_l{}^0+(\varepsilon_l{}^1-\varepsilon_l{}^0)x_1]+$$

$$+\sum_{i=2}^{n}[a_i p+(1-a_i)(1-p)][\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_i][\varepsilon_l{}^0+(\varepsilon_l{}^1-\varepsilon_l{}^0)x_i]\times$$
$$\times\prod_{j=1}^{i-1}[a_j p+(1-a_j)(1-p)][\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_j][1-\varepsilon_l{}^0+(\varepsilon_l{}^0-\varepsilon_l{}^1)x_j]\delta_0+$$
$$+\sum_{i=n+1}^{n+m}(1-a_i)p\prod_{j=1}^{i-1}[a_j p+(1-a_j)(1-p)][\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_j][1-\varepsilon_l{}^0+(\varepsilon_l{}^0-\varepsilon_l{}^1)x_j]\delta_0+$$
$$+\sum_{i=n+1}^{n+m}[a_i p+(1-a_i)(1-p)][\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_i][\varepsilon_l{}^0+(\varepsilon_l{}^1-\varepsilon_l{}^0)x_i]\times$$
$$\times\prod_{j=1}^{i-1}[a_j p+(1-a_j)(1-p)][\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_j][1-\varepsilon_l{}^0+(\varepsilon_l{}^0-\varepsilon_l{}^1)x_j]\delta_0 \qquad (4.1)$$

Here, δ_i represents the length of the current interval; l_i represents the low point in the current interval; a_i represents the current input symbol; p is the probability for the symbol a in the fixed model; and x_j is the current bit in the control string.

As in theoretical arithmetic coding the low bound l is usually used as the encoded file, it is reasonable to use l_{n+m} to represent the output of the scheme.

4.1 The Complexity to Change a File

Suppose the attacker knows the whole file to be encoded, $a_1a_2\ldots a_n\ldots a_{n+m}$, and the corresponding output l_{n+m}, he will succeed if he successfully finds a l'_{n+m} corresponding to his changed file $a'_1a'_2\ldots a'_n a_{n+1}\ldots a_{n+m}$, which can be decoded as a file ended with a correct sequence number together with 50 zeros. We have:

$$l'_{n+m}=l_{n+m}+\delta_0(a_1-a'_1)p+$$
$$+\sum_{i=2}^{n}(1-a'_i)p\prod_{j=1}^{i-1}[a'_j p+(1-a'_j)(1-p)][\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_j][1-\varepsilon_l{}^0+(\varepsilon_l{}^0-\varepsilon_l{}^1)x_j]\delta_0+$$
$$+[a'_1 p+(1-a'_1)(1-p)]\delta_0[\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_1][\varepsilon_l{}^0+(\varepsilon_l{}^1-\varepsilon_l{}^0)x_1]+$$
$$+\sum_{i=2}^{n}[a'_i p+(1-a'_i)(1-p)][\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_i][\varepsilon_l{}^0+(\varepsilon_l{}^1-\varepsilon_l{}^0)x_i]\times$$
$$\times\prod_{j=1}^{i-1}[a'_j p+(1-a'_j)(1-p)][\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_j][1-\varepsilon_l{}^0+(\varepsilon_l{}^0-\varepsilon_l{}^1)x_j]\delta_0-$$
$$-\sum_{i=2}^{n}(1-a_i)p\prod_{j=1}^{i-1}[a_j p+(1-a_j)(1-p)][\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_j][1-\varepsilon_l{}^0+(\varepsilon_l{}^0-\varepsilon_l{}^1)x_j]\delta_0-$$
$$-[a_1 p+(1-a_1)(1-p)]\delta_0[\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_1][\varepsilon_l{}^0+(\varepsilon_l{}^1-\varepsilon_l{}^0)x_1]-$$
$$-\sum_{i=2}^{n}[a_i p+(1-a_i)(1-p)][\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_i][\varepsilon_l{}^0+(\varepsilon_l{}^1-\varepsilon_l{}^0)x_i]\times$$
$$\times\prod_{j=1}^{i-1}[a_j p+(1-a_j)(1-p)][\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_j][1-\varepsilon_l{}^0+(\varepsilon_l{}^0-\varepsilon_l{}^1)x_j]\delta_0 \qquad (4.2)$$

The equation above has similar structure with (4.1) but is slightly more complicated. In order to find the l'_{n+m} the attacker has to find the key in advance or solve the polynomial equation with 70 variables and with the degree (≥) 256. Here we assume that only the key is unknown. If the size of the changed file is different from that of the original file the resulting equation is more complicated.

4.2 The Complexity to Create a File

In this case we have:

$$
\begin{aligned}
&l'_{n+m} = l_0 + \delta_0(1-a'_1)p + \\
&+\sum_{i=2}^{n}(1-a'_i)p\prod_{j=1}^{i-1}[a'_j p+(1-a'_j)(1-p)][\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_j][1-\varepsilon_l{}^0+(\varepsilon_l{}^0-\varepsilon_l{}^1)x_j]\delta_0 + \\
&+[a'_1 p+(1-a'_1)(1-p)]\delta_0[\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_1][\varepsilon_l{}^0+(\varepsilon_l{}^1-\varepsilon_l{}^0)x_1] + \\
&+\sum_{i=2}^{n}[a'_i p+(1-a'_i)(1-p)][\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_i][\varepsilon_l{}^0+(\varepsilon_l{}^1-\varepsilon_l{}^0)x_i]\times \\
&\times\prod_{j=1}^{i-1}[a'_j p+(1-a'_j)(1-p)][\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_j][1-\varepsilon_l{}^0+(\varepsilon_l{}^0-\varepsilon_l{}^1)x_j]\delta_0 + \\
&+\sum_{i=n+1}^{n+m}(1-a'_i)p\prod_{j=1}^{i-1}[a'_j p+(1-a'_j)(1-p)][\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_j][1-\varepsilon_l{}^0+(\varepsilon_l{}^0-\varepsilon_l{}^1)x_j]\delta_0 + \\
&+\sum_{i=n+1}^{n+m}[a'_i p+(1-a'_i)(1-p)][\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_i][\varepsilon_l{}^0+(\varepsilon_l{}^1-\varepsilon_l{}^0)x_i]\times \\
&\times\prod_{j=1}^{i-1}[a'_j p+(1-a'_j)(1-p)][\varepsilon_h{}^0+(\varepsilon_h{}^1-\varepsilon_h{}^0)x_j][1-\varepsilon_l{}^0+(\varepsilon_l{}^0-\varepsilon_l{}^1)x_j]\delta_0
\end{aligned}
\tag{4.3}
$$

The attacker has to solve the equation above, which has 71 variables and the degree at least 256 together with finding the proper sequence number.

4.3 The Complexity to Find a Collision

Finding a collision is one of the most important ways to break a MAC. However, our scheme is uniquely decodable. So with our scheme two files correspond to the same compressed code only when the two files are the same.

4.4 Extension

The analysis above is based on a much simplified fixed binary model and with ideal theoretical arithmetic coding. Our scheme works with the Witten-Cleary first order adaptive model with the alphabet size 256. There has not been any arithmetic method to trace the evolution of the adaptive model. Also our scheme works with Witten-Cleary implementation for arithmetic coding. Quite a few practical strategies in the implementation make the coding procedure much more difficult to trace than the theoretical arithmetic coding. In fact, there are a number of main differences between the scheme with theoretical arithmetic coding with a fixed binary model and the scheme with WNC first order adaptive arithmetic coding. Firstly, for the WNC first adaptive arithmetic coding, there is no way to find the exact current state in the adaptive model. One may argue that as the uncertainty of the current state in the adaptive model, if a

chosen plaintext attack is used and the secret shrinking parameters, the initial interval, and the secret control as well string are known, totally depends on the secret initial state in the model, after encoding a huge known file the effect from the initial model is trivial; i.e., it can be converted from any known initial model. However, this is not true with WNC first order adaptive model. Approximation does not make any sense unless the two current states are the same in arithmetic coding. Secondly, WNC adaptive arithmetic coding uses 16-bit finite precision. That means it has to expand the current interval after encoding (decoding) one symbol. Such expansions are unpredictable and untraceable with our scheme. Therefore, such a regular relation between the input and the output in the encoder in (4.1) definitely does not hold in our scheme with WNC adaptive arithmetic coding. One thing clear is that if a mathematical relation exists in WNC adaptive arithmetic coding it must be much more complicated than (4.1). Now we analyse what happens in the algorithm of our scheme when the encoded file is changed. To avoid underflow, WNC adaptive arithmetic coding expands the current interval after encoding (decoding) a symbol so that the length of current interval is always at least as large as one fourth of the full interval. After every expansion, our scheme further shrinks the current interval two times secretly but the amplitudes are not large. From a conservative estimation, we assume that the current interval is always about 3/4 of the full interval. If a bit is changed, then there is a probability 1/4 that the next symbol can not be decoded and the decoding stops, because the current value from the input of the decoding is located outside the current interval. There is a probability 3/4 that the decoder decodes the next symbol but the resulting symbol is absolutely wrong. After decoding that wrong symbol the adaptive model updates the current state in the decoder which is different from that in the encoder. The procedure is on going. Therefore, from the changed bit the probability that the decoder is still in operation after it has decoded n wrong symbols is about $(3/4)^n$. What exactly the wrong symbols are depends on the exact current interval and the input as well as the position of the changed bit. So, there is a very high probability of the decoder stopping intermediately. Because of the adaptivity of the model, there is definitely no chance for the decoder to recover the correct symbols later and that means there is no chance for the decoded file to be ended with a correct sequence number and 50 zeros. The results are the same when changing a number of bits in the encoded file. As the encoded file is pseudorandom, there is no better way to change or generate an encoded file than randomly. What is the possibility of a succesful forgery from a random guess? If one generates an encoded file which is supposed to be a forgery of a plaintext file with n symbols randomly, the probability of the decoder still being in operation after decoding n symbols and still being ready to try to decode the supposed sequence number and 50 zeros is about $(3/4)^n$. If the file he wants to be authenticated is a random file, the probability of his randomly generated encoded file being succesfully decoded to the same file to be encoded of his choice is about $(3/4)^n(1/256)^n$. The probability of this file being ended with a correct sequence number and 50 zeros is about $(3/4)^n(1/256)^n(1/256)^{50+m}$. Here the m is the size of the sequence number in bytes. Another option for the attacker is to find the key first and then the sequence number. As it was demonstrated before, by using Bergen-Hogan attack, which is, we think, the most powerful attack on the specific scheme, the complexity of finding the key is 2^{168}.

4.5 Compared With MACs

MACs have been widely used in symmetric systems. If the security of our scheme is satisfactory, there are some specific features of our scheme which are superior to normal MACs. Firstly, MACs are used in this way that the file to be authenticated and the corresponding MAC are sent separately. In our scheme we only need to send one encoded file. Secondly, MACs can only detect data integrity. If data security is also required, a separate cipher must be applied. That is a two pass operation with two different algorithms. Our scheme can provide data integrity, encryption, and data compression all together in a one pass operation. The compression ratio depends on which adaptive model we use. Arithmetic coding with a new version of the PPM model can represent one character in 2 bits. We expect that only about 2% redundancy will be added with such a PPM modelling scheme.

5 Conclusions

In this paper we present a scheme that can provide data encryption, data integrity and data compression all together in a one pass operation. The scheme is based on WNC implementation for arithmetic coding in which a first order adaptive model is used. The total key size is 248 bits. The statistical properties of our scheme are very good. Attacking this scheme is difficult. The compression ratio is about 2% worse than WNC implementation for arithmetic coding.

References

1. Bell T., Cleary J. and Witten I.: *Text Compression*, Prentice Hall, 1990.
2. Bergen H. and Hogan J.: "Data security in a fixed-model arithmetic coding compression algorithm", Computer and Security, Vol.11, 1992, pp.445-461.
3. Bergen H. and Hogan J.: "A chosen plaintext attack on an adaptive arithmetic coding compression algorithm", Computers and Security, Vol.12, 1993, pp.157-167.
4. Boyd C.: "Enhancing secrecy by data compression : theoretical and practical aspects", Proceedings of Eurocrypt'91, LNCS-547, Springar-Verlag, 1991, pp.266-280.
5. Boyd C., Cleary J., Irvine S., Rinsma-Melchert I. and Witten I.: "Integrating error detection into arithmetic coding", IEEE Trans. COM, Vol.45, No.1, 1997, pp.1-3.
6. Cleary J., Irvine S. and Rinsma-Melchert I.: "On the insecurity of arithmetic coding", Computers and Security, Vol.14, 1995, pp.167-180.
7. Irvine S. and Cleary J.: "The subset sum problem and arithmetic coding", private communication, 1995.
8. Witten I. and Cleary J.: "On the privacy afforded by adaptive text compression", Computers and Security, Vol.7, 1988, pp.397-408.
9. Witten I., Neal R. and Cleary J.: "Arithmetic coding for data compression", Communications of the ACM, Vol.30, No.6, 1987, pp.520-540.
10. Liu X., Farrell P., and Boyd C.: "Resisting the Bergen-Hogan attack on adaptive arithmetic coding", LNCS-1355, Cryptography and Coding, Springer, December, 1997, pp.199-208.
11. Crypt-X, Statistical Package Manual, Measuring the Strength of Stream and Block Ciphers. Information Security Research Centre, Queensland University of Technology, 1990.

NEGACYCLIC AND CYCLIC CODES OVER $\mathbb{Z}_4$

J.WOLFMANN

ABSTRACT. The negashift ν of $\mathbb{Z}_4^n$ is defined as the permutation of $\mathbb{Z}_4^n$ such that $\nu(a_0, a_1, \dots, a_i, \dots, a_{n-1}) = (-a_{n-1}, a_0, \dots, a_i, \dots, a_{n-2})$ and a negacyclic code of length n over $\mathbb{Z}_4$ is defined as a subset C of $\mathbb{Z}_4^n$ such that $\nu(C) = C$. We prove that the Gray map image of a linear negacyclic code over $\mathbb{Z}_4$ of length n is a binary distance invariant (not necessary linear) cyclic code. We also prove that, if n is odd, then every binary code which is the Gray map image of a linear cyclic code over $\mathbb{Z}_4$ of length n is equivalent to a (not necessary linear) cyclic code and this equivalence is explicitely described. This last result explains and generalizes the existence, already known, of version as doubly extended cyclic codes of Kerdock, Preparata, and other codes. Furthermore, we introduce a family of binary linear cyclic codes which are Gray map images of $\mathbb{Z}_4$-linear negacyclic codes.

1. NEGACYCLIC CODES

1.1. Definitions.

Définition 1.
1) The negashift ν of $\mathbb{Z}_4^n$ is the permutation of $\mathbb{Z}_4^n$ defined by :

$$\nu(a_0, a_1, \dots, a_i, \dots, a_{n-1}) = (-a_{n-1}, a_0, \dots, a_i, \dots, a_{n-2}).$$

2) Let ν be the negashift of $\mathbb{Z}_4^n$.
A negacyclic code of length n over $\mathbb{Z}_4$ is a subset C of $\mathbb{Z}_4^n$ such that $\nu(C) = C$.

Using the classical polynomial representation we easily prove the following proposition

Proposition 2. *A subset C of $\mathbb{Z}_4^n$ is a linear negacyclic code of length n over $\mathbb{Z}_4$ if and only if its polynomial representation is an ideal of the factor ring $\mathbb{Z}_4[x]/(x^n + 1)$.*

1.2. Negacyclic codes of odd length.

Théorème 3. *a) The ring $\mathbb{Z}_4[x]/(x^n + 1)$ with n odd is a principal ideal domain.*

b) If C is a negacyclic code of odd length n, then its polynomial representation I is a principal ideal generated by a constant polynomial or a polynomial of the kind :

$$g(x) = a(x)[b(x) + 2]$$

where $x^n + 1 = a(x)b(x)c(x)$ in $\mathbb{Z}_4[x]$ and $a(x), b(x), c(x)$ are pairwise coprime polynomials.

c) The cardinality of C is $4^{\deg c(x)} 2^{\deg b(x)}$.

Key words and phrases. Negacyclic and cyclic codes over $\mathbb{Z}_4$, Gray map.

2. Gray map and negacyclic codes

2.1. Gray map.

From now on, in order to eliminate ambiguity we denote additions in $\mathbb{F}_2$, $\mathbb{F}_2^n$, $\mathbb{F}_2^{2n}$, $\mathbb{F}_2[x]$ and $\mathbb{F}_2^n \times \mathbb{F}_2^n$ by $\oplus$, while additions in $\mathbb{Z}_4$, $\mathbb{Z}_4^n$, $\mathbb{Z}_4[x]$ are denoted by $+$. We consider $\mathbb{F}_2^n$ as the subset $\{0,1\}^n$ of $\mathbb{Z}_4^n$ where 0 or 1 are in $\mathbb{Z}_4$. In this way, if X and Y are in $\mathbb{F}_2^n$ we have to distinguish $X+Y$ and $X \oplus Y$. The last one is calculated with the binary addition and is viewed as a member of $\mathbb{Z}_4^n$.

Now define two maps r and q of $\mathbb{Z}_4$ into $\mathbb{F}_2$ such that, if $\lambda \in \mathbb{Z}_4$, then the 2-adic expansion of λ is : $\lambda = r(\lambda) + 2q(\lambda)$. We extend r and q to $\mathbb{Z}_4^n$ in the natural following way. If $Z = (z_1, z_2, \dots, z_n) \in \mathbb{Z}_4^n$ then :

$$r(Z) = (r(z_1), r(z_2), \dots, r(z_n)), \quad q(Z) = (q(z_1), q(z_2), \dots, q(z_n))$$

We now identify $\mathbb{F}_2^{2n}$ as $\mathbb{F}_2^n \times \mathbb{F}_2^n$ and recall the definition of the Gray map.

Définition 4. *The Gray map Φ of $\mathbb{Z}_4^n$ into $\mathbb{F}_2^{2n}$ is defined by :*

$$\Phi(Z) = (q(Z), q(Z) \oplus r(Z))$$

2.2. Gray map images of negacyclic codes.

Proposition 5. *If ν is the negashift of $\mathbb{Z}_4^n$, if σ is the shift of $\mathbb{F}_2^{2n}$ and if Φ is the Gray map of $\mathbb{Z}_4^n$ into $\mathbb{F}_2^{2n}$ then*

$$\Phi\nu = \sigma\Phi$$

Théorème 6. *The Gray map image of a linear negacyclic code over $\mathbb{Z}_4$ is a binary distance invariant (not necessary linear) cyclic code.*

Example : Let C be the negacyclic code over $\mathbb{Z}_4$ of length 7, which is the principal ideal generated by $(x+1)(x^3+2x^2+x+1)$. The Gray image of C is a binary non-linear distance invariant cyclic code of length 14, cardinality 64 and minimum weight 6. For same length and cardinality the best minimum weight of a binary linear code is 5 and, from inspection by computer, the best minimum weight of a binary linear cyclic code is 4.

2.3. Gray map images of negacyclic codes of odd length.

Définition 7. *The Nechaev permutation of $\{0, 1, \cdots, 2n-1\}$ is the following permutation π :*

$$(1, n+1)(3, n+3)\dots(2i+1, n+2i+1)\dots(n-2, 2n-2)$$

As usual,if τ is a permutation of $\{0, 1, \cdots, r-1\}$ and A is any set we define the permutation $\overline{\tau}$ of A^r induced by τ by :

$\overline{\tau}(a_0, a_1, \dots, a_i, \dots, a_{r-1}) = (a_{\tau(0)}, a_{\tau(1)}, \cdots, a_{\tau(i)}, \cdots, a_{\tau(r-1)})$.

Proposition 8. *Assume n odd.*
Let $\tilde{\mu}$ be the permutation of $\mathbb{Z}_4^n$ such that :

$$\tilde{\mu}(a_0, a_1, \dots, a_i, \dots, a_{n-1}) = (a_0, -a_1, a_2, \dots, (-1)^i a_i, \dots, (-1)^{n-1} a_{n-1})$$

If $\overline{\pi}$ is the permutation of $\mathbb{F}_2^{2n}$ induced by the Nechaev permutation and if Φ is the Gray map $\mathbb{Z}_4^n$ into $\mathbb{F}_2^{2n}$ then

$$\Phi\tilde{\mu} = \overline{\pi}\Phi$$

Corollaire 9. *Let $\overline{\pi}$ be as defined in the last proposition. If n is odd and if Γ is the Gray map image of linear cyclic code over $\mathbb{Z}_4$, then $\overline{\pi}(\Gamma)$ is a cyclic code.*

Recall that two codes Γ and Δ of length r over $\mathbb{F}_2$ are said to be equivalent if there exists a permutation τ of $\{0, 1, \cdots, r-1\}$ such that $\Delta = \overline{\tau}(\Gamma)$ where $\overline{\tau}$ is the permutation of $\mathbb{F}_2^r$ induced by τ. Obviously, a consequence of the previous result now is :

Théorème 10. *Every binary code which is the Gray map image of a linear cyclic code over $\mathbb{Z}_4$ of odd length is equivalent to a (not necessary linear) cyclic code.*

2.4. **Examples : Kerdock, Preparata and others.** Nechaev in [5], was the first to prove that the binary Kerdock code $\tilde{K}$ of length 2^{m+1} is equivalent to a code $\tilde{K}_1$ which a doubly extended code of a non-linear cyclic code $\tilde{K}_1^-$. He used the Galois ring $GR(4, m)$ and a connection between the trace function of $GR(4, m)$ and the trace function of the finite field $\mathbb{F}_{2^m}$. Later (see [3]), it was proved that $\tilde{K}$ is the Gray map image of a $\mathbb{Z}_4$-linear code K which is the extended code of a cyclic code K^- over $\mathbb{Z}_4$ of length $n = 2^m - 1$. In fact, it can be easily seen that $\tilde{K}^-$ is the Gray map image of K^- which is equivalent to $\tilde{K}_1^-$ by means of the Nechaev permutation defined. This is a special case of corollary 2.6. On the other hand, Delsarte-Goethals codes, Goethals-Delsarte codes and the version of the Preparata codes, all introduced in [3], are Gray map images of extended cyclic codes of length $n = 2^m - 1$ over $\mathbb{Z}_4$. Applying corollary 2.6., they all are equivalent to doubly extended cyclic codes. Same results hold for codes of [1],[2].

Another application of our result is given by the $\mathbb{Z}_4$ trace codes introduced in [4] and [6] which are $\mathbb{Z}_4$-linear cyclic code of length $n = 2^m - 1$. According to corollary 2.6., their Gray map images are equivalent, by means of the Nechaev permutation, to binary doubly extended cyclic codes.

2.5. **More examples.** We now introduce examples of Gray map images of linear negacyclic codes which, for same length and cardinality, have minimum weight close to the largest minimum weight for linear code and have better minimum weight than the largest minimum weight for linear cyclic code.

C : linear negacyclique code over $\mathbb{Z}_4$ generated by $g(x) = a(x)[b(x) + 2]$ where $x^n + 1 = a(x)b(x)c(x)$ in $\mathbb{Z}_4[x]$ and $a(x), b(x), c(x)$ are pairwise coprime polynomials. $\tilde{C} = \Phi(C)$. Length of $\tilde{C}$: $\tilde{n} = 2n$.
Cardinality of $\tilde{C} = 2^k$ with $k = 2 \deg c(x) + \deg b(x)$.
$w_{neg.}$ is the minimum weight of $\tilde{C}$, $w_{lin.}$ is the largest minimum weight for binary linear codes of length $\tilde{n}$ and dimension k and $w_{cycl.}$ is the largest minimum weight for binary linear cyclic codes of length $\tilde{n}$ and dimension k. $w_{lin.}$ comes from tables and $w_{cycl.}$ is obtained with the help of a computer.

1) $\tilde{n} = 14$
$a(x) = (x+1)(x^3 + 2x^2 + x + 1)$, $b(x) = 1$.
$k = 6$, $w_{neg.} = 6$, $w_{lin.} = 5$, $w_{cycl.} = 4$.

$a(x) = x^3 + 2x^2 + x + 1$, $b(x) = 1$.

$k = 8,\ w_{neg.} = 4,\ w_{lin.} = 4,\ w_{cycl.} = 3.$

2) $\tilde{n} = 34$
$a(x) = (x+1)(x^8 + 3x^7 + 3x^6 + +3x^4 + +3x + 1)\ , b(x) = 1.$
$k = 16,\ w_{neg.} = 8,\ w_{lin.} = 8 - 9,\ w_{cycl.} = 6.$

$a(x) = x^8 + 2x^6 + x^5 + x^4 + x^3 + 2x^2 + 1\ , b(x) = x + 1.$
$k = 17,\ w_{neg.} = 8,\ w_{lin.} = 8,\ w_{cycl.} = 5.$

$a(x) = x^8 + 2x^6 + x^5 + x^4 + x^3 + 2x^2 + 1\ , b(x) = 1.$

3. Binary linear cyclic codes which are Gray map images of $\mathbb{Z}_4$ linear negacyclic codes

In general,Gray map images of $\mathbb{Z}_4$-linear codes are not binary linear codes. We now introduce $\mathbb{Z}_4$-linear negacyclic codes whose Gray map images are binary linear cyclic codes and, as corollary, $\mathbb{Z}_4$-linear cyclic codes whose Gray map images are binary linear codes.

Théorème 11. *Let n be an odd positive integer and let $\tilde{a}(x)$, $\tilde{b}(x)$ be in $\mathbb{F}_2[x]$ such that :*

$$x^n - 1 = (x-1)\tilde{a}(x)\tilde{b}(x)$$

where $(x-1)$, $\tilde{a}(x)$, $\tilde{b}(x)$ are pairwise coprime.
Let $a_1(x)$, $b_1(x)$ be respectively Hensel lifts of $\tilde{a}(x)$ and $\tilde{b}(x)$ and define $a(x) = a_1(-x)$ and $b(x) = b_1(-x)$.

If $\tilde{C}$ is the binary linear cyclic code of length $2n$ generated by $\tilde{a}(x)^2\tilde{b}(x)$, then $\tilde{C}$ is the Gray map image of the $\mathbb{Z}_4$-linear negacyclic code of length n generated by $a(x)(b(x)+2)$.

Example If $n = 7$, $a(x) = x^3+2x^2+x+1$, $b(x) = x^3+x^2+2x+1$, the Gray map image of the $\mathbb{Z}_4$-linear negacyclic code of length 7 generated by $g(x) = a(x)(b(x)+2) = x^6+3x^5+x^4+x^3+x^2+x+3$ is the binary linear cyclic code of length 14 generated by $\tilde{g}(x) = \tilde{a}(x)^2\tilde{b}(x) = x^9+x^8+x^6+x^5+x^4+x^3+1$.

The next corollary of theorem 3.1. introduces $\mathbb{Z}_4$-linear cyclic codes whose Gray map images are binary linear codes.

Corollaire 12. *With notations of theorem 3.1. :*
The Gray map image of the $\mathbb{Z}_4$-linear cyclic code of length n generated by $a_1(x)(b_1(x)+2)$ is a binary linear code $\tilde{C}_1$.
If $\overline{\pi}$ is the permutation of $\mathbb{F}_2^{2n}$ induced by the Nechaev permutation, then $\overline{\pi}(\tilde{C}_1)$ is the Gray map image of the $\mathbb{Z}_4$-linear negacyclic code of length n generated by $a(x)(b(x)+2)$.

Example : From the last example, the Gray map image of the $\mathbb{Z}_4$-linear cyclic code of length 7 generated by $g_1(x) = g(-x) = a_1(x)(b_1(x)+2) = x^6 + x^5 + x^4 + 3x^3 + x^2 + 3x + 3$ is the binary linear code of length 14 and is equivalent to a cyclic code.

4. References.

[1] A.R.Calderbank,G. McGuire,"Construction of a $(64, 2^{37}, 12)$ code via Galois Rings"
Design, Codes and Cryptography, Vol.10,number 2 (1997), pp.157-165

[2] A.R.Calderbank,G. McGuire, P.V. Kumar, T.Helleseth,"Cyclic codes over $\mathbb{Z}_4$, locator polynomials and Newton's identities"
IEEE Trans. Inform. Theory, Vol.IT-42 (1996),pp.217-226

[3] A.R. Hammons,Jr.,P.V. Kumar,A.R. Calderbank,N.J.A. Sloane,P. Solé, "The $\mathbb{Z}_4$-Linearity of Kerdock,Preparata,Goethals, and Related Codes"
IEEE Trans. Inform. Theory, Vol.IT-40 (1994), pp.301-319

[4] T.Helleseth, P.V. Kumar, O. Moreno, A.G. Shanbag, "Improved estimates via exponential sums for the minimum distance of $\mathbb{Z}_4$-linear trace codes"
IEEE Trans. Inform. Theory, Vol.IT-42 (1996), pp.1212-1216

[5] A.A. Nechaev, "The Kerdock code in a cyclic form"
Discrete Math. Appl. Vol.1 (1991) pp.365-384 (English translation of "Diskret.Mat",1989)

[6] A.G. Shanbag, P.V. Kumar, T.Helleseth, "Improved binary codes and sequence families from $\mathbb{Z}_4$-linear codes"
IEEE Trans. Inform. Theory, Vol. IT-42 (1996), pp.1582-1587

GECT, Université Toulon-Var, 83957 LA GARDE Cedex, FRANCE
E-mail address: wolfmann@univ-tln.fr

Codes of constant Lee or Euclidean weight

Jay A. Wood
Department of Mathematics, Computer Science & Statistics
Purdue University Calumet
Hammond, Indiana 46323–2094 USA
wood@calumet.purdue.edu
http://www.calumet.purdue.edu/public/math/wood

ABSTRACT. Carlet [2] has determined the linear codes over $\mathbb{Z}/(4)$ of constant Lee weight. This extended abstract describes a different approach to this problem, along the lines of [4], which has the potential to apply to a wide class of examples. In particular, we show that linear codes of constant Lee or Euclidean weight seldom exist over $\mathbb{Z}/(p^2)$ when p is an odd prime.

Over finite fields, any linear code with constant Hamming weight is a replication of simplex (i.e., dual Hamming) codes. There are several proofs of this result, including [1], [3], and [4]. Recently, Carlet [2] has proved a similar result for linear codes of constant Lee weight over $\mathbb{Z}/(4)$, indeed, over any $\mathbb{Z}/(2^m)$.

In this extended abstract we generalize the approach of [4]. While more complicated than Carlet's proof, our approach has the potential to apply to a wide class of weight functions over any finite commutative chain ring.

For the purposes of this extended abstract, we will discuss codes over rings of the form $\mathbb{Z}/(p^2)$, p prime. In the case of $\mathbb{Z}/(4)$, we recover Carlet's result as Theorem 6. For p odd, we show in Theorem 11 that very few constant weight codes exist.

1991 *Mathematics Subject Classification.* Primary 94B05.
Key words and phrases. Constant weight codes, Lee weight, Euclidean weight, extension theorem.
Partially supported by Purdue University Calumet Scholarly Research Awards.

1. Linear codes as modules

Throughout this extended abstract, the ground ring will be $R = \mathbb{Z}/(p^2)$, p prime. It will be convenient to view $\mathbb{Z}/(p^2)$ as the set

$$\mathbb{Z}/(p^2) = \{t \in \mathbb{Z} : -p^2/2 < t \leq p^2/2\}. \tag{1}$$

Only for $p^2 = 4$ is equality possible in $t \leq p^2/2$. A *linear code* C of length n is a submodule of R^n.

The *Lee weight* $w(x)$ of any element $x = (x_1, \ldots, x_n) \in R^n$ is defined to be

$$w(x) = \sum_{i=1}^{n} a_{x_i}, \tag{2}$$

where $a_t = |t|$, with $t \in R$, as in (1). Similarly, the *Euclidean weight* uses $a_t = |t|^2$. We will denote both types of weight by $w(x)$; the context will make clear which is being discussed.

We wish to determine the linear codes of *constant weight*, i.e., codes for which there exists $L > 0$ with $w(x) = L$ for all nonzero $x \in C$. As above, $w(x)$ refers to a fixed choice of either Lee or Euclidean weight.

Observe that reduction mod p makes $\mathbb{Z}/(p)$ a module over $R = \mathbb{Z}/(p^2)$.

PROPOSITION 1. *Any linear code C is isomorphic, as an R-module, to a direct sum*

$$C \cong (\mathbb{Z}/(p))^{l_1} \oplus \left(\mathbb{Z}/(p^2)\right)^{l_2}. \tag{3}$$

A *linear automorphism* of C is any R-homomorphism $f : C \to C$ which is invertible. Note that this definition does not involve the weight function w, so that f need not be a code automorphism. However, if C has constant weight, then any linear automorphism f is a code automorphism. Denote the group of all linear automorphisms of C by $\mathrm{Aut}(C)$.

THEOREM 2. *For C as in* (3), $\mathrm{Aut}(C)$ *consists of all equivalence classes of matrices over R of the form*

$$A = \begin{pmatrix} M & N \\ pP & Q \end{pmatrix},$$

where M and Q are invertible. Two such matrices A, A' are equivalent if $M \equiv M' \bmod p$ and $N \equiv N' \bmod p$.

2. Orbit structures

The linear automorphism group $\mathrm{Aut}(C)$ acts on C and on $C^\sharp = \mathrm{Hom}_R(C, R)$, the linear dual of C. Our main interest is the action on $C^\sharp$. However, $C^\sharp \cong C$, so we will work directly with the action on C.

In the next theorem, we will denote elements of C as pairs $x = (x_{(1)}, x_{(2)})$, where $x_{(i)} \in \mathbb{Z}/(p^i)^{l_i}$, as in (3). An asterisk $*$ means the entry can assume any value; $p*$ means that every component of the entry is a multiple of p; u means that *at least one* component of the entry is a unit. We write e for the tuple $e = (1, 0, \ldots, 0)$.

THEOREM 3. *The orbits of* $\mathrm{Aut}(C)$ *on* C *are as in Table 1.*

Orbit	Representative	Size
$(*, u)$	$(0, e)$	$p^{l_1+l_2}(p^{l_2}-1)$
$(u, p*)$	$(e, 0)$	$(p^{l_1}-1)p^{l_2}$
$(0, pu)$	$(0, pe)$	$p^{l_2}-1$
$(0, 0)$	$(0, 0)$	1

TABLE 1. Orbits of $\mathrm{Aut}(C)$ on C.

3. Constant weight codes

A linear code $C \subset R^n$ can be viewed as an abstract R-module as in (3), equipped with an embedding in R^n. The embedding is given by n *coordinate functionals* $\lambda_1, \ldots, \lambda_n \in C^\sharp$. If C has a generator matrix G, then the columns of G are the values of the λ_i evaluated on a set of generators for C.

The main restriction on constant weight codes is that entire orbits of linear functionals must occur as coordinate functionals of C.

THEOREM 4. *Let* $C \subset R^n$ *be a linear code of constant weight, either Lee or Euclidean weight. If* $\lambda \in C^\sharp$ *occurs as a coordinate functional of* C, *then (up to* $\pm$ *signs) every other linear functional* μ *in the* $\mathrm{Aut}(C)$*-orbit of* λ *also occurs as a coordinate functional of* C.

PROOF. Given μ in the orbit of λ, there exists some $f \in \mathrm{Aut}(C)$ carrying λ to μ. On the other hand, f preserves weight (i.e., $w(f(x)) = w(x)$, for all $x \in C$), since C has constant weight. By the extension theorem [5], [6], f extends to a signed permutation automorphism of R^n. Thus $\pm\mu$ is another coordinate functional of C. □

A similar argument shows that $\pm\lambda$ and $\pm\mu$ occur with the same multiplicity.

REMARK 5. We caution the reader that Theorem 4 is a theorem only to the extent that the extension theorem holds for Lee or Euclidean weight. The extension theorem is *not* known for the general case of $R = \mathbb{Z}/(p^k)$. It holds for various small values of p^k where the conditions of [5] and [6] can be verified by hand.

4. Classification of constant weight codes: $p = 2$

A linear code of length n can always be viewed as a code of length $n+1$ by adding a zero entry, i.e., by enlarging the set of coordinate functionals $\lambda_1, \ldots, \lambda_n$ to include $\lambda_{n+1} = 0$. We call a linear code *nondegenerate* if it has no zero coordinate functionals.

THEOREM 6 (Carlet [2]). *Let C be a nondegenerate linear code of constant Lee weight over $R = \mathbb{Z}/(4)$. Then C is equivalent to the replication of a code D whose coordinate functionals consist of all the nonzero linear functionals on D.*

The linear codes C and D are isomorphic as R-modules, each of cardinality $2^{l_1}4^{l_2} = 2^{l_1+2l_2}$. The code D has length $|D|-1 = 2^{l_1+2l_2}-1$, while the code C has length $r(2^{l_1+2l_2}-1)$, for some positive integer r. Every nonzero element of D has Lee weight $L = |D| = 2^{l_1+2l_2}$, while every nonzero element of C has Lee weight rL.

Let us clarify some terminology. In the context of Lee or Euclidean weight, two linear codes of length n over R are *equivalent* if one can be obtained from the other by a signed permutation automorphism of R^n. This means the two codes have the same collections of coordinate functionals, up to $\pm$ signs. An r-fold *replication* of a code D of length n is a new code of length rn having the same coordinate functionals as D, but with each having multiplicity r.

PROOF. By Theorem 4, entire orbits of linear functionals (up to $\pm$ signs) must occur in the collection of coordinate functionals of C. Because C is nondegenerate, no zero functionals occur.

Referring to Table 1, let α, β, γ denote the number of times the orbits $(*, u)$, $(u, 2*)$, $(0, 2u)$, modulo $\pm$ signs (relevant for $(*, u)$ only), occur in the coordinate functionals of C.

For any $x \in R^n$, let $s_1(x) = |\{i : x_i = \pm 1\}|$ and $s_2(x) = |\{i : x_i = 2\}|$. Then $w(x) = s_1(x) + 2s_2(x)$. Note that $w(2x) = 2s_1(x)$.

Over $R = \mathbb{Z}/(4)$, any nonzero element of C has order 2 or 4. Suppose $x \in C$ has order 4. A consequence of constant Lee weight is that $w(x) = w(2x)$. It then follows that $s_1(x) = 2s_2(x)$. If y has order 2, then $s_1(y) = 0$, so that $w(y) = 2s_2(y)$. Because $2y = 0$, there is no additional restriction on $w(y)$.

Let $x = (0, e)$ and $y = (e, 0)$; x has order 4, while y has order 2. A detailed examination of the orbits in Table 1 reveals that

$$s_1(x) = 2^{l_1+2l_2-2}\alpha,$$
$$s_2(x) = 2^{l_1+l_2-2}(2^{l_2-1}-1)\alpha + (2^{l_1}-1)2^{l_2-1}\beta + 2^{l_2-1}\gamma,$$
$$s_2(y) = 2^{l_1+l_2-2}(2^{l_2}-1)\alpha + 2^{l_1+l_2-1}\beta.$$

From the constant weight conditions $w(x) = w(2x) = w(y)$, it follows that $s_1(x) = 2s_2(x) = s_2(y)$. We then conclude that $\beta = \gamma$ and $\alpha = 2\beta = 2\gamma$. Thus C is a β-fold replication of D. (Note that the orbit $(*, u)$ is effectively cut in half by the $\pm$ sign restriction. Having $\alpha = 2\beta$ restores the orbit to full size.) □

EXAMPLE 7. For $l_1 = l_2 = 1$, the smallest example occurs when $\alpha = 2$, $\beta = \gamma = 1$. A generating matrix has the form

$$G = \begin{pmatrix} 0 & 2 & 0 & 2 & 2 & 2 & 0 \\ 1 & 1 & 1 & 1 & 0 & 2 & 2 \end{pmatrix}.$$

The code has cardinality 8, length 7, and constant Lee weight 8.

Turn now to Euclidean weight, so that the weight function w has $a_2 = 4$, as in (2). An argument similar to that in the proof of Theorem 6 shows that $\alpha = 2\beta$ and $\gamma = (2^{l_1+l_2-2}+1)\beta$. This proves the next theorem.

THEOREM 8. *For a fixed isomorphism type* (3), *there exists a linear code D of constant Euclidean weight having minimal length. The code D is unique up to equivalence. The cardinality of D is $|D| = 2^{l_1}4^{l_2} = 2^{l_1+2l_2}$, and its length is $2^{l_1+2l_2} - 1 + 2^{l_1+l_2-2}(2^{l_2}-1)$. Every nonzero element of D has Euclidean weight $L = 2|D| = 2^{l_1+2l_2+1}$.*

Any nondegenerate linear code C of constant Euclidean weight and having isomorphism type (3) *is equivalent to an r-fold replication of D.*

EXAMPLE 9. If $l_1 = l_2 = 1$, then $\alpha = \gamma = 2\beta$. The smallest example has $\beta = 1$, $\alpha = \gamma = 2$. A generating matrix has the form

$$G = \begin{pmatrix} 0 & 2 & 0 & 2 & 2 & 2 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 2 & 2 & 2 \end{pmatrix}.$$

The code has cardinality 8, length 8, and constant Euclidean weight 16.

5. Classification of constant weight codes: p odd

When the prime p is odd, there are several surprises. One technical difference from the case of $p = 2$ is that $x = -x$ implies $x = 0$ when p

is odd. In contrast, $x = -x$ implies $x = 0$ or 2 in $\mathbb{Z}/(4)$. This affects the counting of orbits modulo $\pm$ signs.

Throughout this section $R = \mathbb{Z}/(p^2)$ with p an odd prime.

PROPOSITION 10. *A linear code has constant Lee weight over R if and only if it has constant Euclidean weight. The ratio of the weights is $p^2/3$.*

THEOREM 11. *Suppose C is a nondegenerate linear code over R of constant Lee or Euclidean weight. Then the isomorphism type (3) of C satisfies $l_2 = 0$ or $l_1 + l_2 \leq 2$.*

The code C is equivalent to an r-fold replication of a constant weight code D whose properties are listed in Table 2. The codes C and D are isomorphic as R-modules and of the same cardinality $|C| = |D| = p^{l_1+2l_2}$. The length and constant Lee weight of C are r times those of D.

When $l_2 = 0$, the coordinate functionals of D consist of all the nonzero linear functionals on D, modulo $\pm$ signs.

l_1	l_2	$\lvert D\rvert$	Length	Weight L
l_1	0	p^{l_1}	$(p^{l_1}-1)/2$	$p^{l_1}(p^2-1)/8$
0	1	p^2	$(p^3-2p+1)/2$	$p^3(p^2-1)/8$
1	1	p^3	$p(p^2-1)/2$	$p^3(p^2-1)/8$
0	2	p^4	$p^2(p^2-1)/2$	$p^4(p^2-1)/8$

TABLE 2. Properties of constant weight code D.

PROOF. We keep the notation from the proof of Theorem 6. If $l_2 = 0$, only orbit $(u, p*)$ can occur. Then $\alpha = \gamma = 0$, and β is arbitrary. When $l_2 > 0$, the constant weight condition implies that

$$ps_p(x) = (p-1)s_1(x),$$
$$ps_1(x) = s_p(y).$$

The second condition occurs only when $l_1 > 0$.

In terms of orbit contributions ($\pm$ signs are now relevant for all three types of orbits), we see that

$$s_1(x) = p^{l_1+2l_2-2}\alpha,$$
$$s_p(x) = (p^{l_1+2l_2-2} - p^{l_1+l_2-1})\alpha + (p^{l_1+l_2-1} - p^{l_2-1})\beta + p^{l_2-1}\gamma,$$
$$s_p(y) = (p^{l_1+2l_2-1} - p^{l_1+l_2-1})\alpha + p^{l_1+l_2-1}\beta.$$

From the condition $ps_1(x) = s_p(y)$ it follows that $\alpha = \beta$. From $ps_p(x) = (p-1)s_1(x)$ we obtain

$$(p^{l_1+l_2-2} - 1)\alpha + \gamma = 0.$$

Since $\alpha, \gamma \geq 0$, there are only zero solutions once $l_1 + l_2 > 2$.

The reader may verify the other claims and cases. □

EXAMPLE 12. Let $l_1 = 2$, $l_2 = 0$. Then $\alpha = \gamma = 0$, with β arbitrary. The shortest example has $\beta = 1$. Over $R = \mathbb{Z}/(9)$, a generating matrix has the form

$$G = \begin{pmatrix} 3 & 3 & 3 & 0 \\ 0 & 3 & -3 & 3 \end{pmatrix}.$$

The code has cardinality 9, length 4, constant Lee weight 9, and constant Euclidean weight 27.

Over $R = \mathbb{Z}/(25)$, a generating matrix has the form

$$G = \begin{pmatrix} 5 & 5 & 5 & 5 & 5 & 10 & 10 & 10 & 10 & 10 & 0 & 0 \\ 0 & 5 & 10 & -10 & -5 & 0 & 5 & 10 & -10 & -5 & 5 & 10 \end{pmatrix}.$$

The code has cardinality 25, length 12, constant Lee weight 75, and constant Euclidean weight 625.

EXAMPLE 13. Let $l_1 = 0$, $l_2 = 1$. Then $\beta = 0$ and $(p-1)\alpha = p\gamma$. The shortest example has $\alpha = p$, $\gamma = p - 1$. Over $R = \mathbb{Z}/(9)$, a generating matrix has the form

$$G = \begin{pmatrix} 1 & 2 & 4 & 1 & 2 & 4 & 1 & 2 & 4 & 3 & 3 \end{pmatrix}.$$

The code has cardinality 9, length 11, constant Lee weight 27, and constant Euclidean weight 81.

Over $R = \mathbb{Z}/(25)$, a generating matrix G has one row, consisting of 5 copies of

$$1 \quad 2 \quad 3 \quad 4 \quad 6 \quad 7 \quad 8 \quad 9 \quad 11 \quad 12$$

concatenated with 4 copies of

$$5 \quad 10.$$

The code has cardinality 25, length 58, constant Lee weight 375, and constant Euclidean weight 3125.

EXAMPLE 14. Let $l_1 = 1$, $l_2 = 1$. Then $\gamma = 0$ and $\alpha = \beta$. The shortest example has $\alpha = \beta = 1$. Over $R = \mathbb{Z}/(9)$, a generating matrix has the form

$$G = \begin{pmatrix} 0 & 3 & -3 & 0 & 3 & -3 & 0 & 3 & -3 & 3 & 3 & 3 \\ 1 & 1 & 1 & 2 & 2 & 2 & 4 & 4 & 4 & 0 & 3 & -3 \end{pmatrix}.$$

The code has cardinality 27, length 12, constant Lee weight 27, and constant Euclidean weight 81.

EXAMPLE 15. Let $l_1 = 0$, $l_2 = 2$. Then $\beta = \gamma = 0$, with α arbitrary. The shortest example has $\alpha = 1$. Over $R = \mathbb{Z}/(9)$, a generating matrix has the form

$$G = \begin{pmatrix} 1 & 2 & 4 & 0 & 0 & 0 & 3 & 3 & 3 & -3 & -3 & -3 \\ * & * & * & 1 & 2 & 4 & 1 & 2 & 4 & 1 & 2 & 4 \end{pmatrix},$$

where $*$ indicates 9 entries, running over the elements of $\mathbb{Z}/(9)$. The code has cardinality 81, length 36, constant Lee weight 81, and constant Euclidean weight 243.

6. Possible generalizations

The major ideas in this extended abstract generalize to any finite commutative chain ring (i.e., local, with principal ideals). However, there are serious technical and notational difficulties to be overcome in order to understand the orbit structure of $\mathrm{Aut}(C)$ on C and to manipulate the equations arising from the constant weight condition.

ACKNOWLEDGMENTS. I thank the referee and C. Carlet for their advice on revising this extended abstract.

References

[1] A. Bonisoli, *Every equidistant linear code is a sequence of dual Hamming codes*, Ars Combin. **18** (1984), 181–186.

[2] C. Carlet, *One-weight $\mathbb{Z}_4$-linear codes*, preprint, 1998.

[3] H. N. Ward, *A bound for divisible codes*, IEEE Trans. Inform. Theory **38** (1992), 191–194.

[4] H. N. Ward and J. A. Wood, *Characters and the equivalence of codes*, J. Combin. Theory, series A **73** (1996), 348–352.

[5] J. A. Wood, *Weight functions and the extension theorem for linear codes over finite rings*, Finite Fields: Theory, Applications and Algorithms (R. C. Mullin and G. L. Mullen, eds.), Contemp. Math., vol. 225, Amer. Math. Soc., Providence, 1999, pp. 231–243.

[6] ———, *Factoring the semigroup determinant of a finite commutative chain ring*, Lecture Notes in Comput. Sci., Springer-Verlag, (to appear).

On the Covering Radius of $\mathbb{Z}_4$-Codes and Their Lattices

Toru Aoki*, Philippe Gaborit†, Masaaki Harada‡
Michio Ozeki§ and Patrick Solé¶

Abstract

In this talk, we investigate the covering radius of codes over $\mathbb{Z}_4$, especially self-dual codes. We show that covering radii of codes over $\mathbb{Z}_4$ are related to ones of binary nonlinear codes and lattices obtained by the Gray map and Construction A_4, respectively. We give several upper and lower bounds of covering radii, including $\mathbb{Z}_4$-analogues of the sphere-covering bound, the Delsarte bound , the packing radius bound and the redundancy bound. The covering radii of some self-orthogonal codes and self-dual codes are obtained, in particular, we show that any Type II code of length 24 extremal for the Euclidean weight has covering radius 8 with respect to Euclidean distance. We also give lower and upper bounds for the covering radii of the 23 Niemeier lattices with minimum norm 2.

1 Definitions and Properties

For codes over $\mathbb{Z}_4$, there are three different distances, namely Hamming, Lee and Euclidean distances. Thus we can define covering radius for the three distances. However the Lee distance and the Euclidean distance have remarkable applications to binary nonlinear codes and unimodular lattices, respectively. Hence, we consider only covering radius with respect to Lee and Euclidean distances. First we define the *covering radius* of a code C over $\mathbb{Z}_4$ by

$$r_L(C) = \max_{u \in \mathbb{Z}_4^n} \{\min_{c \in C} d_L(u,c)\} \text{ and } r_E(C) = \max_{u \in \mathbb{Z}_4^n} \{\min_{c \in C} d_E(u,c)\},$$

respectively. It is easy to see that $r_L(C)$, $r_E(C)$ are the minimum values r_L, r_E such that

$$\mathbb{Z}_4^n = \cup_{c \in C} S_{r_L}(c) \text{ and } \mathbb{Z}_4^n = \cup_{c \in C} S_{r_E}(c)$$

*Department of Mathematical Sciences, Yamagata University, Yamagata 990–8560, Japan.
†INRIA - MSCS Department, University of Illinois at Chicago, Chicago, IL, 60607, USA. Email address: gaborit@math.uic.edu.
‡Department of Mathematical Sciences, Yamagata University, Yamagata 990–8560, Japan. Email address: harada@kszaoh3.kj.yamagata-u.ac.jp.
§Department of Mathematical Sciences, Yamagata University, Yamagata 990–8560, Japan. Email address: ozeki@kszaoh3.kj.yamagata-u.ac.jp.
¶CNRS-I3S, ESSI, Route des Colles, 06 903 Sophia Antipolis, France. Email address: sole@essi.fr.

respectively, where

$$S_{r_L}(u) = \{v \in \mathbb{Z}_4^n \mid d_L(v,u) \leq r_L\} \text{ and } S_{r_E}(u) = \{v \in \mathbb{Z}_4^n \mid d_E(v,u) \leq r_E\}.$$

Here u is an element of $\mathbb{Z}_4^n$.

Lemma 1.1 *For a code C over $\mathbb{Z}_4$, $r_L(C) \leq r_E(C)$.*

Proof. Follows from $d_L(x,y) \leq d_E(x,y)$ for any two vectors x and y. □

Proposition 1.2 *Let C be a code over $\mathbb{Z}_4$ and $\phi(C)$ the Gray map image of C. Then $r_L(C) = r(\phi(C))$.*

Proof. From the definition of $r_L(C)$, there is a codeword c of C with $d_L(u,c) \leq r_L(C)$ for any vector $u \in \mathbb{Z}_4^n$. Since $d_L(u,c) = d(\phi(u),\phi(c))$, we have $r(\phi(C)) \leq r_L(C)$. ϕ is one-to-one thus $r(\phi(C)) = r_L(C)$. □

We now relate the covering radius with respect to Lee or Euclidean distance to the weight in cosets. The translate

$$u + C = \{u + c \mid c \in C\}$$

is called a coset of C where u is a vector of $\mathbb{Z}_4^n$. A vector of minimum weight in a coset is called a *coset leader*. The following proposition is immediate but useful. The proof is omitted.

Proposition 1.3 *The covering radius of C with respect to Lee (resp. Euclidean) distance is the largest minimum Lee (resp. Euclidean) weight among all cosets.*

2 Upper and Lower Bounds

We prove several upper and lower bounds on covering radii of codes over $\mathbb{Z}_4$, which are enumerated without proof in the following.

2.1 Lower Bounds

The sphere-covering bound holds for binary nonlinear codes as well as linear codes. We have a $\mathbb{Z}_4$-analogy of the sphere-covering bound.

Proposition 2.1 (Sphere-Covering Bounds) *For any code C of length n over $\mathbb{Z}_4$,*

$$\frac{2^{2n}}{|C|} \leq \sum_{i=0}^{r_L(C)} \binom{2n}{i} \quad \text{and} \quad \frac{2^{2n}}{|C|} \leq \sum_{i=0}^{r_E(C)} V_i,$$

where

$$\sum_{i=0}^{4n} V_i x^i = (1 + 2x + x^4)^n.$$

We say that the first and second bounds are sphere-covering bounds with respect to Lee and Euclidean distance, respectively.

We now prove a Packing Radius Bound-like theorem (in the sense that the bounds obtained are close to half the minimum considered distance) which is true for linear and non-linear $\mathbb{Z}_4$ codes.

Theorem 2.2 (Packing Radius Bounds) *Let C be a code of length n over $\mathbb{Z}_4$, let d_L and d_E denote respectively the minimum Lee and Euclidean distances of C then*

$$\begin{aligned} r_E(C) &\geq 4\lfloor d_E/8 \rfloor, \text{ and} \\ r_L(C) &\geq \lfloor d_L/2 \rfloor. \end{aligned}$$

For a code C over $\mathbb{Z}_4$, let d_i denote the minimum distance of the related binary linear codes $C^{(i)}$ ($i = 1$ and 2).

Theorem 2.3 *If $d_1 \geq 4$ and $d_2 \geq 2$ then*

$$\begin{aligned} r_E(C) &\geq 4\min(\lfloor d_1/4 \rfloor, \lfloor d_2/2 \rfloor), \text{ and} \\ r_L(C) &\geq 2\min(\lfloor d_1/4 \rfloor, \lfloor d_2/2 \rfloor). \end{aligned}$$

2.2 Upper Bounds

We consider an upper bound for the covering radius of codes over $\mathbb{Z}_4$. Let C be a code over $\mathbb{Z}_4$ and

$$s(C^\perp) = |\{i \mid A_i(C^\perp) \neq 0, i \neq 0\}|$$

where $A_i(C^\perp)$ be the number of codewords of Lee weight i in $C^\perp$. Let B be a nonlinear binary code and let $s'(B)$ be the distinct numbers of nonzero distances in the distance distribution of the formally defined dual code of B, which is obtained from the distance distribution of B by the binary MacWilliams transform (see [6] for the definition of the distance distributions). Delsarte [6] showed that the covering radius $r(B)$ of B is bounded by $r(B) \leq s'(B)$. This is known as the Delsarte bound.

Theorem 2.4 (Delsarte Bound) *Let C be a code over $\mathbb{Z}_4$ then $r_L(C) \leq s(C^\perp)$.*

By the above two bounds, it is possible to determine the covering radius for some codes. For example, consider the unique Type II code $\mathcal{D}_4^\oplus$ in [4] of length 4. By the sphere-covering bound, $r_L(C) \geq 2$ for any self-dual code C of length 4. Since $s(\mathcal{D}_4^\oplus) = 2$, we have $r_L(\mathcal{D}_4^\oplus) = 2$.

Now we give a redundancy bound for codes over $\mathbb{Z}_4$ expressed in terms of Lee and Euclidian distances.

Theorem 2.5 (Redundancy Bound) *Let C be a code over $\mathbb{Z}_4$ of type $4^{k_1}2^{k_2}$ then*

$$\begin{aligned} r_L(C) &\leq 2(n - k_1 - k_2) + k_2, \text{ and} \\ r_E(C) &\leq 4(n - k_1 - k_2) + k_2. \end{aligned}$$

Remark. It is easy to see that these bounds are tight for codes with generator matrices:

$$\begin{pmatrix} I_{n-k_1-k_2} & 0 & 0 \\ 0 & 2I_{k_2} & 0 \end{pmatrix}.$$

There arises a natural question for the covering radius with respect to Euclidean distance, namely, is such a covering radius related to the covering radius of a unimodular lattice obtained by Construction A_4?

Theorem 2.6 *Let C be a code of length n over $\mathbb{Z}_4$. Let $A_4(C)$ be the n-dimensional lattice constructed from C by Construction A_4. Then*

$$r_E(C) \leq 4(\rho(A_4(C)))^2 \leq r_E(C) + \frac{n}{4} + \sqrt{n r_E(C)},$$

where $\rho(L)$ is the covering radius of a lattice L.

As an example we consider the four inequivalent Type II codes of length 8 of [4], namely $\mathcal{O}_8$, $\mathcal{K}_8$, $\mathcal{Q}_8$ and $\mathcal{K}_8'$.

Proposition 2.7 *The Euclidean covering radius of the four inequivalent Type II codes of length 8 is 4.*

As the above example, often we can determine the exact value of the covering raidus using the bounds in this section.

3 Covering Radius of Self-Dual Codes over $\mathbb{Z}_4$

3.1 Numerical Results

In this subsection, we give the numerical results of covering radii of some self-orthogonal codes and self-dual codes over $\mathbb{Z}_4$, some of these results are theoritical and some are obtained with help of computer. The program we used, is based on the coset distribution of a code as described in Th. 2.5 and allowed us to compute the covering radius of a code of length up to 16, in a few hours. It also allowed us to find lower bounds for the covering radii of codes with greater lengths. Conway and Sloane [4] gave the list of all self-dual codes of length up to 9 and a number of examples of self-orthogonal codes.

We list in Table 1 the values of covering radii $r_L(C)$ and $r_E(C)$ with respect to the Lee and Euclidean distances for some self-orthogonal codes and self-dual codes given in [4] (see [4] for the notations of codes), together with the values $s(C^\perp)$.

We now prove the following result:

Proposition 3.1 *If C is a Type II code of length 8 then $r_L(C) = r_E(C) = 4$ and $r_L(C)$ attends the sphere-covering bound.*

As described in [7], $\phi(\mathcal{O}_8)$ is the Nordstrom-Robinson code. Combining Proposition 1.2 with Table 1, the precedent result gives a theoritical proof of the following:

Corollary 3.2 *The covering radius of the Nordstrom-Robinson code is 4.*

Table 1: Covering Radii of Self-Orthogonal Codes over $\mathbb{Z}_4$

Code C	type	$r_L(C)$	$r_E(C)$	$s(C^\perp)$
$\mathcal{D}_4$	4^1	4	6	4
$\mathcal{D}_6$	4^2	5	8	10
$\mathcal{D}_8$	4^3	6	10	8
$\mathcal{D}_{10}$	4^4	7	12	16
$\mathcal{D}_{12}$	4^5	10	16	12
$\mathcal{D}_4^\circ$	$4^1 2^1$	4	4	4
$\mathcal{D}_6^\circ$	$4^2 2^1$	5	7	8
$\mathcal{E}_7$	4^3	5	8	10
$\mathcal{A}_1$ (self-dual)	2^1	1	1	1
$\mathcal{D}_4^\oplus$ (Type II)	$4^1 2^2$	2	4	2
$\mathcal{E}_7^+$ (self-dual)	$4^3 2^1$	5	7	5
$\mathcal{E}_8$ (self-dual)	4^4	4	8	6
$\mathcal{O}_8$ (Type II)	4^4	4	4	4
$\mathcal{K}_8$ (Type II)	$4^1 2^6$	4	4	4
$\mathcal{Q}_8$ (Type II)	$4^3 2^2$	4	4	6
$\mathcal{K}_8'$ (Type II)	$4^2 2^4$	4	4	6

We also prove the following general result on the bound of extremal Euclidean codes of length 24:

Proposition 3.3 *The Euclidean covering radius of any Euclidean extremal Type II code of length 24 is 8.*

Proof. Let C be an extremal Euclidean Type II code of length 24. It is known that $A_4(C)$ is the Leech lattice whose covering radius is $\sqrt{2}$ (cf. [5]). The minimum Euclidean distance has to be 16, therefore combining Theorems 2.6 and 2.2, we obtain:

$$8 \leq r_E(C) \leq 8.$$

□

3.2 Niemeier Lattices

The 24-dimensional 23 even unimodular lattices with minimum norm 2 are called Niemeier lattices. It was shown in [1] that all the Niemeier lattices can be constructed from some Type II codes over $\mathbb{Z}_4$ by Construction A_4. Let $C(N)$ be the Type II code given in [1] corresponding to a Niemeier lattice N. Obtaining lower bounds on covering radii with respect to Euclidean distance of the 23 Type II codes, we have lower bounds on covering radii of the Niemeier lattices. The results are listed in Table 2 where lower bounds on $r_E(C(N))$ are obtained by Proposition 2.1 or Theorem 2.3, or computer search and Proposition 3.3, and lower bounds on $\rho(N)$ follow therefrom by Theorem 2.6 . The upper bounds are obtained by computing the covering radius of the root sublattice using the information in [5, Chap. 4]. The following lower bound is obtained in [3].

Theorem 3.4 *(Borcherds) If the Niemeier lattice of root system N has kissing number $24h$ then*

$$\rho(N) \geq 2 + \frac{2}{h}.$$

Table 2: The Covering Radii of the Niemeier Lattices and Type II Codes

Lattice N	$\rho(N)^2$	$r_E(C(N))$	$d_1(C(N))$	$d_2(C(N))$	h	$2+2/h$
D_{12}^2	$3-6$	≥ 12	8	8	22	2.091
$A_{15}D_9$	$2.5-6.25$	≥ 10	8	8	16	2.125
D_{24}	$3-6$	≥ 12	24	2	46	2.435
D_8^3	$3-6$	≥ 12	4	4	14	2.143
A_8^3	$2.25-\sim 6.67$	≥ 9	4	4	9	2.222
$A_7^2D_5^2$	$2.25-\sim 11.17$	≥ 9	8	8	8	2.25
A_6^4	$2.25-\sim 6.86$	≥ 9	8	8	7	2.143
A_3^8	$2.5-8$	≥ 10	4	4	4	2.5
A_1^{24}	$2-24$	≥ 8	8	8	2	3
A_4^6	$2.25-7.2$	≥ 9	8	8	5	2.4
A_2^{12}	$2.25-16$	≥ 9	8	8	3	2.667
A_{24}	$2.5-6.24$	≥ 10	4	2	25	2.04
A_{12}^2	$2.5-5.54$	≥ 10	8	2	13	2.154
$A_{17}E_7$	$2.5-5.74$	≥ 10	4	2	18	2.111
$D_{10}E_7^2$	$2.5-5.5$	≥ 10	4	2	18	2.111
$A_{11}D_7E_6$	$2.5-5.81$	≥ 10	4	2	12	2.167
E_6^4	$2.5-24$	≥ 10	8	2	12	2.167
$A_9^2D_6$	$2.5-6.5$	≥ 10	4	2	10	2.2
D_6^4	$3-6$	≥ 12	12	2	10	2.2
$A_5^4D_4$	$2.5-7$	≥ 10	8	2	6	2.333
D_4^6	$3-6$	≥ 12	8	2	6	2.333
E_8^3	3	≥ 12	4	4	30	2.666
$D_{16}E_8$	$3-5$	≥ 12	4	2	30	2.666

References

[1] A. Bonnecaze, P. Gaborit, M. Harada, M. Kitazume, P. Solé, Niemeier Lattices and Type II codes over $\mathbb{Z}_4$, *to appear in Discrete Math.*

[2] A. Bonnecaze, P. Solé, C. Bachoc and B. Mourrain, Type II codes over $\mathbb{Z}_4$, *IEEE Trans. Inform. Theory* **43** (1997), 969–976.

[3] R. Borcherds, The Leech lattice, Proc. of London Math Soc. 398 (1985) 365-376.

[4] J.H. Conway and N.J.A. Sloane, Self-dual codes over the integers modulo 4, *J. Combin. Theory Ser. A* **62** (1993), 30–45.

[5] J.H. Conway and N.J.A. Sloane, *Sphere Packing, Lattices and Groups (2nd ed.)*, Springer-Verlag, New York, 1993.

[6] P. Delsarte, Four fundamental parameters of a code and their combinatorial significance, *Inform. Contr.* **23** (1973), 407–438.

[7] A.R. Hammons, Jr., P.V. Kumar, A.R. Calderbank, N.J.A. Sloane and P. Solé, The $\mathbb{Z}_4$-linearity of Kerdock, Preparata, Goethals and related codes, *IEEE Trans. Inform. Theory* **40** (1994), 301–319.

Permutation Groups of Extended Cyclic Codes over Galois Rings

Tom Blackford
The Ohio State University
tblackfd@math.ohio-state.edu

In an important paper, Hammons et. al. [3] showed how to construct well known binary codes like the Kerdock, Preparata and Delsarte-Goethals codes by applying the Gray map to extended cyclic codes over Z_4. From this comes the motivation to study cyclic and extended cyclic codes over the integer residue rings Z_{p^a}, where p is prime. In previous literature, these codes are characterized by generating polynomials. As an alternative method, we characterize them via a Galois ring transform (as done by Rajan and Siddihi, [5]) and the concept of multiple defining sets. Using this characterization, we can determine the exact permutation group of extended cyclic codes over Z_{p^a} of length p^m that are affine-invariant. In particular, it is possible to calculate the permutation groups of the quaternary analogues of the Preparata, Goethals, and Goethals-Delsarte codes without using their binary images under the Gray map. In at least one case, the permutation group of the quaternary Goethals-Delsarte code is larger than the group described in [3].

1 Galois Rings

Let p be prime, a and m be positive integers, and $n = p^m - 1$. Recall that the Galois ring $GR(p^a, m)$ of characterisitic p^a and dimension m is an algebraic extension $Z_{p^a}[\zeta]$ of Z_{p^a}, where ζ is a primitive nth root of unity that is also a root of a monic irreducible polynomial $m(X)$ in $Z_{p^a}[X]$ of degree m. (Familar examples include the finite field F_{p^m}, which is $GR(p, m)$, and Z_{p^a}, which is $GR(p^a, 1)$.) ζ is called a **primitive element** of $GR(p^a, m)$. $R = GR(p^a, m)$ is a local ring with maximal ideal pR, and a complete set of

coset representitives of R modulo pR is given by the set

$$\mathcal{T}_m = \{0, 1, \zeta, \ldots, \zeta^{n-1}\},$$

called the **Teichmuller set of representitives.** Note the nonzero elements of $\mathcal{T}_m$ are the nth roots of unity. Let $\mu : R \to F_{p^m}$ be the reduction map modulo p; extend μ to a map from $R[X]$ to $F_{p^m}[X]$ in the usual way. Observe $\mu : \mathcal{T}_m \to F_{p^m}$ is a bijection. Each element $r \in GR(p^a, m)$ has a unique p-adic expansion

$$r = \zeta_0 + p\zeta_1 + \cdots + p^{a-1}\zeta_{a-1},$$

where $\zeta_0, \zeta_1, \ldots, \zeta_{a-1} \in \mathcal{T}_m$. Denote $\overline{r} = \mu(r)$.

2 Extended Cyclic Codes

Now suppose $n = p^m - 1$. Since $\mathcal{T}_m$ consists of 0 and the nth roots of unity, $\mathcal{T}_m$ may be used as an index set for the coordinates of a vector $\mathbf{c}$ of length $p^m = n + 1$ over any alphabet:

$$\mathbf{c} = (c_x)_{x \in \mathcal{T}_m} = (c_0, c_1, c_\zeta, \ldots, c_{\zeta^{n-1}}).$$

If $[0, n] = \{0, 1, \ldots, n\}$, let $T_a \subseteq T_{a-1} \subseteq \cdots \subseteq T_1 \subseteq [0, n]$ be (possibly empty) subsets that are each unions of p-cyclotomic cosets modulo n (i.e., if $s \in T_j$, $s \neq n$, then $ps \pmod n \in T_j$ for $1 \leq j \leq a$).

Definition 1. Define the **extended cyclic code** $\mathcal{C}$ over Z_{p^a} of length p^m with defining sets $(T_1, \ldots, T_a)$ to be the set of all vectors $(c_x)_{x \in \mathcal{T}_m} \in (Z_{p^a})^{n+1}$ such that

$$\sum_{x \in \mathcal{T}_m} c_x x^s \equiv 0 \pmod{p^j} \quad \forall s \in T_j,$$

for $1 \leq j \leq a$, with the convention that $0^0 = 1$.

Note that the code of length n obtained by deleting the 0 coordinate of $\mathcal{C}$ will be a cyclic code.

Example 1. Let $m \geq 3$ be odd. The quaternary Goethals code $\mathcal{G}(m)$ is defined in [3] as the set of vectors $(c_x)_{x \in \mathcal{T}_m}$ of length 2^m over Z_4 that are orthogonal to the rows of the parity check matrix

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & \zeta & \cdots & \zeta^{n-1} \\ 0 & 2 & 2\zeta^3 & \cdots & 2\zeta^{3(n-1)} \end{bmatrix}$$

where ζ is a primitive element of $GR(4,m)$. Alternatively, $\mathcal{G}(m)$ is the extended cyclic code over Z_4 of length 2^m with defining sets (T_1, T_2), with

$$T_1 = \{0\} \cup cl_2(1) \cup cl_2(3),$$

$$T_2 = \{0\} \cup cl_2(1),$$

$$cl_2(s) = \{s, 2s, \ldots, 2^{m-1}s \pmod{n}\}.$$

Example 2. The quaternary Preparata code $\mathcal{P}(m)$ is defined in [3] as the set of vectors of length 2^m over Z_4 that are orthogonal to the rows of the parity check matrix

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & \zeta & \cdots & \zeta^{n-1} \end{bmatrix}$$

where ζ is a primitive element of $GR(4,m)$. Alternatively, $\mathcal{P}(m)$ is the extended cyclic code over Z_4 of length 2^m with defining sets (T_1, T_2), where

$$T_1 = T_2 = \{0\} \cup cl_2(1).$$

Example 3. Supoose $m \geq 3$ is odd, $m = 2t+1$, $1 \leq r \leq t$, $\delta = \frac{m+1}{2} - r$. The quaternary Goethals-Delsarte code $\mathcal{GD}(m,\delta)$ is defined to be the extended cyclic code over Z_4 of length 2^m having defining sets (T_1, T_2), where

$$T_1 = \{0\} \cup cl_2(1) \cup \bigcup_{j=1}^{r} cl_2(1+2^j),$$

$$T_2 = \{0\} \cup cl_2(1).$$

(Note that then $r = 1$, this code is just $\mathcal{G}(m)$.) It is shown in [3] that the binary image of $\mathcal{GD}(m,\delta)$ under the Gray map has the same parameters and weight distribution as the original binary Goethals-Delsarte code $GD(m+1,\delta)$. In the same paper, it is shown that the Gray image of $\mathcal{GD}(m,\delta)^{\perp}$ is in fact the original binary Delsarte-Goethals codes $DG(m+1,\delta)$.

Using the concept of defining sets, it is straight forward to determine the size and dual of extended cyclic codes.

3 Permutation Groups

Recall that a permutation on any set S is a bijection $\sigma : S \to S$. If σ is any permutation of the elements of F_{p^m}, there exists a unique permutation polynomial $f_\sigma \in F_{p^m}[X]$ such that

$$\sigma(\alpha) = f_\sigma(\alpha) \quad \forall \alpha \in F_{p^m}.$$

Any permutation of F_{p^m} induces a permutation on the Teichmuller set $\mathcal{T}_m$. Define $f_\sigma^{(L)}(X) \in \mathcal{T}_m[X]$ to be the unique polynomial such that $\mu(f_\sigma^{(L)}(X)) = f_\sigma(X)$. When a=2, the corresponding permutation on $\mathcal{T}_m$ is given by

$$\sigma(x) = (f_\sigma^{(L)}(x))^{p^m} \quad \forall x \in \mathcal{T}_m.$$

(This follows from the fact that if $r \in GR(p^2, m)$, then $r^{p^m} \in \mathcal{T}_m$.)

Definition 2. Let $e|m$. Then the **affine group** over F_{p^e}, denoted $AGL(m/e, p^e)$, is the set of all permutations σ of F_{p^m} whose permutation polynomials are of the form

$$f_\sigma(X) = \sum_{i=0}^{(m/e)-1} a_i X^{p^{ei}} + b,$$

where $a_i, b \in F_{p^m}$. The **semi-affine group** over F_{p^e}, denoted $A\Gamma L(m/e, p^e)$, is the group

$$< AGL(m/e, p^e), \theta >$$

where θ is the Frobenius automorphism defined by $f_\theta(X) = X^p$.

A permutation σ of F_{p^m} acts on a vector $\mathbf{c}$ of length p^m by permuting the coordinates. $Per(\mathcal{C})$, the permutation group of the code $\mathcal{C}$, is the group of all permutations of the coordinates of $\mathcal{C}$ which preserve the code (i.e., if $\sigma \in Per(\mathcal{C})$ and $\mathbf{c} \in \mathcal{C}$, then $\sigma(\mathbf{c}) \in \mathcal{C}$). Let $\mathcal{C}$ be a code (over any alphabet) of length p^m. Then its coordinates may be indexed by the elements α of F_{p^m} (or $\mathcal{T}_m$). We define $\mathcal{C}$ to be **affine-invariant** if $AGL(1, p^m) \subseteq Per(\mathcal{C})$ (i.e., it is invariant under any permutation $\alpha \mapsto a\alpha + b$, where $a, b \in F_{p^m}$ and $a \neq 0$).

As a generalization of Theorem 5 and Corollary 2 of [1] and the fact that codes over Z_{p^a} are invariant under the Frobenius map, we have

Theorem 1 *Let $\mathcal{C}$ be a nontrivial affine-invariant extended cyclic code over Z_{p^a} of length p^m. Then there exists a divisor e of m such that $Per(\mathcal{C}) = A\Gamma L(m/e, p^e)$.*

4 Main Results

Extended cyclic codes over F_p of length p^m which are affine-invariant were classified by Kasami et. al. in [4], using a combinatorial property of their defining set. Berger and Charpin [1] developed other combinatorial properties to completely determine their permutation groups. These results can be generalized to give information about the permutation groups of affine-invariant extended cyclic codes over the integer residue ring Z_{p^a}. Here we consider only the case $p^a = 4$.

Charpin [2] developed the following partial order on the set $S = [0, n]$. For $s \in S$, let

$$s = \sum_{i=0}^{m-1} s_i p^i, \quad 0 \leq s_i \leq p-1,$$

be its p-adic expansion. If $s, t \in S$, define

$$s \preceq t \iff s_i \leq t_i \ \ \forall i, 0 \leq i \leq m-1.$$

If $I \subseteq S$ satisfies the property that

$$s \in I, t \preceq s \implies t \in I,$$

then I is called a **lower ideal** of S. Finally, if $s, k \in [1, n-1]$, then

$$M_m(s,k) := |\{(i,j) : i, j \preceq s, i < j, i + j \equiv k \pmod{n}\}|.$$

For example, if $p = 2, s = 3$, and $m = 3$, then $M_m(3,k) = 1$ if $k = 1, 2, 4, 5$, $M_m(3,3) = 2$, and $M_m(3,k) = 0$ for all other values of k.

Theorem 2 (Kasami, [4]) *If $C \subseteq (F_p)^{n+1}$ is an extended cyclic code over F_p of length p^m with defining set T, then C is affine-invariant if and only if T is a lower ideal of $[0, n]$.*

(Note: p may be replaced with any power of p.) We have generalized this result for codes over Z_4.

Theorem 3 *If C is an extended cyclic code over Z_4 of length 2^m with defining sets (T_1, T_2), then C is affine-invariant if and only if*

1. *T_1 and T_2 are lower ideals of $[0, n]$*
2. *$s \in T_2, M_m(s,k) \not\equiv 0 \pmod 2 \implies k \in T_1$.*

In particular, this result shows that while binary extended BCH codes are always affine-invariant, their Hensel lifts may not be. We may use this theorem to give an alternate proof of the following result found in [3].

Corollary 1 *The quaternary Preparata, Goethals, and Goethals-Delsarte codes are affine-invariant.*

Proof: It is easy to check that the defining sets of these codes are lower ideals of $[0, 2^m - 1]$. For all of these codes, $T_2 = \{0\} \cup cl_2(1)$, and $M_m(2^\lambda, k) = 0$ for all $k \neq 0$ or 2^λ for any integer λ, and clearly $0, 2^\lambda \in T_2 \subseteq T_1$. □

The following is a generalization of a lemma that appears in Berger and Charpin [1].

Lemma 1 *Suppose σ is a permutation of F_{p^m} such that $\sigma(0) = 0$, and let $f_\sigma(X)$ be its permutation polynomial, and suppose C is an extended cyclic code over Z_{p^2} of length p^m with defining sets (T_1, T_2). Furthermore, suppose that for any $s \in [1, n]$,*

$$(f_\sigma^{(L)}(X))^{sp^m} \equiv \sum_{i=1}^{n-1} a_{s,i} X^i + p \sum_{i=1}^{n-1} b_{s,i} X^i \pmod{X^{p^m} - X}.$$

where $a_{i,s}, b_{i,s} \in \mathcal{T}_m$. Then $\sigma \in Per(C)$ if and only if

1. *$s \in T_2, a_{s,i} \neq 0 \implies i \in T_2$*
2. *$s \in T_2, b_{s,i} \neq 0 \implies i \in T_1$*
3. *$s \in T_1, a_{s,i} \neq 0 \implies i \in T_1$.*

Using this lemma, it is easy to construct codes whose permutation group is larger than $A\Gamma L(1, p^m)$.

Example 4. Let $\mathcal{C}$ be the extended cyclic code over Z_4 of length 16 with parity check matrix

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & \zeta & \cdots & \zeta^{14} \\ 0 & 2 & 2\zeta^5 & \cdots & 2\zeta^{5(14)} \end{bmatrix}$$

where ζ is a primitive element of $GR(4,4)$. The defining sets of $\mathcal{C}$ are

$$T_1 = \{0,1,2,4,8,5,10\},$$

$$T_2 = \{0,1,2,4,8\}$$

It is easy to check that these sets satisfy the conditions of Theorem 3, so $\mathcal{C}$ is affine-invariant. Now choose $\beta \in \mathcal{T}_4$ so that $N(\overline{\beta}) \neq 1$, where N is the norm map from F_{2^4} to F_{2^2}. Then the polynomial

$$f_\beta(X) = X - \overline{\beta}X^4$$

is a permutation polynomial of F_{2^4} (see [1]), and its corresponding permutation σ_β is in $A\Gamma L(2,2^2)$, and not in $A\Gamma L(1,2^4)$. Note

$$\begin{aligned} (X-\beta X^4)^{2^4} &\equiv X + \beta X^4 + 2\sqrt{\beta}X^{5(2^3)} \pmod{X^{16}-X} \\ (X-\beta X^4)^{5(2^4)} &\equiv X^5 + \beta X^8 + \beta^4 X^2 + \beta^5 X^5 \pmod{2, X^{16}-X} \end{aligned}$$

It is easy to check from these equations that σ_β satisfies the conditions of Lemma 1. Thus $A\Gamma L(2,2^2) \subseteq Per(\mathcal{C})$. $A\Gamma L(4,2)$ is *not* in $Per(\mathcal{C})$, since this would then imply that the binary extended cyclic code of length 16 with defining set T_1 is invariant under this group, a contradiction since it is not a Reed-Muller code. Thus, $Per(\mathcal{C}) = A\Gamma L(2,2^2)$.

The Goethals-Delsarte code $\mathcal{GD}(m,\delta)$ is invariant under the group generated by $A\Gamma L(1,2^m)$ and the negation map ($\mathbf{c} \mapsto$ - $\mathbf{c}$). Hammons et. al. claim in [3] that this is the full automorphism group of $\mathcal{GD}(m,\delta)$ for odd $m \geq 5$. Unfortunately, this is incorrect for certain values of m and δ. We give the following counterexample.

Example 5. Suppose $\mathcal{C} = \mathcal{GD}(9,1)$. This is the extended cyclic code over Z_4 of length 2^9 with defining sets (T_1, T_2), where

$$T_1 = \{0\} \cup cl_2(1) \cup cl_2(3) \cup cl_2(5) \cup cl_2(9) \cup cl_2(17),$$

$$T_2 = \{0\} \cup cl_2(1).$$

Choose $\beta \in \mathcal{T}_9$ such that $N(\overline{\beta}) \neq 1$, where N is the norm mapping from F_{2^9} to F_{2^3}. Then the polynomial

$$f_\beta(X) = X - \overline{\beta} X^{2^3}$$

is a permutation polynomial for F_{2^9} (see [1]). If σ_β is the corresponding permutation, note that σ_β is in $A\Gamma L(3, 2^3)$ and not in $A\Gamma L(1, 2^9)$. Observe that

$$\begin{aligned} (f_\beta^{(L)}(X))^{1 \cdot 2^9} &= (X - \beta X^{2^3})^{2^9} \\ &\equiv X + \beta X^{2^3} + 2\sqrt{\beta} X^{(1+2^3)2^8} \pmod{X^{2^9} - X} \end{aligned}$$

and that for any positive integer j,

$$\begin{aligned} (f_\beta^{(L)}(X))^{(1+2^j)2^9} &= (X - \beta X^{2^3})^{(1+2^j)2^9} \\ &\equiv X^{1+2^j} + \beta X^{2^3+2^j} + \beta^{2^j} X^{1+2^{3+j}} \\ &\quad + \beta^{1+2^j} X^{2^3(1+2^j)} \pmod{2, X^{2^9} - X} \end{aligned}$$

Since $1 + 2^j, 2^3 + 2^j, 1 + 2^{3+j}, 2^3(1 + 2^j) \pmod{2^9 - 1} \in T_1$, it follows from the previous lemma that $\sigma_\beta \in Per(\mathcal{C})$, and in fact $A\Gamma L(3, 2^3) \subseteq Per(\mathcal{C})$. It turns out that $Per(\mathcal{C}) = AGL(9, 2)$, which is larger than the group described in [3].

In conclusion, we remark that the results in this paper have been generalized for extended cyclic codes over Galois rings $GR(4, m)$ for $m > 1$, and similar results can be obtained for extended cyclic codes over Z_{p^a}, where $p > 2$.

References

[1] T. Berger and P. Charpin, "The Permutation Group of Affine-Invariant Extended Cyclic Codes", *IEEE Trans. Inform. Theory*, 42, (1996), 2194-2209.

[2] P. Charpin, "Codes cycliques étendus affines-invariants et antichaines d'un ensemble partiellement ordeoné", *Discr. Math*, 80, (1990), 229-247.

[3] A.R. Hammons Jr., P.V. Kumar, A.R. Calderbank, N.J.A. Sloane and P. Sole, "The Z_4-linearity of Kerdock, Preparata, Goethals, and related codes," *IEEE Trans. Inform. Theory*, 40, (1994), 301-319.

[4] T. Kasami, S. Lin, and W.W. Peterson, "Some results on cyclic codes which are invariant under the affine group and their applications," *Inform, Contr.*, 11, (1967), 475-496.

[5] B.S. Rajan and M.U. Siddigi, "Transform Domain Characterization of Cyclic Codes over Z_m," *Applicable Algebra in Engineering, Communication and Computing*, 5, (1994), 261-275.

Complete weight enumerators of generalized Kerdock code and linear recursive codes over Galois ring

Aleksey Kuzmin and Aleksandr Nechaev

Center of New Informational Technologies
of Moscow State University, 119899, Russia;
e-mail: nechaev@cnit.chem.msu.su

Let us remind that a *complete weight enumerator* (*c.w.e.*) of a code $\mathcal{K} \subset \Omega^n$ in alphabet Ω is a polynomial over $\mathbf{Z}$ of variables $\{x_r : r \in \Omega\}$ of the form

$$W_{\mathcal{K}}(x_r : r \in \Omega) = \sum_{\mathbf{u} \in \mathcal{K}} \prod_{r \in \Omega} x_r^{\sigma_r(\mathbf{u})},$$

where $\sigma_r(\mathbf{u})$ is the number of coordinates of the word $\mathbf{u} \in \mathcal{K}$ which are equal to $r \in \Omega$. If $\mathcal{K}$ is a distance invariant code (in particular a linear code over some abelian group $(\Omega, +)$) then its c.w.e. is a full enough characteristic of correcting properties of $\mathcal{K}$.

Let $q = 2^l$, $l \geq 1$ and $m = 2\lambda + 1 \geq 3$. A *generalized Kerdock code* $K_q(m+1)$ over the field $GF(q)$ was constructed in [1, 2]. It is a nonlinear distance invariant $\left(n, n^2, \frac{q-1}{q}(n - \sqrt{n})\right)$-code, $n = q^{m+1}$. If $q = 2$ then it is equivalent to the original binary Kerdock code [3]. The Hamming weight enumerator of this code was calculated in [4].

Theorem 1 *The c.w.e. of* $K_q(m+1)$ *is*

$$W_{K_q(m+1)}(x_0,...,x_{q-1}) = \sum_{j=0}^{q-1} x_j^n + (q^{m+2}-q)\prod_{j=0}^{q-1} x_j^{n/q}$$

$$+\frac{1}{2}q(q^m-1)(q^m+q^{\lambda+1})\prod_{j=0}^{q-1} x_j^{\frac{n}{q}-q^\lambda}\sum_{j=0}^{q-1} x_j^{q^{\lambda+1}}$$

$$+\frac{1}{2}q(q^m-1)(q^m-q^{\lambda+1})\prod_{j=0}^{q-1} x_j^{\frac{n}{q}+q^\lambda}\sum_{j=0}^{q-1} x_j^{-q^{\lambda+1}}.$$

Let $R = GR(q^2,4)$ be a Galois ring of characteristic 4 and cardinality q^2 with identity e. The generalized Kerdock code is a concatenation of some linear over the ring R code $\mathcal{K}_R(m)$, called *base code* and a linear over $GF(q)$ $[q,2,q-1]$-*Reed–Solomon* code.

The base code can be described in the following way. There exists a monic reversible polynomial $F(x) \in R[x]$ of degree m such that its period is $\tau = q^m - 1$. We call it *distinguished polynomial.* Let $G(x) = F(x)(x-e)$ and $L_R^{\overline{0,\tau-1}}(F(x))$ be the set of all segments $v(\overline{0,\tau-1}) = (v(0),...,v(\tau-1))$ of the length τ of all linear recurrences v over R with the characteristic polynomial $G(x)$. Then $\mathcal{K}_R(m)$ is the set of all words of the length $h = q^m = \tau+1$ of the form $\mathbf{v} = v(\overline{0,\tau})$: $v(\overline{0,\tau-1}) \in L_R^{\overline{0,\tau-1}}(F(x))$, $v(\tau) = v(0)+...+v(\tau-1)$. It is a linear over R $[h, m+1, \frac{q-1}{q}h]$-code.

Let $\Gamma(R) = \{\alpha \in R : \alpha^q = \alpha\}$. Then $\Gamma(R)$ is closed with respect to multiplication and consists of q elements. Any element $\beta \in R$ is the unique sum $\beta = \beta_0 + 2\beta_1$, where $\beta_t = \gamma_t(\beta) \in \Gamma(R)$, $t = 0,1$. If we define $\oplus$ on $\Gamma(R)$ by the rule $u \oplus v = \gamma_0(u+v)$, then $(\Gamma(R), \oplus, \cdot)$ is $GF(q)$. Let $\Gamma(R) = \{\omega_0 = 0,\ \omega_1,\ ...,\ \omega_{q-1}\}$ and γ_*: $R \to \Gamma(R)^q$ be a map acting on elements $r = r_0 + 2r_1 \in R$ by the rule

$$\gamma_*(r) = (r_1,\ r_1 \oplus \omega_1 r_0,\ ...,\ r_1 \oplus \omega_{q-1} r_0).$$

Then $\gamma_*(R)$ is a $[q,2,q-1]$-Reed–Solomon code over $\Gamma(R)$ and the code $\mathcal{K}_q(m+1)$ consists of all words

$$\gamma_*^h(\mathbf{v}) = (\gamma_*(v(0)),\ ...,\ \gamma_*(v(\tau))), \quad \mathbf{v} \in \mathcal{K}_R(m).$$

Theorem 2 *The c.w.e. of the base code $\mathcal{K}_R(m)$ is*

$$W_{\mathcal{K}_R(m)}(x_r : r \in R) = \sum_{r\in R} x_r^h + q(q^m-1) \sum_{r_0\in\Gamma(R)} \left(\prod_{r_1\in\Gamma(R)} x_r^{h/q} \right)$$

$$+ \frac{1}{2}(q^m-1)(q^{m-1}+q^\lambda) \prod_{r\in R} x_r^{q^{m-2}-q^{\lambda-1}} \left(\sum_{a\in R} \sum_{\omega\in\Gamma(R)} \prod_{\delta\in\Gamma(R)} x_{(\omega(a_0\oplus\delta)-a)}^{q^\lambda} \right)$$

$$+ \frac{1}{2}(q^m-1)(q^{m-1}-q^\lambda) \prod_{r\in R} x_r^{q^{m-2}+q^{\lambda-1}} \left(\sum_{a\in R} \sum_{\omega\in\Gamma(R)} \prod_{\delta\in\Gamma(R)} x_{(\omega(a_0\oplus\delta)-a)}^{-q^\lambda} \right).$$

Earlier for the code $\mathcal{K}_{\mathbf{Z}_4}(m)$ Lee w.e. [5] and c.w.e. [4] were calculated.

The investigation of weight parameters of linear recurrences with distinguished characteristic polynomials $F(x)$ is interesting not only for coding theory, but also for theory of pseudorandom sequences. In this connection we have

Theorem 3 *The c.w.e. of the code $L_R^{\overline{0,\tau-1}}(F(x))$ is*

$$W(x_r : r \in R) = x_0^\tau + (q-1) \prod_{a\in\Gamma(R)} x_{2a}^{(\tau/q)-\delta_{a,0}}$$

$$+ \sum_{\epsilon=0}^{1} \frac{1}{q}(q^m-1)(q^{m-1}+(-1)^\epsilon q^\lambda)$$

$$\times \prod_{c\in R} x_c^{q^{m-2}+(-1)^\epsilon q^{\lambda-1}-\delta_{c,0}} \left(\sum_{\omega\in\Gamma(R)} \prod_{\delta\in\Gamma(R)} x_{\omega(1+2\delta)}^{(-1)^\epsilon q^\lambda} \right).$$

In particular we have the following estimation. Let $\mathbf{u} \in L_R^{\overline{0,\tau-1}}(F(x))$ be a nondegenerate word, i.e., $\mathbf{u} \not\equiv 0 \bmod 2$. Then for any $r \in R$

$$\left| \sigma_r(\mathbf{u}) - \frac{\tau}{|R|} \right| \le \frac{q-1}{q} q^{[m/2]} < \sqrt{\tau}.$$

Note that in [6] the trigonometric sums method gives more rough estimation, with the right part equal to $q^{m/2}$.

The proofs of the presented results are based on the following reduction to quadrics over finite field of the characteristic 2. Let $S = GR(q^{2m}, 4)$ be an extension of degree m of the ring R and $Tr_R^S(x)$ be the *trace-function* from S onto R [4]. The polynomial $F(x)$ has a root $\theta \in S$ of the order τ and the code $\mathcal{K}_R(m)$ can be described as the set of all words $\mathbf{v} = (v(0) \dots v(h-1))$ such that (for some $\xi \in S, c \in R$) $v(i) = Tr_R^S(\xi\theta^i) + c, \ i = \overline{0, h-2}, \quad v(h-1) = c$. So the description of the code $\mathcal{K}_R(m)$ c.w.e. is reduced to the calculation of the number $N_\xi(c)$ of solutions of the equation $Tr_R^S(\xi x) = c$ in the set $\Gamma(S)$ and to the sorting of such numbers. This problem can be reduced to the calculation of the number of zeros of some special quadric on the spase $\Gamma(S) = GF(q^m)$ over the field $\Gamma(R) = GF(q)$ in some hiperplane of this spase (see [4]).

References

[1] Kuzmin A. S., Nechaev A. A. Linearly presented codes and Kerdock code over an arbitrary Galois field of the characteristic 2. *Russian Math. Surveys*, **49** (1994), No 5.

[2] Kurakin V. L., Kuzmin A. S., Mikhalev A. V., Nechaev A. A. Linear recurrences over rings and modules. *J. of Math. Sciences*, **76** (1995), No 6, 2793–2915.

[3] Kerdock A. M. A class of low-rate non-linear codes. *Inform. Control*, **20** (1972), 182–187.

[4] Nechaev A. A., Kuzmin A. S. Trace-function on a Galois ring in coding theory. *Lecture Notes in Computer Science*, **1255**. Springer, 1997, 277–290.

[5] Yang K., Helleseth T., Kumar P. V., Shanbhag A. G. On the weight hierarchy of Kerdock codes over $\mathbf{Z}_4$. *IEEE Trans. Inf. Theory*, **42** (1996), No 5, 1587–1593.

[6] Kumar P. V., Helleseth T., Calderbank A. R. An upper bound for Weil exponential sums over Galois ring and applications. *IEEE Trans. Inform. Theory*, **41** (1995), No 2, 456–468.

On the structure and Hamming distance of linear codes over Galois rings *

Graham Norton Ana Sălăgean–Mandache

Algebraic Coding Research Group
Centre for Communications Research
University of Bristol, U.K.

e-mail: Graham.Norton@bristol.ac.uk
Ana.Maria.Mandache@bristol.ac.uk

November 25, 1998

Abstract

We generalise structure theorems of Calderbank and Sloane for linear and cyclic codes over $\mathbb{Z}_{p^a}$ to a Galois ring $R = GR(p^a, l)$. Our results are more detailed and do not use Commutative Algebra.

We prove that $\mathrm{d}(C)$, the Hamming distance of a linear code C over R, is $\mathrm{d}(\overline{(C : p^{a-1})})$, where $(C : p^{a-1})$ is the submodule quotient and $\overline{}$ denotes projection to the residue field K of R. These two codes also have the same set of minimal codeword supports. We explicitly construct a generator matrix/ generator polynomial of $\overline{(C : p^{a-1})}$ from C. We show that in general $\mathrm{d}(C) \leq \mathrm{d}(\overline{C})$ with equality for free codes (i.e. for free R-submodules of R^n) and in particular for Hensel lifts of cyclic codes over K. Most of the codes over rings described in the literature fall into this class.

We characterise MDS codes over R and prove several analogues of properties of MDS codes over finite fields. We compute the Hamming weight enumerator of free MDS codes over R.

*Research supported by the U.K. Engineering and Physical Sciences Research Council under Grant L07680.

1 Introduction

Codes over finite rings have received much attention recently after it was proved that important families of binary non-linear codes are in fact images under the Gray map of linear codes over $\mathbb{Z}_4$, see [HKC+94] and the references cited there. For codes over $\mathbb{Z}_4$, it is usually the Lee distance which is studied, due to the fact that it coincides with the Hamming distance of the image of the code under the Gray map.

However, the Hamming distance of (linear) codes over finite rings is still important for a number of reasons. For codes over $\mathbb{Z}_{2^a}$ with $a > 2$ the Lee distance is no longer equal to the Hamming distance of the image of the code under the Gray map. (For example the elements of $\mathbb{Z}_8$ have Lee weights ranging from 0 to 4, whereas their images under any Gray map are elements of $(\mathbb{Z}_2)^3$ which can have Hamming weight at most 3.) Consequently it is not clear which metric is the most appropriate in this case. Most of the well-known algebraic decoding algorithms for codes over finite fields use the Hamming distance. Some of these algorithms can be generalised to codes over finite rings. For example analogues of Berlekamp-Massey algorithm were devised for $\mathbb{Z}_m$ in [RS85], for Galois rings in [IPE97] and more generally for any finite chain ring (i.e. a finite ring whose ideals can be linearly ordered, or equivalently a finite local ring with principal maximal ideal) in [Nor98, NS98]. There are many results on the exact value or lower bounds for the Hamming distance of codes over finite fields. Thus it is useful to have a simple mechanism to transfer all these results to codes over finite rings. Finally, let us note that the Hamming distance is obviously a lower bound for the Lee distance of the code.

2 Structure

We work with codes over Galois rings rather than over $\mathbb{Z}_{p^a}$. Our preference is motivated by the fact that Galois rings are the natural setting for Reed-Solomon and generalised Reed-Muller codes. BCH codes can also be defined over Galois rings, in analogy to BCH codes over Galois fields, see [MS77, Chaper 7].

We denote the Galois ring $GR(p^a, l)$ by R, its residue field by $K = GF(p^l)$, and projection to $K[X]$ and K^n by $\overline{}$.

We give first more detailed versions of the structure theorems for linear and for cyclic codes given in [CS95]. We generalise these results to Galois rings and give new, elementary proofs. An important role is played by the tower $\overline{C} = \overline{(C : p^0)} \subseteq \ldots \subseteq \overline{(C : p^i)} \subseteq \ldots \subseteq \overline{(C : p^{a-1})}$ of linear/cyclic codes over K associated to any linear/cyclic code C over R, where $(C : p^i) = \{e \in R^n \mid p^i e \in C\}$. We construct generator matrices/ generator polynomials for these codes, given a generator matrix/ generator polynomials of C. We also prove that for a cyclic code over R there is a unique set of generators of a form similar to the one given in [CS95] for $R = \mathbb{Z}_{p^a}$. Our proofs avoid the non-trivial Commutative Algebra invoked in [CS95], using

instead properties of the codes $\overline{(C : p^i)}$ and the technique used in [CS95, Corollary to Theorem 6] for proving that $\mathbb{Z}_{p^a}[X]/(X^n - 1)$ is principal. The fact that for any Galois ring R, $R[X]/(X^n - 1)$ is principal follows by the same technique.

There is a different set of generators which also provides a useful description of properties of a cyclic code; see [PQ96] for codes over $\mathbb{Z}_4$ and [KL97] for the generalisation to $\mathbb{Z}_{p^a}$. For a discussion of the connection between these two sets of generators, see [KL97].

3 Hamming distance

The main result of the paper consists in showing that the (Hamming) distance of a linear/cyclic code C over R is equal to the distance of $\overline{(C : p^{a-1})}$. We show that in general the distance of a linear code C over R is at most the distance of $\overline{C}$. Hence we cannot increase distance by working over finite rings rather than over finite fields. For free codes (i.e. codes which are free R-submodules of R^n) the distance of C is the same as the distance of $\overline{C}$. In particular, the (extended) Hensel lift of a cyclic code has the same distance as the original code over the finite field. Hence the classical BCH, Hartmann-Tzeng, Roos etc. bounds for cyclic codes over a finite field also hold for their Hensel lifts over Galois rings. (The BCH bound is stated in [Sha79, Theorem 4] with an incorrect proof.)

We examine a number of codes over Galois rings described in the literature and apply our results to either determine or give lower bounds for their distance.

4 MDS codes

Finally, we examine MDS codes over a Galois ring. We characterise these codes: a code C over R is MDS iff $\overline{(C : p^{a-1})}$ is an MDS code over K. For free codes, this means that C is MDS iff $\overline{C}$ is MDS. We prove a number of properties of MDS codes over R analogous to properties of MDS codes over finite fields. We determine the weight enumerator of a free MDS code.

References

[CS95] A. R. Calderbank and N. J. A. Sloane. Modular and p-adic codes. *Designs, Codes and Cryptography*, 6:21–35, 1995.

[HKC+94] A. R. Hammons, Jr., P. V. Kumar, A. R. Calderbank, N. J. A. Sloane, and P. Solé. The $\mathbb{Z}_4$ linearity of Kerdock, Preparata, Goethals and related codes. *IEEE Trans. Inform. Theory*, 40:301–319, 1994.

[IPE97] J. C. Interlando, R. Palazzo, and M. Elia. On the decoding of Reed-Solomon and BCH codes over integer residue rings. *IEEE Trans. Inform. Theory*, 43(3):1013–1021, 1997.

[KL97] P. Kanwar and S. R. López-Permouth. Cyclic codes over the integers modulo $\mathbb{Z}_{p^m}$. *Finite Fields and Their Applications*, 3:334–352, 1997.

[MS77] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error-correcting Codes*. North Holland, Amsterdam, 1977.

[Nor98] G. H. Norton. On minimal realization over a finite chain ring. *Designs, Codes and Cryptography*, 1998. Accepted for publication.

[NS98] G. H. Norton and A. Sălăgean-Mandache. On the key equation over a commutative ring. *Designs, Codes and Cryptography*, 1998. Accepted for publication.

[PQ96] V. S. Pless and Z. Qian. Cyclic codes and quadratic residue codes over $\mathbb{Z}_4$. *IEEE Trans. Inform. Theory.*, 42(5):1594–1600, 1996.

[RS85] J. A. Reeds and N. J. A. Sloane. Shift-register synthesis (modulo m). *SIAM J. Computing*, 14:505–513, 1985.

[Sha79] P. Shankar. On BCH codes over arbitrary integer rings. *IEEE Trans. Inform. Theory*, 25(4):480–483, 1979.

Cyclic and affine invariant codes

Kanat S. Abdukhalikov
Institute for Pure and Applied Mathematics
Pushkin Str 125, Almaty 480100, Kazakhstan
E-mail: abdukhalikov@itpm.sci.kz

Abstract

Affine invariant and cyclic codes over p-adic numbers and over integers modulo p^d are studied. It has been determined what extended cyclic code has an extension which is affine invariant.

Keywords: cyclic codes, affine invariant codes, codes over rings.

1 Introduction

Let p be a prime number, $q = p^m$ and V be the additive subgroup of a finite field $\mathbf{F}_q$ of p^m elements. Let F be a ring. We consider the group ring $F[V]$ as the set of all formal linear combinations $a = \sum a_v X^v$ with $v \in V$ and $a_v \in F$. Addition and scalar multiplication are component-wise and the multiplication is given by the addition in V:

$$\sum a_v X^v + \sum b_v X^v = \sum (a_v + b_v) X^v,$$

$$\left(\sum a_v X^v\right) \cdot \left(\sum b_v X^v\right) = \sum_{u,v} a_u b_v X^{u+v} = \sum_w \left(\sum_u a_u b_{w-u}\right) X^w.$$

The element X^0 is the unity of the ring $F[V]$ and we write $X^0 = 1$. So $F[V]$ is a free module over F of rank p^m and with basis $\{X^v \mid v \in V\}$. We can consider submodules of $F[V]$ as codes of length p^m based on the alphabet F.

Note that the elements of $F[V]$ can be considered as functions from V to F. As a function, an element $\sum a_v X^v$ is the one that assigns a_v to the element v of V.

Now we consider V as an one-dimensional vector space over the field $\mathbf{F}_q$ of p^m elements. Affine group $G = AGL(V) = V \cdot GL(V)$ is a semidirect product of the abelian group V and the multiplicative group $T = GL(V) = \mathbf{F}_q^*$ of the field $\mathbf{F}_q$. We make $F[V]$ into a G-module by putting

$$\sigma_u(X^v) = X^{u+v}, \quad u \in V;$$

$$\rho_t(X^v) = X^{tv}, \quad t \in T.$$

A code $C \subset F[V]$ is said to be affine invariant if it is invariant under the group G. In our case affine invariant codes closely related to cyclic codes. Cyclic code C of length $q-1$ over F is an ideal in the quotient ring $F[Y]/(Y^{q-1}-1)$. Let ω be a primitive element of the field $\mathbf{F}_q$. Then the correspondence

$$\sum_{i=0}^{q-2} c_i Y^i \mapsto \sum_{i=0}^{q-2} c_i X^{\omega^i} \tag{1}$$

gives a bijective map from $F[Y]/(Y^{q-1}-1)$ to the module $\sum_{v \neq 0} FX^v$. Thus description of cyclic codes of length $q-1$ is equivalent to the description of subcodes in $\sum_{v \neq 0} FX^v$ that are invariant under the group $T = GL(V) \cong \mathbf{F}_q^*$. The extended cyclic code is obtained by embedding:

$$\sum_{i=0}^{q-2} c_i Y^i \mapsto (-\sum_{i=0}^{q-2} c_i) X^0 + \sum_{i=0}^{q-2} c_i X^{\omega^i}.$$

We are going to study affine invariant codes in the cases $F = \mathbf{Z}_p$ (the ring of p-adic integers) and $F = \mathbf{Z}/p^d\mathbf{Z}$ (the ring of integers modulo p^d). Consider modules

$$W = \{\sum_{v \in V} a_v X^v \mid a_v \in \mathbf{Z}_p\},$$

$$W' = \{\sum_{v \neq 0} a_v X^v \mid a_v \in \mathbf{Z}_p\}.$$

The module W is the ambient space for our affine invariant codes over $\mathbf{Z}_p$, and we will understand as a cyclic code of length $q-1$ over $\mathbf{Z}_p$ a code in W' invariant under the group T. Similarly for codes over $\mathbf{Z}/p^d\mathbf{Z}$ we set

$$W_d = \{\sum_{v \in V} a_v X^v \mid a_v \in \mathbf{Z}/p^d\mathbf{Z}\},$$

$$W'_d = \{\sum_{v\neq 0} a_v X^v \mid a_v \in \mathbf{Z}/p^d\mathbf{Z}\}.$$

We have $W_d \cong W/p^dW$, since $\mathbf{Z}_p/p^d\mathbf{Z}_p \cong \mathbf{Z}/p^d\mathbf{Z}$. Duality is defined with respect to the standard inner product

$$\left(\sum a_v X^v, \sum b_v X^v\right) = \sum a_v b_v.$$

If C is a cyclic code in W' (resp. W'_d) then the extended cyclic code $\hat{C}$ is obtained by embedding:

$$\sum_{v\neq 0} a_v X^v \mapsto (-\sum_{v\neq 0} a_v)X^0 + \sum_{v\neq 0} a_v X^v.$$

Set

$$\mathcal{K} = \{\overline{k} = (k_0, \ldots, k_{m-1}) \mid 0 \leq k_i \leq p-1, i = 0, \ldots, m-1\},$$

$$\mathcal{K}' = \mathcal{K} \setminus \{(0, \ldots, 0)\}.$$

For element $\overline{k} \in \mathcal{K}$ we will assume $k = \sum_{i=0}^{m-1} k_i p^i$. It is clear that the map $k \mapsto \overline{k}$ gives bijective correspondence between the set $\{0, 1, \ldots, q-1\}$ of integer numbers and the set $\mathcal{K}$. So $\overline{k}$ corresponds to p-adic expansion of k. We define an equivalence relation $\sim$ on $\mathcal{K}$. We will say that

$(k_0, \ldots, k_{m-1}) \sim (n_0, \ldots, n_{m-1})$, if $k_0 = n_s$, $k_1 = n_{1+s}$, $k_2 = n_{2+s}$, ... for some integer s (that is, $\overline{k}$ and $\overline{n}$ coincide for some cyclic shift). In other words, equivalence $\overline{k} \sim \overline{n}$ for $\overline{k}$, $\overline{n} \in \mathcal{K}'$ means that numbers $k = \sum_{i=0}^{m-1} k_i p^i$ and $n = \sum_{i=0}^{m-1} n_i p^i$ belong to one cyclotomic coset modulo $q-1$. Let $K = \mathcal{K}/\sim$, $K' = K \setminus \{(0, \ldots, 0)\}$, so m-tuples in K are identified under cyclic shift: $(k_0, \ldots, k_{m-1}) = (k_{m-1}, k_0, \ldots, k_{m-2})$.

In the case $F = \mathbf{F}_p$ the ambient space $W'_1 = \sum_{\overline{k}\in K'} P_1(\overline{k})$ is a direct sum of minimal cyclic codes $P_1(\overline{k})$. If ω is a primitive generator of the field $\mathbf{F}_q$, then the cyclic code $P_1(\overline{k})$ specified by the generator polynomial $(x^{q-1}-1)/h_k(x)$, where $h_k(x)$ is the minimal polynomial of ω^k. Furthermore, any cyclic code C can be described as $C = \sum_{\overline{k}\in M} P_1(\overline{k})$ for some $M \subset K'$, and the generator polynomial of C is $(x^{q-1}-1)/\prod_{\overline{k}\in M} h_k(x)$.

We prove that $W' = \oplus P(\overline{k})$ is a direct sum of cyclic codes $P(\overline{k})$ over $\mathbf{Z}_p$, where the code $P(\overline{k})$ over $\mathbf{Z}_p$ is obtained from the cyclic code $P_1(\overline{k})$ over $\mathbf{F}_p$ as the limit of sequence of Hensel lifts.

Theorem 1 *Let C be a cyclic code over $\mathbf{Z}_p$. Then there exist a subset $M \subset K'$ and rational nonnegative integers $l(\overline{k})$, $\overline{k} \in M$, such that $C = \sum_{\overline{k} \in M} p^{l(\overline{k})} P(\overline{k})$. The extended cyclic code $\hat{C}$ is affine invariant if and only if a condition $p^{l(\overline{k})} P(\overline{k}) \subset C$ follows $p^{l(\overline{k})} P(\ldots, k_{i-1}, k_i+1, k_{i+1} \ldots) \subset C$ for any i, where it is assumed $P(\ldots, k_{i-1}, p, k_{i+1}, \ldots) = pP(\ldots, k_{i-1}, 0, k_{i+1}+1, \ldots)$ (indices are considered modulo m).*

Cyclic codes over $\mathbf{Z}/p^d\mathbf{Z}$ are obtained from these universal codes (i.e. codes over $\mathbf{Z}_p$) by reduction modulo p^d.

Example. Let us consider the case $d = 1$ ($F = \mathbf{F}_p$). The previous theorem says that a cyclic code $C = \sum_{\overline{k} \in M} P_1(k)$ has an extension which is affine invariant if and only if a condition $(\ldots, k_{i-1}, k_i, k_{i+1} \ldots) \in M$, $k_i < p-1$ implies $(\ldots, k_{i-1}, k_i + 1, k_{i+1} \ldots) \in M$ for all i. Recall that the defining set $J(C)$ of a cyclic code C with the generator polynomial $h(x)$ is $J(C) = \{j \mid 1 \leq j \leq q-1,\ h(\omega^j) = 0\}$ (note we consider here elements $j \in J(C)$ in the range $1 \leq j \leq q-1$ rather than $0 \leq j \leq q-2$). It is clear that $J(C) = \{k \mid \overline{k} \in K' \setminus M\}$. Therefore, we have the well-known result [3, 7, 12]: the extended code $\overline{C}$ is affine invariant if and only if its defining set $J(C)$ has the property that $j \in J(C)$ implies $s \in J(C)$ for all $s \preceq j$.

Now we consider a particular case of cyclic and extended cyclic codes over $\mathbf{Z}/4\mathbf{Z}$ (i. e. $p^d = 4$). Following [10, 11] we can describe codes in terms of the set $\mathcal{T} = \{0, 1, \xi, \xi^2, \ldots, \xi^{2^m-2}\}$, where ξ is a primitive (2^m-1)th root of unity in the Galois ring $GR(4^m)$. For any element $\alpha \in \mathbf{F}_q$ we will denote by $\tilde{\alpha}$ the corresponding element in $\mathcal{T}$.

Theorem 2 *Elements $\sum_{\alpha \neq 0} a_\alpha X^\alpha$ of a cyclic code C in W_2' can be determined by the following system of equations:*

$$\sum_{\alpha \neq 0} a_\alpha \tilde{\alpha}^k = 0, \quad \overline{k} \in B_1;$$

$$2 \sum_{\alpha \neq 0} a_\alpha \tilde{\alpha}^k = 0, \quad \overline{k} \in B_2,$$

where $B_1 \cup B_2 \subset K'$, $B_1 \cap B_2 = \emptyset$.

Note that these systems of equations are well-defined by condition $\overline{k} \in B_i$, since $\sum_{\alpha \neq 0} a_\alpha \tilde{\alpha}^k = 0$ if and only if $\sum_{\alpha \neq 0} a_\alpha \tilde{\alpha}^{kp} = 0$.

Theorem 3 *Let C be a cyclic code in W_2' and B_1, B_2 be as in theorem 2. Then the extended cyclic code $\hat{C}$ is affine invariant if and only if the following conditions hold:*

(i) $(k_0, \ldots, k_{i-1}, 1, k_{i+1}, \ldots) \in B_1$ *implies* $(k_0, \ldots, k_{i-1}, 0, k_{i+1}, \ldots) \in B_1 \cup \{(0, \ldots, 0)\}$;

(ii) $(k_0, \ldots, k_{i-1}, 1, k_{i+1}, \ldots) \in B_2$ *implies* $(k_0, \ldots, k_{i-1}, 0, k_{i+1}, \ldots) \in B_1 \cup B_2 \cup \{(0, \ldots, 0)\}$;

(iii) $(k_0, \ldots, k_{i-1}, 0, 1, k_{i+2}, \ldots) \in B_1$ *implies* $(k_0, \ldots, k_{i-1}, 1, 0, k_{i+2}, \ldots) \in B_1 \cup B_2$.

There is one interesting result that has no analogies in the general case.

Corollary 4 *Let $K^n = \{(k_0, \ldots, k_{m-1}) \in K \mid \sum k_i < n\}$, $1 \leq n \leq m$, and let $\hat{C}^n$ be an extended cyclic code defined by equations*

$$\sum_{\alpha} a_\alpha \tilde{\alpha}^k = 0, \quad \overline{k} \in K^n.$$

Then the code $\hat{C}^n$ is affine invariant.

Modulo 2 the codes $\hat{C}^n$ are Reed-Muller codes $\mathcal{R}(m-n, m)$ [3, 4]. They are affine invariant. In terms of papers [5, 13] the previous corollary means that the Hensel liftings of binary Reed-Muller codes to codes over

$\mathbf{Z}/4\mathbf{Z}$ are still affine invariant (but this is not true with respect to codes over $\mathbf{Z}/8\mathbf{Z}$). These codes are called quaternary Reed-Muller codes in [10]. It can be proved that $(\hat{C}^n)^\perp = \hat{C}^{m+1-n}$.

The binary images of the codes $\hat{C}^2$ and $\hat{C}^{m-1}$ under the

Gray map are the "Preparata" and Kerdock codes [10].

In [12] (see also [3] and [7]) the affine invariance property of extended cyclic codes was studied for $F = \mathbf{F}_p$ in terms of defining sets, and from theorem 1 one can get an analog of this result for any d (see [1]).

We can also give a classification of affine invariant codes of length p^m, as well as a classification of codes invariant under the full affine group $AGL_m(\mathbf{F}_p)$.

Some classes of extended cyclic codes over $\mathbf{Z}/4\mathbf{Z}$ with affine invariant property were considered in [6], and theorem 2.1 from [6] is a particular case of our theorem 3. Cyclic codes over $\mathbf{Z}/4\mathbf{Z}$ were also studied in [11, 13, 14]. In [7, 8, 9] it has been studied affine invariant codes and group algebra codes over a finite field.

The approach of this paper was inspired by the work [2], where invariant integral lattices were considered.

References

[1] K. S. Abdukhalikov, Affine invariant and cyclic codes over p-adic numbers and finite rings (submitted).

[2] K. S. Abdukhalikov, Invariant integral lattices in Lie algebras of type A_{p^m-1}, *Mat. Sb.*, Vol. 184, No. 4 (1993) pp. 61–104; English transl. in *Russian Acad. Sci. Sb. Mat.*, Vol. 78, No. 4 (1994) pp. 447–478.

[3] E. F. Assmus and J.D.Key, Polynomial codes and finite geometries, to appear in Handbook of Coding Theory, edited by V. Pless, W. C. Huffman and R. Brualdi.

[4] S. D. Berman, On the theory of group codes, *Kibernetika*, Vol. 3, No. 1 (1967) pp. 31–39.

[5] A. R. Calderbank and N. J. A. Sloane, Modular and p-adic cyclic codes, *Designs, Codes and Cryptography*, Vol. 6 (1995) pp. 21–35.

[6] A. R. Calderbank and G. Mcguire, Construction of a $(64, 2^{37}, 12)$ code via Galois Rings, *Designs, Codes and Cryptography* Vol. 10 (1997) pp. 157–165.

[7] P. Charpin, *Codes cycliques étendus invariants sous le groupe affine.* Thèse de Doctorat d'État, Université Paris VII, 1987.

[8] P. Charpin, Une generalization de la construction de Berman des codes de Reed et Muller p-aries, *Comm. Algebra* Vol. 16, No. 11 (1988) pp. 2231–2246.

[9] P. Charpin and F. Levy-Dit-Vehel, On Self-dual Affine-Invariant Codes, *J. Combin. Theory Ser. A* Vol. 67 (1994) pp. 223–244.

[10] R. Hammons, P. V. Kumar, A. R. Calderbank, N. J. A. Sloane and P. Solé, The Z_4-linearity of Kerdok, Preparata, Goethals, and related codes, *IEEE Trans. Inform. Theory* Vol. 40, No. 2 (1994) pp. 301–319.

[11] T. Helleseth, P. V. Kumar and A. Shanbhag, Codes with the Same Weight Distributions as the Goethals Codes and the Delsarte-Goethals Codes, *Designs, Codes and Cryptography* Vol. 9 (1997) pp. 257–266.

[12] T. Kasami, S. Lin and W. W. Peterson, Some results on cyclic codes which are invariant under the affine group and their applications, *Inform. and Control*, Vol. 11 (1967) pp. 475–496.

[13] V. Pless, P. Solé and Z. Qian, Cyclic Self-Dual Z_4-Codes, *Finite Fields and Their Appl.*, Vol. 3, No. 1 (1997) pp. 48–69.

[14] V. Pless and Z. Qian, Cyclic Codes and Quadratic Residue Codes over Z_4, *IEEE Trans. Inform. Theory* Vol. 42, No. 5 (1996) pp. 1594–1600.

Lattices, Codes, and Radon Transforms

M. Boguslavsky*

December 9, 1998

Abstract

We present a new approach based on integral geometry methods which allows us to obtain some relations on generalized spectra of codes and lattices and gives a unifying point of view on some known results on linear codes. Actually, we construct analogues of the Radon transform in certain spaces connected with codes and lattices and use then the Plancherel formula.

ici Lattice = reseau.

1 Introduction

Generalized Hamming weights (gHw) of linear codes (also known as *minuimum support sizes* or *dimension/length profile*) were first introduced by Helleseth, Kløve and Mykkeltveit [HKM] in 1976. They describe the performance of a linear code when used in certain wire-tap channels [Wei1] and are related to trellis complexities of codes [For1]. There exists a noticeable amount of papers concerning generalized weights, including the surveys [TV2], [Wei2].

The theory of lattice sphere packings in Eucledean spaces is in many instances similar to the theory of linear block codes. Generalized weights admit lattice analogues, which are called *generalized Hermitian parameters* (gHp) or *density/length profile*. Although gHp were introduced [Ran] more than 20 years earlier than gHw, they are much less studied. Forney [For2] showed that they are related to trellis complexities in the same way as gHw do. We give the definition and a brief overview of properties of gHp in section 2.

In section 3, we mention some definitions and facts from the theory of homogenous spaces in duality as developed in [Hel1] and [Hel2]. This theory allows to extend the classical notion of the *Radon transform* in $\mathbb{R}^n$ to a rather general situation. The Plancherel formula (11) becomes then an ample source of identities for functional sums.

*The author is with Kortweg–de Vries Institute for Mathematics, University of Amsterdam and with Institute for Information Transmission Problems, Russian Academy of Sciences, Moscow; e-mail: michaelb@wins.uva.nl

$\gamma_{n,q} = \max \gamma_r(L)$.

- analogues of generalized Hamming weights
- related to Weil's conjecture and decoding.
- appears naturally:
  - Riemann geom (systole bds).
  - adelic geometry
  - lattices from global fields

The generalized spectra of a linear block code is the distribution of subcode support sizes; the generalized spectra of a lattice in $\mathbb{R}^n$ is the distribution of sublattice volumes. It is often convenient to store the information about the spectra as the coefficients of a power series. We introduce the notion of *the r-th T-function of a lattice*, which is a kind of generalization of the classical Θ-function of a lattice. There is an obvious duality between the T-functions of a lattice and its dual.

We use the Plancherel formula for a Radon transform in a suitable space to obtain amazing identities on T-functions. One may prove a similar identity for generalized weight distributions of a linear code. We also use the Plancherel formula to reprove two known results from coding theory, namely, an upper bound on r-th generalized weight and Nogin inversion formula for projective multiset multiplicities.

In all these cases the proofs can be easily reformulated without any use of Radon transforms. However, it seems to be useful to have a unifying point of view at all these problems, so we use the theory of homogenous spaces in duality rather as a convenient language than as a tool.

2 Generalized Hermitian parameters

2.1 Definitions

Let L be a full rank lattice in $\mathbb{R}^n$. The determinant of the Gram matrix of any base of L does not depend on the choice of a base; it is called the *determinant* of L and is denoted by $\det L$. By $\mathrm{vol}(L)$ we denote the volume of the fundamental domain of L; clearly, $\det L = \mathrm{vol}^2(L)$. We denote the length of a minimal vector of L by $r(L)$. The number $r^2(L)$ is also called the *minimum norm* of L. The *Hermitian parameter* $\gamma(L)$ of L is defined by

$$\gamma(L) := r^2(L)/\det^{1/n} L.$$

It does not depend on the scaling and is also called the *coding gain* of L. The maximum of $\gamma(L)$ over all lattices of rank n is denoted by γ_n and is called the *(true) Hermitian constant.*

Let $\mathrm{vol}_m(L)$ denote the minimum volume of an m-sublattice ($m = 1, \ldots, n$) of L:

$$\mathrm{vol}_m(L) := \min_{M \subset L,\ \mathrm{rk} M = m} \mathrm{vol}(M).$$

It is clear that $\mathrm{vol}_1(L) = r(L)$ and $\mathrm{vol}_n(L) = \mathrm{vol}(L)$. One way to normalize these volumes is to consider *generalized Hermitian parameters* $\gamma_m(L)$;

$$\gamma_m(L) := \mathrm{vol}_m^2(L)/\det^{m/n} L.$$

Rankin [Ran] proved that the *(true) generalized Hermitian constants*

$$\gamma_{n,m} = \max_{\mathrm{rk}(L)=n} \gamma_m(L)$$

are well-defined.

2.2 Properties

A) Generalized Mordell inequality [Ran]. If $1 \le m < r \le n-1$ then we have

$$\gamma_{n,m} \le \gamma_{r,m}(\gamma_{n,r})^{m/r}, \tag{1}$$

and for any n-lattice L and $1 \le m < r \le n-1$ we have

$$\gamma_m(L) \le \gamma_{r,m}(\gamma_r(L))^{m/r}. \tag{2}$$

Rankin proved (1); his argument also proves Eq. (2) although he did not state it explicitly.

Substituting $m = 1$ into Eq. (2) we get the inequality

$$\gamma_r(L) \ge \gamma_1^r(L)/\gamma_r^r, \tag{3}$$

which is equivalent to a bound from [For2]. A lattice L meets bound (3) iff it has the densest r-dimensional lattice as a sublattice with the same minimum norm as L. For example, $\gamma_r(\Lambda_n) = \gamma(\Lambda_n)/\gamma_r^r$ for $r = 1, \ldots, 8$ and any n.

B) Coulangeon [Cou] proved that we have the following upper bound on $\gamma_r(L)$

$$\gamma_r(L) \le \gamma_n^r. \tag{4}$$

C) Bounds via antipodal spherical codes [Bog1]. These are a family of bounds valid only for lattices with several minimal vectors. We state here only several simplest bounds of this type on $\gamma_2(L)$. Let τ be the kissing number of a lattice L, i.e. the number of vectors in L with the minimum norm. It is clear that τ is at least two. For a "random" lattice τ is equal to two, but most known "nice" lattices have a large kissing number. We have

$$\begin{aligned}
\frac{\gamma_2(L)}{\gamma_1(L)^2} &\le \frac{n-1}{n} \times \frac{\tau}{\tau-2}, \quad \text{whenether } \tau > 2; \\
\frac{\gamma_2(L)}{\gamma_1(L)^2} &\le \frac{n-1}{n+2} \times \frac{\tau}{\tau-2n}, \quad \text{whenether } \tau > 2n; \\
\frac{\gamma_2(L)}{\gamma_1(L)^2} &\le \frac{n-1}{n}, \quad \text{whenether } \tau > n(n+1).
\end{aligned}$$

D) Duality [Ran]. Scale a lattice L so that $\det L = 1$. Then to any r-sublattice M of a rank-n lattice L one may associate an $(n-r)$ sublattice $\tilde{M}$ of $L^\perp$ such that

$$\det M = \frac{1}{\det \tilde{M}} \tag{5}$$

This implies, in particular, that for any $1 \le m < n$ we have

$$\gamma_{n,m} = \gamma_{n,n-m}.$$

2.3 T-functions

Let $L \subset \mathbb{R}^n$ be a lattice and let $M \subset L$ be a sublattice of L. By $\langle M \rangle_{\mathbb{R}}$ we denote the space of all *real* linear combinations of vectors from M. We shall say that $M \subset L$ is *primitive* iff $\langle M \rangle_{\mathbb{R}} \cap L = M$. This is equivalent to the property that M is not contained in any sublattice of L of the same dimension as M. Let $L_0^{[r]}$ denote the set of all primitive r-sublattices of L and let $L^{[r]}$ denote the set of all shifts of primitive r-sublattices of L by vectors from L.

It is convenient to describe the spectrum of a lattice (the distribution of lengths of lattice vectors) by a Θ-*function* of the lattice

$$\Theta_L(q) := \sum_{v \in L} q^{\|v\|} \tag{6}$$

(sometimes it is convenient to substitute $q = e^{\pi i z}$.)

The code analogue of the Θ-function is the MacWilliams weigth enumerator

$$W_C(x,y) := \sum_{c \in C} x^{n-\mathrm{wt}(c)} y^{\mathrm{wt}(c)},$$

where C is a linear $[n,k]$-code and $\mathrm{wt}(c)$ denotes the Hamming weight of a codeword c. It is convenient to enumerate the determinants of primitive subalttices by the set of lattice T-*functions* $\{T_L^r(q), r = 1, \ldots, n\}$:

$$T_L^r(q) := \sum_{\xi \in L_0^{[r]}} q^{\det \xi}. \tag{7}$$

Eq. (5) implies that if $\det L = 1$ then

$$T_L^r(q) = T_{L^\perp}^{n-r}(q). \tag{8}$$

Similarly, one may define generalized MacWilliams weight enumerators $W_C^r(x,y)$ of a linear code C. Analogues of the MacWilliams identity for generalized spectra were found in [Klo].

The first T-function $T_L^1(q)$ does not coincide with the Θ-function of L; the summation in (7) is over all primitive vectors of L, while in (6) the summation is over all vectors of L. However, T_L^1 and Θ_L are connected by a kind of Moebius transform and Θ_L may be expressed via a summation over all primitive 1-sublattices of the third Jacobi θ-function θ_3 (see [Bog2]).

3 Homogenous spaces in duality

In this section we give the definition and mention some properties of homogenous spaces in duality. For more details, see [Hel1] and [Hel2].

Let G be a locally compact group, X and Ξ two (left) coset spaces $X = G/H_X$ and $\Xi = G/H_\Xi$, where H_X and H_Ξ are two closed subgroups of G. Let us make the following assumptions: (i) The groups G, H_X, H_Ξ, $H_X \cap H_\Xi$ are unimodular (i.e. the left-invariant Haar measures are right-invariant); (ii) For any $h_X \in H_X$ the inclusion $h_X H_\Xi \subset H_\Xi H_X$ implies $h_X \in H_\Xi$; for any $h_\Xi \in H_\Xi$ the inclusion $h_\Xi H_X \subset H_X H_\Xi$ implies $h_\Xi \in H_X$; (iii) The set $H_X H_\Xi$ is closed.

Homogenous spaces X and Ξ are called *homogenous spaces in duality.*

In this paper we shall deal only with discrete groups so (i) and (iii) will be satisfied automatically.

We shall say that $x \in X$ and $\xi \in \Xi$ are *incident* and denote it by $x \bowtie \xi$ if the cosets xH_X and ξH_Ξ are not disjoint.

We put $\check{x} = \{\xi \in \Xi : x \bowtie \xi\} \subset \Xi$ and $\hat{\xi} = \{x \in X : \xi \bowtie x\} \subset X$. The factor G/K may be identified with the set $\{(x, \xi) \in X \times \Xi : x \bowtie \xi\}$. Given Haar measures that satisfy (i) we may construct nice G-invariant measures $m(x)$ on each $\hat{\xi}$ and $\mu(\xi)$ on each $\check{x}$ (cf. [Hel1, p. 143].)

The *Radon transform* $\hat{f} : \Xi \to \mathbb{C}$ of a function $f : X \to \mathbb{C}$ is defined by

$$\hat{f}(\xi) = \int_{\hat{\xi}} f(x)\, dm(x); \tag{9}$$

the *dual Radon transform* $\check{\phi} : X \to \mathbb{C}$ of a function $\phi : \Xi \to \mathbb{C}$ is defined by

$$\check{\phi}(x) = \int_{\check{x}} \phi(\xi)\, d\mu(\xi). \tag{10}$$

Lemma 1 [Hel1], Plancherel formula. *Let $f : X \to \mathbb{C}$ and $\phi : \Xi \to \mathbb{C}$ be continous compact support functions. Then $\hat{f}$ and $\check{\xi}$ are continous and*

$$\int_X f(x)\check{\phi}(x)\, dx = \int_\Xi \hat{f}(\xi)\phi(\xi)\, d\xi. \tag{11}$$

Note. For a discrete group G the formal equalities

$$\sum_{x \in X} f(x)\check{\phi}(x) = \sum_{(x,\xi) \in X \times \Xi:\ x \bowtie \xi} f(x)\phi(\xi) = \sum_{\xi \in \Xi} \hat{f}(\xi)\phi(\xi) \tag{12}$$

show that Eq. (11) holds also for any functions f and ϕ such that all series in (12) converge absolutely. The proof for the general case is similar but requires some additional facts about measures and groups.

Actually, equality (11) holds in a much more general situation than that of conditions (i)-(iii).

4 Sublattices as a homogenous space

4.1 A duality between $(n-1)$-sublattices and vectors

Let $A(n)$ denote the group of integer $n \times n$ matrices with the determinant ± 1:

$$A(n) := \{M \in GL_n(\mathbb{Z}) : \quad |\det M| = 1\} \simeq SL_n(\mathbb{Z}) \times \mathbb{Z}_2.$$

The subset $G = R(n) \subset GL_{n+1}(\mathbb{Z})$ defined by

$$G = R(n) := \begin{pmatrix} A(n) & \mathbb{Z}^n \\ 0 & 1 \end{pmatrix} \tag{13}$$

is a group with the respect to the usual matrix multiplication. The map $x \mapsto Mx = M'x + v$, $M = \begin{pmatrix} M' & v \\ 0 & 1 \end{pmatrix} \in R(n), x \in \mathbb{Z}^n$ defines an action of $R(n)$ on $\mathbb{Z}^n$. Note that this is also a transitive action of $R(n)$ on the set of all shifts of bases of $\mathbb{Z}^n$.

Let H_X denote the stabilizer of the point 0:

$$H_X = St(0) \simeq A(n); \tag{14}$$

let Π be a shift of an $(n-1)$-sublattice of $\mathbb{Z}^n$ and let H_Ξ denote the stabilizer of Π. Assume now that Π is the sublattice $\Pi_0 \subset \mathbb{Z}^n$ spanned by the first $n-1$ base vectors; then

$$H_\Xi = St(\Pi_0) = St\langle v_1, v_2, \ldots, v_{n-1}\rangle \simeq R(n-1) \times \mathbb{Z}_2. \tag{15}$$

Lemma 2 *The spaces $X = G/H_X$ and $\Xi = G/H_\Xi$ defined by Eq. (13), (14) and (15) satisfy conditions (i), (ii) and (iii).*

We omit the proof.

The coset space $X = G/H_X$ may be identified with $\mathbb{Z}^n$ and the coset space $\Xi = G/H_\Xi$ may be identified with the set $(\mathbb{Z}^n)^{[n-1]}$ of all shifts of primitive $(n-1)$-sublattices of $\mathbb{Z}^n$. It can be checked that with our choice of Π_0 a point $x \in X$ is incident with a shift of sublattice $\xi \in \Xi$ iff $x \in \xi$.

Note. A different choice of Π_0 in (15) will give a different incidence relation; for example, when H_Ξ stabilizes

$$\Pi = \Pi_\lambda = \langle v_1, v_2, \ldots, v_{n-1}\rangle + \lambda v_n,\ \lambda \in \mathbb{Z},$$

we get the incidence relation $x \overset{\lambda}{\bowtie} \xi \Leftrightarrow \mathrm{ind}_{\mathbb{Z}^n}(x, \xi) = \lambda$.

The Plancherel formula (11) gives

$$\sum_{x \in L} f(x)\check{\phi}(x) = \sum_{\xi \in L^{n-1}} \hat{f}(\xi)\phi(\xi) \tag{16}$$

for any $f : L \to \mathbb{C}$ and $\phi : L^{[n-1]} \to \mathbb{C}$ such that both series converge absolutely. As a corollary of the Plancherel formula one may prove the following theorem.

Theorem 1 *For any lattice L of rank n*

$$\Theta_L(q) \cdot T_L^{n-1}(p) = \sum_{\xi \in L^{[n-1]}} p^{\|\xi\|}\Theta_\xi(q). \tag{17}$$

Thus the product of the Θ-function of a lattice with the $(n-1)$-th T-function of the same lattice equals the weighted sum of Θ-functions of all shifts of $(n-1)$-sublattices. Using the duality (8) we get

Corollary 1 *For any lattice L of rank n*

$$\Theta_L(q) \cdot T^1_{L^\perp}(p) = p^{\det L^\perp} \sum_{\xi \in L^{[n-1]}} p^{\|\xi\|} \Theta_\xi(q).$$

4.2 Sublattices of other ranks

There exists a duality between the sets X_a and X_b of sublattices of dimensions a and b respectively, $a + b = n - 1$, which is similar to the duality between $(n-1)$-sublattices and vectors. The previous subsection may be regarded as the case $a = 0$. Let G be the same group as in Eq. (13). Now, instead of subgroups H_X and H_Ξ defined by Eq. (14) and (15) let us consider subgroups $H_a \subset G$ and $H_b \subset G$, $a + b = n - 1$ defined by

$$H_a = St\langle v_1, v_2, \ldots, v_a \rangle \tag{18}$$

$$H_b = St\langle v_{n-b+1}, v_{n-b+2}, \ldots, v_n \rangle, \tag{19}$$

where $v_1, \ldots, v_n$ is the standard base of $\mathbb{Z}^n$. The corresponding coset spaces $X_a = G/H_a$ and $X_b = G/H_b$ may be identified with the sets of shifts of primitive sublattices in $\mathbb{Z}^n$ of dimensions a and b. It is not hard to check that conditions (i)–(iii) hold. For these spaces one may write down an analogue of Theorem 1 giving a relation between T^a_L and T^b_L.

5 Homogenous spaces and codes

In this section, we study different forms of the Radon transform in the n-dimensional projective space over a finite field $\mathbb{F}_q$. We shall reprove a known upper bound on r-th generalized Hamming weight [TV2], obtain a code analogue of Eq. (17) and an interpretation of Nogin [Nog] results on projective systems.

5.1 Radon transform in a linear space over a finite field

To obtain a code analogue of Eq. 17 we shall consider the following homogenous spaces in duality. Let the subgroup $G = R_q(n) \subset GL(n+1, \mathbb{F}_q)$ be defined by

$$G = R_q(n) := \begin{pmatrix} GL(n, \mathbb{F}_q) & \mathbb{F}_q^n \\ 0 & 1 \end{pmatrix}. \tag{20}$$

Define the action of G on $\mathbb{F}_q^n$ by the map $x \mapsto Mx = M'x + v$, $M = \begin{pmatrix} M' & v \\ 0 & 1 \end{pmatrix} \in R_q(n), x \in \mathbb{F}_q^n$. This is also a transitive action of $R_q(n)$ on the set of all shifts of bases of $\mathbb{F}_q^n$.

As in section 4, take the stabilizers of a hyperplane and of a point in this hyperplane. Similarly to theorem 1 one may obtain a formula connecting the weight enumerator of a code and the r generalized weight enumerator.

5.2 Radon transform in a projective space over a finite field

We may also introduce a Radon transform in the projective space $\mathbb{P}^m$ over a finite field $\mathbb{F}_q$. Take the standard action of $PGL(m, \mathbb{F}_q)$ there, let H_Ξ be the stabilizer of a hyperplane H and let H_X be the stabilizer of a point $P \in H$. Then the Radon transform is defined by $\hat{f}(\xi) = \sum_{x \in \xi} f(x)$ and the dual transform is $\check{\phi}(x) = \sum_{\xi \ni x} \phi(\xi)$. In this case, we may obtain a nice inversion formula. Let $s(\phi)$ be the operator acting on functions $\phi : \Xi \to \mathbb{R}$ defined by $s(\phi) := \sum_{\xi \in \Xi} \phi(\xi)$ and let p_n denote the number of points in $\mathbb{P}^n$, $p_n = \frac{q^{n+1}-1}{q-1}$. Then

$$f(x) = \frac{1}{q^{m-1}} (D\hat{f})\check{}(x), \tag{21}$$

where the operator $D\phi$ is defined by $D\phi(\xi) := \phi(\xi) - \frac{p_{m-2}}{p_{m-1}} s(\phi)$. We shall use this inversion formula in subsection 5.4.

5.3 An upper bound on generalized weghts

We shall need the following lemma, which is due to van der Geer and van der Vlugt.

Lemma 3 ([GV]) *For any r-subcode D of a code C holds*

$$w(D) = \frac{1}{q^r - q^{r-1}} \sum_{c \in D} w(c). \tag{22}$$

Let $GL(n, \mathbb{F}_q)$ act in the standart way on r-subcodes of an $[n,k]_q$-code C. One may easily find the subgroups such that the the corresponding homogenous spaces X and Ξ are be identified with the set of all codewords and the set of all r-subcodes respectively with the obvious incidence relation

$$c \bowtie D \Leftrightarrow c \in D.$$

Theorem 2 ([TV2]) *The r-th generalized Hamming weight of a linear $[n,k]_q$-code C satisfies the inequality*

$$d_r(C) \leq \frac{n(q^r - 1)q^{k-r}}{q^k - 1}.$$

The proof proceeds by applying twice lemma 3 and once lemma 1.

5.4 An interpretation of Nogin weight/multiplicity duality

It is well known that many results on linear codes may be naturally described via *projective systems* (also called *projective multisets*) introduced in [TV1]. To

any linear $[n,k,d]_q$-code C corresponds an n-point multiset X_C in a $(k-1)$-dimensional projective space over $\mathbb{F}_q$. This set is called the *projective system* corresponding to C; the points of the system correspond to the coordinates of the code; codewords correspond to hyperplanes in $\mathbb{P}^{k-1}$. See [TV2] for details.

Nogin [Nog] considered the projective system X_C embedded into the projectivisation $\mathbb{P}C$ of the code C. Then the coordinates of a code are some distingiushed n hyperplanes. If C has repetitions then these hyperplanes have multiplicities $\nu(H) > 1$. It is natural to assign multiplicities zero to all other hyperplanes. Up to a factor, points of $\mathbb{P}C$ are identified with codewords. The weight of a codeword c may be then expressed via the set of $\nu(H)$:

$$\mathrm{wt}(c) = \sum_{H \not\ni c} \nu(H). \tag{23}$$

Nogin obtained the following inversion formula

$$\nu(H_0) = \frac{\sum_{c \in \mathbb{P}C} \mathrm{wt}(c) - q \sum_{c \in H_0} \mathrm{wt}(c)}{q^{k-1}}. \tag{24}$$

He used then this formula to give a new construction of linear codes.

It is quite easy to reprove Eq. (24) using the Radon transform. Define the homogenous spaces in duality as in subsection 5.2. We get a pair of homogenous spaces which correspond to hyperplanes and points of $\mathbb{P}C$ with the obvious incidence $c \bowtie H \Leftrightarrow c \in H$. Then Eq. (23) becomes

$$\mathrm{wt}(c) = n - \check{\nu}(c) \tag{25}$$

and (24) is then the inversion (21).

References

[Bog1] M.Boguslavsky, 'Generalized Hermitian constants and kissing numbers', in *Proceedings of the Sixth international Workshop on Algebraic and Combinatorial Coding Theory*, Pskov, Russia, September, 1998.

[Bog2] M. Boguslavsky, 'Enumerating sublattice norms: T-functions', preprint, 1998.

[Cou] R. Coulangeon, 'Réseaux k-extrêmes,' (French), Proc. London Math. Soc. (3), **73** (1996), no. 3, 555–574.

[For1] G. Forney, Jr, 'Dimension/length profiles and trellis complexity of linear block codes,' IEEE Trans. Inform. Theory, **40**, (1994), no. 6, 1734–1752.

[For2] G. Forney, Jr, 'Density/length profiles and trellis complexity of lattices,' IEEE Trans. Inform. Theory, **40**, (1994), no. 6, 1753–1772.

[GV] G. van der Geer and M. van der Vlugt, 'Quadratic forms, generalized Hamming weights and curves over finite fields with many points', J. of Number Theory, **59**, (1996), 20–36.

[Hel1] S. Helgason, *Groups and Geometric Analysis. Integral geometry, invariant differential operators, and spherical functions*, Academic press, (1984).

[Hel2] S. Helgason, *Geometric Analysis on Symmetric Spaces*, AMS, (1994).

[HKM] T. Helleseth, T. Kløve and J. Mykkeltveit, 'The weight distribution of irreducible cyclic codes with block length $n_1\left(\left(q^l-1\right)/N\right)$,' Discr. Math., vol. 18, (1977), 179–211.

[Klo] T. Kløve, 'Support weight distribution of linear codes', Discrete Math., **106/107**, 311-313, (1992).

[Nog] D. Nogin, 'Weight/multiplicity duality', in *Proceedings of the Sixth international Workshop on Algebraic and Combinatorial Coding Theory*, Pskov, Russia, September, 1998.

[TV1] M. Tsfasman and S. Vlăduţ, *Algebraic - Geometric Codes.* Dordrecht: Kluwer Academic Publishers, (1991).

[TV2] M. Tsfasman and S.G. Vlăduţ, 'Geometric approach to higher weights'. IEEE Trans. Info. Theory, vol. 41 (1995), 1564–1588.

[Ran] R. Rankin, 'On positive definite quadratic forms,' J. London Math. Soc., **28**, (1953), 309–314.

[Wei1] V.K. Wei, 'Generalized Hamming weights for linear codes', IEEE Trans. Inform. Theory, vol.38, pp. 1125-1130, May 1992.

[Wei2] V.K. Wei, 'Generalized Hamming weights; Fundamental open problems in coding theory', *"Arithmetic, Geometry and Coding Theory,"* Walter de Gruyter, Berlin, (1996), 269–281.

Designs, Harmonic Functions, and Codes

Christine Bachoc

December 1, 1998

The theory of discrete harmonic functions and its connection with combinatorial designs was developed by Delsarte ([D]). Some orthogonal polynomials among the family of Hahn polynomials play a special role and, in particular, provide a nice caracterisation of t-designs.

On the other hand, Assmus-Mattson theorem ([MWS]) gives some conditions under which the set of codewords of given weight support a t-design; this theorem was slightly strengthened in [CDS] by using the setting of harmonic functions.

We propose here another approach, inspired by the theory of lattices and their associated "spherical theta series": we define polynomials $W_{C,f}$ associated to a binary code C and a harmonic function f of degree k and prove a MacWilliams type equality for $Z_{C,f} = (xy)^{-k} W_{C,f}$. The case $f = 1$ is the usual weight enumerator W_C of C. Hence, these polynomials turn out to be invariant polynomials (possibly with a character) for the same group as the one acting on the usual weight enumerator of the code. With the help of these harmonic weight enumerators, we give a new proof of Assmus-Mattson theorem and of its strenghthened version for extremal type II codes.

The most interesting application of these results is to the computation of the intersection numbers of the code, which are defined to be, for a fixed t-set T,

$$n_{w,i}(T) := \mathrm{Card}\{u \in C \mid wt(u) = w, |u \cap T| = i\}.$$

These numbers turn out to be a powerfull tool in classification problems, especially when one is only interested in the classification of the extremal ones (i.e. the ones with the best minimum distance), as illustrated in [FGHP1], [FGHP2].

Certain specific degree k harmonic functions $H_{k,T}$ associated to T have the nice property that $H_{k,T}(u)$ only depends on t, $|u|$, and $|u \cap T|$, and can be expressed in terms of Hahn polynomials. Hence the coefficients of $W_{C,H_{k,T}}$

are linear forms in the intersection numbers. By making use of the fact that these polynomial fall into certain subspaces of invariant polynomials, one can derive some linear equations on the $n_{w,i}(T)$ only depending on t. In [B], this method is illustrated by the case of even formally self-dual codes. Here we must consider the polynomials $(xy)^{-k}(W_{C,H_{k,T}} \pm W_{C^\perp,H_{k,T}})$ which provide relative invariants for the group $\mathcal{G}_2$ generated by $\frac{1}{\sqrt{2}}\left(\begin{smallmatrix} 1 & 1 \\ 1 & -1 \end{smallmatrix}\right)$ and $\left(\begin{smallmatrix} 1 & 0 \\ 0 & -1 \end{smallmatrix}\right)$. From the information obtained on the intersection numbers, we derive a classification of the extremal codes in length 12. The classification of extremal even formally self-dual codes was afterwards extended in [FGHP1], [FGHP2] to lengths 14, 20, 22.

References

[B] C. Bachoc, *On harmonic weight enumerators of binary codes*, preprint

[CDS] A.R. Calderbank, P. Delsarte, N,J,A. Sloane, *A strengthening of the Assmus-Mattson theorem* IEEE Trans. Inf. Th. **37** (1991), 1261-1268

[D] P. Delsarte, *Hahn polynomials, discrete harmonics and t-designs*, SIAM J. Appl. Math. **34.1** (1978), 157-166

[FGHP1] J. E. Fields, P. Gaborit, W. C. Huffmann, V. Pless, *On the classification of extremal even formally self-dual codes*, preprint

[FGHP2] J. E. Fields, P. Gaborit, W. C. Huffmann, V. Pless, *On the classification of extremal even formally self-dual codes of length* 20 *and* 22, preprint

[KMcG] S. Karlin, J. McGregor, *The Hahn polynomials, formulas and an application*, Scripta Math. **26** (1961), 33-46

[MWS] F.J. MacWilliams, N.J.A. Sloane, "The theory of error-correcting codes", North-Holland Editor, 1977

[RS] E. Rains, N.J.A. Sloane, *Self-dual codes*, to appear in the Handbook of Coding Theory

Extremal polynomials of degree $\tau+2$ and $\tau+3$, which improve the Delsarte bound for τ-designs

Svetla Nikova
Department of Mathematics and Computing Science
Eindhoven University of Technology
5600 MB Eindhoven, P.O.Box 513
The Netherlands
e-mail: svetla@win.tue.nl

Ventsislav Nikov
Department of Mathematics and Informatics
Veliko Tarnovo University
5000 Veliko Tarnovo
Bulgaria

Abstract

We investigate bounds for τ-designs in infinite polynomial metric spaces. When the necessary and sufficient conditions for improving the Delsarte bound are satisfied we derive extremal polynomials of certain degree and obtain new bounds.

1 Introduction

Let $\mathcal{M}$ be a **polynomial metric space** (PMS). They are finite metric spaces represented by P- and Q- polynomial association schemes as well as infinite metric spaces, which are completely classified as the real sphere, a real, complex or quaternions projective space and the Cayley projective plane. Hamming $H(n,r)$, Johnson $J(n,w)$ and Grassmann spaces are the most important examples of finite polynomial metric spaces.

Every polynomial metric space $\mathcal{M}$ is characterized by its metric $d(x,y)$, and normalized measure $\mu_{\mathcal{M}}(.)$.

A basic property of a polynomial metric space $\mathcal{M}$ is the existence of a decomposition of the Hilbert space $\mathcal{L}_2(\mathcal{M},\mu)$ of complex-valued quadratic-integrable functions with the usual inner product, into a direct sum of mutually orthogonal subspaces V_i of dimension r_i. Besides, there exist real polynomials $\{Q_i(t)\}_{i=0}^{\infty}$, ($Q_i(t)$ of degree i), called **zonal spherical functions**,

Any finite nonempty subset C of $\mathcal{M}$ is called a code.

Definition 1.1 *A code $C \subset \mathcal{M}$ is called a* τ-**design** *if $\sum_{x\in C} v(x) = 0$ for all $v(x) \in V_1 \oplus \cdots \oplus V_\tau$,*

The designs in $H(n,r)$ are known as orthogonal arrays. The designs in the Johnson space are nothing but the classical $t-(v,k,\lambda)$ designs.

For each a and $b \in \mathbb{N}$, one can associate the zonal spherical functions (ZSF) $\{Q_i(t)\}_{i=0}^{\infty}$ with their *adjacent systems* of orthogonal polynomials $\{Q_i^{a,b}(t)\}_{i=0}^{\infty}$ [6]. These polynomials are orthogonal with respect to the measure $\nu^{a,b}(t)$ defined by $d\nu^{a,b}(t) = c^{a,b}(1-t)^a(1+t)^b\, d\nu(t)$ ($c^{a,b}$ is a constant), i.e.

$$r_i^{a,b} \int_{-1}^{1} Q_i^{a,b}(t) Q_j^{a,b}(t) d\nu^{a,b}(t) = \delta_{ij},$$

for $i, j \geq 0$, where $Q_i^{a,b}(1) = 1$, $Q_0^{a,b}(t) \equiv 1$, $r_0^{a,b} = 1$.

A polynomial metric space $\mathcal{M}$ is called **antipodal** if for every point $x \in \mathcal{M}$ there exists a point $\overline{x} \in \mathcal{M}$ such that for any point $y \in \mathcal{M}$ we have $\sigma(d(x,y)) + \sigma(d(\overline{x},y)) = 0$.

The universal lower bound $D(\mathcal{M}, \tau)$, so called Delsarte bound, for the cardinality of a τ-design can be presented in the following form [5]:

$$|C| \geq D(\mathcal{M}, \tau) = 2^{\theta} c^{0,\theta} \sum_{i=0}^{k} r_i^{0,\theta}, \tag{1}$$

where $\theta \in \{0, 1\}$ and $\tau = 2k + \theta$.

The bound (1) can be obtained by using the polynomial $f^{(\tau)}(t) = (t+1)^{\theta}((Q_k^{1,\theta}(t))^2$ in the following Theorem.

Theorem 1.2 *Let $C \subset \mathcal{M}$ τ-design and let $f(t)$ be a real nonzero polynomial such that*

(B1) *$f(t) \geq 0$, for $-1 \leq t \leq 1$,*

(B2) *the coefficients in the ZSF expansion $f(t) = \sum_{i=0}^{k} f_i Q_i(t)$ satisfy $f_0 > 0$, $f_i \leq 0$ for $i = \tau + 1, \ldots, k$.*

Then, $|C| \geq f(1)/f_0 = \Omega(f)$.

Definition 1.3 *A polynomial $f(t) \in B_{\mathcal{M},\tau}$ is called $B_{\mathcal{M},\tau}$-extremal if*

$$\Omega(f) = \max\{\Omega(g) : \; g(t) \in B_{\mathcal{M},\tau}, \; \deg(g) \leq \deg(f)\}.$$

The coefficient f_0, which is very important for our investigations, can be expressed as follows

$$f_0 = \int_{-1}^{1} f(t) d\nu(t). \tag{2}$$

The conditions of the Theorem 1.2 are independent from the multiplication of f(t) with a positive constant. Thus we shall not distinguish proportional polynomials from $B_{\mathcal{M},\tau}$.

In this paper we apply the necessary and sufficient conditions for the optimality of the Delsarte bound and when it is not the best bound possible we give an analytical form of the $B_{\mathcal{M},\tau}$-extremal polynomials.

2 Preliminary results

In this paper we consider only infinite polynomial metric spaces. We define the following linear functional, which was introduced and investigated in [7, 8].

$$G_\tau(\mathcal{M}, f) = \frac{f(1)}{D(\mathcal{M}, \tau)} + \sum_{i=1}^{k+\theta} \rho_i^{(\tau)} f(\alpha_i) \tag{3}$$

where $\alpha_i, \rho_i^{(\tau)}$ are defined in [Theorem2.1 [7]].

We will call it briefly "test" functions Similar test functions for codes were introduced and investigated by Boyvalenkov, Danev and Bumova in [1] (for $\mathcal{M} = \mathbf{S}^{n-1}$) and by Boyvalenkov and Danev [2] (in the general case).

Theorem 2.1 *The bound $D(\mathcal{M}, \tau)$ can be improved by a polynomial $f(t) \in B_{\mathcal{M},\tau}$ of degree at least $\tau+1$, if and only if $G_\tau(\mathcal{M}, Q_j) < 0$ for some $j \geq \tau+1$. Moreover, if $G_\tau(\mathcal{M}, Q_j) < 0$ for some $j \geq \tau+1$, then $D(\mathcal{M}, \tau)$ can be improved by a polynomial in $B_{\mathcal{M},\tau}$ of degree j.*

This theorem gives us necessary and sufficient conditions for the optimality of the Delsarte bound.

Lemma 2.2 **a)** *Let $\mathcal{M}$ be PMS, then $G_\tau(\mathcal{M}, Q_{\tau+1}) > 0$*

b) *For $\mathcal{M}$ antipodal*

$$G_\tau(\mathcal{M}, Q_{\tau+2}) \begin{cases} > 0, & \text{for } \tau = 2k \\ = 0, & \text{for } \tau = 2k+1. \end{cases}$$

The "test" functions $G_\tau(\mathcal{M}, Q_{\tau+2})$, $G_\tau(\mathcal{M}, Q_{\tau+3})$ are negative for $3 \leq n \leq N(\tau, \mathcal{M})$. The exact values of $N(\tau, \mathcal{M})$ are given in [8].

3 Extremal polynomials of degree $\tau+2$ and $\tau+3$

¿From the previous section we have necessary and sufficient conditions for improving the Delsarte bound by using linear programming. The investigations of the test functions for designs show that the smallest possible degree of an improving polynomial in non-antipodal PMS is $\tau+2$ and for antipodal and is $\tau+3$ (see Lemma 2.2).

Theorem 3.1 *Let $\mathcal{M}$ be non-antipodal PMS. Then, any $B_{\mathcal{M},\tau}$-extremal polynomial of degree $\tau+2$ ($\tau = 2k+\theta$) has the form*

$$f^{(\tau)}(t; \tau+2) = (1+t)^{1-\theta}[q(t+1) + (1-t)][\eta Q_{k-1+\theta}^{1,1-\theta}(t) + Q_{k+\theta}^{1,1-\theta}(t)]^2 \tag{4}$$

Corollary 3.2 *Let $\mathcal{M}$ be a non-antipodal PMS and let τ be an integer. Then*

$$B(\mathcal{M}, \tau) \geq S(\mathcal{M}, \tau) = \Omega(f^{(\tau)}(t; \tau+2)).$$

Corollary 3.3 *Let $\mathcal{M}$ be a non-antipodal PMS and let τ be an integer. Then $S(\mathcal{M},\tau) > D(\mathcal{M},\tau)$ if and only if $G_\tau(\mathcal{M}, Q_{\tau+2}) < 0$.*

As we mentioned before, for antipodal PMS the corresponding $B_{\mathcal{M},\tau}$-extremal polynomial is of degree $\tau+3$. We can prove in a similar way analogous theorem for the form of this polynomials.

Theorem 3.4 *Let $\mathcal{M}$ be antipodal PMS. Then, any $B_{\mathcal{M},\tau}$-extremal polynomial of degree $\tau+3$ ($\tau = 2k+\theta$) has the form*

$$f^{(\tau)}(t;\tau+3) = (1+t)^\theta [q(t+1)+(1-t)][\eta_1 Q_{k-1}^{1,\theta}(t) + \eta_2 Q_k^{1,\theta}(t) + Q_{k+1}^{1,\theta}(t)]^2 \qquad (5)$$

Corollary 3.5 *Let $\mathcal{M}$ be an antipodal PMS and let τ be an integer. Then*

$$B(\mathcal{M},\tau) \geq S(\mathcal{M},\tau) = \Omega(f^{(\tau)}(t;\tau+3)),$$

We compared by computer the results in [3, 4] and the bounds from Corollary 3.2 and 3.5. We also compared the polynomials, which we use for obtaining new bounds in [3] and [4], and the polynomials described in (4) and (5). This investigation showed the coincidence between the corresponding polynomials and coincidence between the corresponding bounds.

Acknowlidgements. The authors thank to Peter Boyvalenkov and Henk van Tilborg for the helpful discussions and comments.

References

[1] P.G.Boyvalenkov, D.P.Danev, S.P.Dimcheva, Upper bounds on the minimum distance of spherical codes, *IEEE Transactions on Information Theory* 42, 1996, 1576-1581.

[2] P.G.Boyvalenkov, D.P.Danev, On Linear Programming Bounds for the Codes in Polynomial Metric Spaces, to appear in *Problems of Information Transmission*, English translation from *Problemy Peredachi Informatsii.*

[3] P.G.Boyvalenkov, S.I.Nikova, Improvements of the lower bounds for the size of some spherical designs, *Mathematica Balkanica*, to appear.

[4] P.G.Boyvalenkov, S.I.Nikova, On Lower Bounds on the Size of Designs in Compact Symmetric Spaces of Rank 1, *Archiv der Mathematik* 68, 1997, 81-88.

[5] P.Delsarte, J.-M.Goethals, J.J.Seidel, Spherical codes and designs, *Geometriae Dedicata* 6, 1977, 363-388.

[6] V.I.Levenshtein, Universal bounds for codes and designs, Chapter in Handbook of Coding Theory, V.Pless, W.C.Huffman, and R.A.Brualdi, Eds. Amsterdam: Elsevier, to appear.

[7] S.I.Nikova, V.S.Nikov, Necessary and sufficient conditions for improving the Delsarte bound, Proc. Sixth International Workshop on Algebr. and Comb. Coding Theory, Pskov, September 6-12, 1998.

[8] S.I. Nikova, *Bounds for Designs in Infinite Polynomial Metric Spaces*, Ph.D. Thesis, Eindhoven University of Technology, 1998.

On the maximum T-wise independent systems of Boolean functions*

Vladimir I. Levenshtein

Keldysh Institute for Applied Mathematics, RAS, Moscow
Email: leven@spp.keldysh.ru

The problem of finding the maximum number $N(n,T)$ of Boolean functions in n variables of which any T are independent is considered. The independence of functions is treated as that of the output random variables in uniformly distributed n input variables. One can see that $N(n,T)$ equals the maximum length N of a binary (not necessarily linear) code of the size 2^n which has the dual distance at least $T+1$ (or, equivalently, forms an orthogonal array of strength T). Moreover, linear codes give rise to systems of linear functions. This relation allows one to use design and coding theory in order to estimate $N(n,T)$ (see [1], [5]).

First, note that the minimum size of a binary orthogonal array of length $n+1$ and strength $2m+1$ is equal to the doubled minimum size of a orthogonal array of length n and strength $2m$ [2], [5]. This implies

$$N(n,2m) = N(n+1,2m+1) - 1 \tag{1}$$

and allows one to consider only the case of odd T. The Rao bound for orthogonal arrays implies the following necessary condition:

$$\sum_{j=0}^{m} \binom{N-1}{j} \le 2^{n-1} \tag{2}$$

which is satisfied for any $(2m+1)$-wise independent system from N (in particular, $N = N(n,2m+1)$) Boolean functions in n variables. Similarly, special cases of the author bound [4] for orthogonal arrays imply the necessary condition:

$$2^N \le 2^{n+1} \sum_{j=0}^{i} \binom{N-1}{j} \quad \text{if} \quad d_i(N-2) \le 2m+1 < d_{i-1}(N-2), \tag{3}$$

*The research was partially supported by the Russian Foundation for Basic Research under grant 98-01-00146.

$\forall 1 \le i_1 < i_2 < \dots < i_T \le N \quad \forall \sigma_1 \dots \sigma_T \ \sigma_i \in \{0,1\}$.

$\left|\{x \in H_2^n : f_{i_1}(x) = \sigma_1, \dots f_{i_T}(x) = \sigma_T\}\right| = 2^{n-T}$.

H_2^n Hamming Scheme

where $d_i(h)$ is the smallest root of the Krawtchouk polynomial

$$K_i^h(x) = \sum_{j=0}^{i} (-1)^j \binom{x}{j}\binom{h-x}{i-j}$$

of degree i. Notice, that the extended Goley (24,12,8)-code generates a 7-wise independent system from 24 linear functions in 12 variables which gives equality in the both inequalities (2) and (3), since $d_3(22) = 7$ and $\sum_{j=0}^{3}\binom{23}{j} = 2^{11}$. This system is maximum and $N(12,7) = 24$.

Recently the author found one more bound for orthogonal arrays based on the existence of a relationship between linear programming bounds on the size of orthogonal arrays and block designs [6]. Special cases of this bound imply for any integer w, $1 \le w \le N/2$, the following necessary condition for the existence of a $(2m+1)$-wise independent system from N (in particular, $N = N(n, 2m+1)$) Boolean functions in n variables:

$$\binom{N}{w} \le 2^n \binom{N}{i} \quad \text{if} \quad d_i(N,w) \le m+1 < d_{i-1}(N,w), \tag{4}$$

where $d_i(h,w)$ is the smallest root of the polynomial $\left(J_i^{h,w}(x) - J_{i+1}^{h,w}(x)\right)/x$ with

$$J_i^{h,w}(x) = \sum_{j=0}^{i} (-1)^j \frac{\binom{i}{j}\binom{h+1-i}{j}}{\binom{w}{j}\binom{h-w}{j}}\binom{x}{j}.$$

The linear (n,k)-codes with $n = 2^l$ and $k = ml+1$, $1 \le m \le 2^{[(l-1)/2]}$, which are dual to the extended BCH codes with designed distance $2m+1$, have the dual distance $2m+2$ (or more) and give rise to $(2m+1)$-wise independent systems from 2^l linear Boolean functions in $ml+1$ variables. Therefore, we have

$$N(lm+1, 2m+1) \ge 2^l \quad \text{if} \quad 1 \le m \le 2^{[(l-1)/2]}.$$

Due to (2) this bound is attained for $m = 1$. However, for larger m some better results are obtained from the known nonlinear codes. In particular, the Kerdock code of length $n = 2^{2l}$ and size n^2 has the dual distance 6 and together with (2) implies that

$$N(4l, 5) = 2^{2l} \quad \text{for} \quad l \ge 2.$$

Analogously, from (2) and the existence of the nonlinear Delsarte-Goethals code of length $n = 2^{2l}$ and size $n^3/2$ which has the dual distance 8 it follows that

$$2^{2l} \le N(6l-1, 7) \le c2^{2l} + 1,$$

where $c = (3/2)^{1/3} = 1,1447$.

The Varshamov bound on the existence of linear codes and (1) imply that the condition

$$1+\sum_{i=0}^{2m-1}\binom{N-2}{i}\le 2^{n-1} \tag{5}$$

is sufficient for the existence of a $(2m+1)$-wise independent system from N linear Boolean functions in n variables. Hence $N(n,2m+1)$ is larger than or equal to the maximum integer N satisfying (5). For $m=1$ the sufficient condition (5) coincides with necessary condition (2).

At the asymptotic process when $n\to\infty$ and $T/n\to\delta$ $(0<\delta\le 1)$ this bound gives

$$N(n,T)\gtrsim\begin{cases} n/x(\delta) & \text{if } 0<\delta\le 1/2\\ n & \text{if } 1/2\le\delta\le 1\end{cases}$$

where $x(\delta)$ is the unique positive root of the equation $x=H(\delta x)$ and $H(p)$ is the Shannon entropy. At the same process (2) shows that

$$N(n,T)\lesssim\begin{cases} n/f(\delta) & \text{if } 0<\delta\le 1/2\\ n & \text{if } 1/2\le\delta\le 1\end{cases}$$

where

$$f(\delta)=1-H\left(\frac{1}{2}-\sqrt{\delta(1-\delta)}\right).$$

Thus

$$N(n,T)\sim n \text{ if } 1/2\le\delta\le 1.$$

The better (for $\delta<0.272$) asymptotic upper bound can be obtained from (5). It is known that if

$$\lim_{h\to\infty}\frac{i}{h}=\zeta,\ \lim_{h\to\infty}\frac{w}{h}=\eta,\ \text{where } 0\le\zeta\le\eta,\ 0\le\eta\le 1/2,$$

then

$$\frac{d_i(h,w)}{h}=\xi_\eta(\zeta)+o(1)\ ,$$

where

$$\xi_\eta(x)=\frac{\eta(1-\eta)-x(1-x)}{1+2\sqrt{x(1-x)}}$$

is a decreasing continuous function which maps the interval $[0,\eta]$ onto $[0,\eta(1-\eta)]$ (see [3], [7]). The inverse function $\xi_\eta^{-1}(x)$ can be expressed in the following explicit form:

$$\xi_\eta^{-1}(x)=\frac{1}{2}\left(1-\sqrt{1-4\left(\sqrt{\eta(1-\eta)-x(1-x)}-x\right)^2}\right).$$

The bound (5) gives

$$N(n,T) \lesssim n/g(\delta) \quad \text{if} \quad 0 < \delta \leq 1/2,$$

where

$$g(\delta) = \max_{\frac{1}{2}(1-\sqrt{1-2\delta}) \leq \eta \leq \frac{1}{2}} \left(H(\eta) - H(\xi_\eta^{-1}(\frac{\delta}{2})) \right).$$

Thus, the main asymptotic results for error-correcting codes can be extended to the problem of constructing T-wise independent systems of functions although this problem is connected with the existence of (in general, nonlinear) orthogonal arrays.

References

[1] F.J. MacWilliams and N.J.A. Sloane, *The Theory of Error-Correcting Codes.* New York: North-Holland, 1977.

[2] E.Seiden, R.Zemach, "On orthogonal arrays", *Ann. Math. Stat.*, **37** (1966), 1355–1370.

[3] R.J. McEliece, E.R. Rodemich, H. Rumsey, Jr., and L.R. Welch, "New upper bounds on the rate of a code via the Delsarte–MacWilliams inequalities", *IEEE Trans. Inform. Theory*, **23** (1977), 157–166.

[4] V.I. Levenshtein, "Krawtchouk polynomials and universal bounds for codes and designs in Hamming spaces", *IEEE Trans. Inform. Theory*, **41**, no. 5 (1995), 1303–1321.

[5] V.I. Levenshtein, "Split orthogonal arrays and maximum independent resilient systems of functions", *Designs, Codes and Cryptography*, **12**, no. 2 (1997), 131–160.

[6] V.I. Levenshtein, "Equivalence of Delsarte's bounds for codes and designs in symmetric association schemes and some applications, *Discrete Mathematics*, **197/198** (1998).

[7] V.I. Levenshtein, "Universal bounds for codes and designs", in *Handbook of Coding Theory*, V. Pless and W.C. Huffman, Eds. Amsterdam: Elsevier, 1998.

DECODING CONVOLUTIONAL CODES USING A MULTIPROCESSOR BIDIRECTIONAL CREEPER ALGORITHM

V. Imtawil and D. J. Tait
School of Engineering,
The University,
Manchester M13 9PL UK.
Email: mbhmivv@afs.mcc.ac.uk and david.tait@man.ac.uk

Abstract: Creeper is an algorithm for the sequential decoding of convolutional codes which combines the best properties of both the stack and the Fano algorithms [1]. We propose an efficient decoding strategy based on a multiprocessor architecture we call the Bidirectional Creeper Algorithm (BCA). The proposed technique comprises two identical autonomous decoders (processors) each searching for the correct path in the code tree but one working in the forward direction and the other working in the backward direction. A third high speed processor is used to monitor the progress of both decoders, terminating the decoding process when agreement is detected. It is shown that the proposed technique for decoding convolutional codes can significantly reduce the computational variability of the conventional or unidirectional Creeper algorithm (UCA). In addition, the decoding speed is much faster than that of the UCA.

1 Introduction

A major problem in sequential decoding of convolutional codes is the large computational variability which is highly undesirable. Noisy frames need a large number of computations, and decoding times occasionally exceed some upper limit, causing information to be erased [2]. Recently some work designed to alleviate this problem based on multiprocessor techniques has been reported [3]-[5]. In 1997, Kallel and Li proposed a bidirectional sequential decoding strategy [5] using the stack algorithm which was shown to substantially reduce the computational variability but which has poor error performance. We propose and analyse in this paper an alternative efficient decoding system which applies a bidirectional decoding search and a multiprocessor architecture to the Creeper Algorithm [1]. It is shown that the proposed system significantly reduces the computational variability and speeds up the decoding process while retaining as good an error performance as the Unidirectional Creeper Algorithm (UCA). An introduction to the Creeper Algorithm is presented in Section 2 and some background on backward coding/decoding is then presented in Section 3. In section 4, our proposal for a Bidirectional Creeper Algorithm (BCA) is outlined. Simulation results are given in Section 5, followed by a conclusion and discussion in Section 6.

2 Creeper: an algorithm for sequential decoding.

Creeper [1] is an efficient algorithm for the sequential decoding of convolutional codes. It combines the best properties of the stack algorithm and the Fano algorithm. Memory requirements are low; at most $2L$ nodes need to be stored where L is the code tree length. The main conceptual idea is to move forward in the code tree as far as possible and when leaving the current subtree, store the best metric of the nodes that have been examined (i.e., have had their metrics computed) but not visited. This value is used to determine the later actions taken by the decoder.

Let n_{cur} denote the node currently visited by the decoder. Assuming a rate 1/2 encoder, the successors of n_{cur} are called n_x and n_y with cumulative metrics μ_x and μ_y respectively. We will also assume that $\mu_x \geq \mu_y$. The algorithm uses two stacks, one is used to store nodes and is called the node stack and the other, called the threshold stack (or μ_T stack),

is used to store metric values of selected nodes in the node stack. The algorithm starts at the root node and begins by computing the metrics of the successor nodes and the decoder then decides whether to move forward or not. If both n_x and n_y are considered "good" ($\mu_x \geq T$ and $\mu_y \geq T$), both nodes are stored in the node stack. If only n_x is good, the decoder moves to this node but the node is not stacked. If both nodes are considered "bad", the decoder moves to the node on the top of the node stack - a "sideways-backwards" move in Creeper terminology. For convenience we denote by *NP* the stack pointer that points to the top element of the node stack, and by *TP* the stack pointer that points to the top element of the threshold stack. A stacked node is denoted by $n(NP\text{-}k)$ where k is the location of the node on the stack relative to the top element. Along with each node, a value $\mu_\Gamma(TP\text{-}k)$ is stored in the threshold stack but not every node has its metric entered in the threshold stack. A node on the node stack with an entry in the μ_Γ stack is said to be a *T*-node and those nodes that do not have an entry on the μ_Γ stack are called non-*T*-nodes. Along with each node in the node stack, a flag is stored indicating whether it is a *T*-node or not. The flag corresponding to $n(NP\text{-}k)$ is denoted by $F(NP\text{-}k)$. A threshold $T = Q(\mu_\Gamma(TP - 1))$ is used to determine which nodes are stored; those with metrics above *T* are placed on the node stack ($Q(x) = \lfloor x/\Delta \rfloor \Delta$ is a quantisation function that converts an integer x to the largest multiple of Δ not larger than x). When a node, n_x, is found with a metric exceeding μ_{max}, the maximum metric of all nodes previously visited, n_x and n_y are stacked on the node stack as *T*-nodes, and the two μ_Γ-values corresponding to theses nodes, are placed on the μ_Γ stack. Otherwise, if $\mu_x < \mu_{max}$, n_x and n_y are stored as non *T*-nodes on the node stack.

The algorithm is initialised with $\mu_{max} \leftarrow -\infty$, $\mu_\Gamma(-1) \leftarrow -\infty$, $n(0) \leftarrow root$, $n_{cur} \leftarrow root$, and $TP = NP = 0$. Next, the successor metrics μ_x, μ_y and the threshold $T = Q(\mu_\Gamma(TP - 1))$ are computed and one of the rules in Table 1 is applied.

After the application of the rule, the new successor metrics and the new threshold are computed, and the decoder applies one of the rules again, and so on. The algorithm terminates when n_{cur} is at the end of the code tree. Figure 1-8 show examples of how the decoder proceeds.

Table 1: Rules used in Creeper

Rule	Condition	decoder actions
one	$T \leq \mu_y$ and $\mu_x > \mu_{max}$	$n_{cur} \leftarrow n_x$ $n(NP + 1) \leftarrow n_y$ $n(NP + 2) \leftarrow n_x$ $\mu_\Gamma(TP + 1) \leftarrow \mu_y$ $\mu_\Gamma(TP + 2) \leftarrow -\infty$ $F(NP + 1) \leftarrow 1$ $F(NP + 2) \leftarrow 1$ $NP \leftarrow NP + 2$ $TP \leftarrow TP + 2$ $\mu_{max} \leftarrow \mu_x$
two	$T \leq \mu_y$ and $\mu_x \leq \mu_{max}$	$n_{cur} \leftarrow n_x$ $n(NP + 1) \leftarrow n_y$ $n(NP + 2) \leftarrow n_x$ $F(NP + 1) \leftarrow 0$ $F(NP + 2) \leftarrow 0$ $NP \leftarrow NP + 2$
three	$\mu_y < T \leq \mu_x$	$n_{cur} \leftarrow n_x$ $\mu_\Gamma(TP) \leftarrow \max(\mu_x, \mu_\Gamma(TP))$ $\mu_{max} \leftarrow \max(\mu_x, \mu_{max})$

four	$\mu_x < T$ and $F(NP - 1) = 1$ and $\text{Max}(\mu_x, \mu_\Gamma(TP))$ $\geq Q(\mu_\Gamma(TP - 3))$	$n_{cur} \leftarrow n(NP - 1)$ $\mu_\Gamma(TP - 1) \leftarrow \max(\mu_x, \mu_\Gamma(TP))$ *exchange*$(n(NP), n(NP - 1))$ *exchange*$(F(NP), F(NP - 1))$ $\mu_\Gamma(TP) \leftarrow -\infty$
five	$\mu_x < T$ and $F(NP - 1) = 1$ and $\max(\mu_x, \mu_\Gamma(TP))$ $< Q(\mu_\Gamma(TP - 3))$	$n_{cur} \leftarrow n(NP - 1)$ $\mu_\Gamma(TP - 2) \leftarrow \max(\mu_x, \mu_\Gamma(TP),$ $\mu_\Gamma(TP - 2))$ $NP \leftarrow NP - 2$ $TP \leftarrow TP - 2$
six	$\mu_x < T$ and $F(NP - 1) = 0$	$n_{cur} \leftarrow n(NP - 1)$ $\mu_\Gamma(TP) \leftarrow \max(\mu_x, \mu_\Gamma(TP))$ $NP \leftarrow NP - 2$

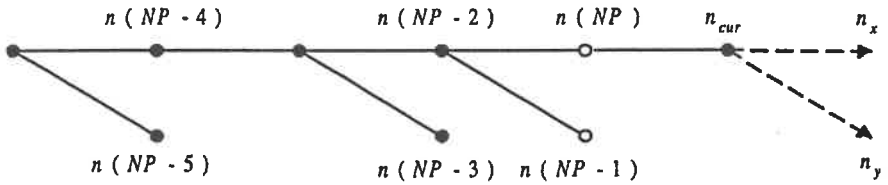


Figure 1: The situation could look like this before any of the rules one, two, three or six is applied. The *T*-nodes are marked black, the others white.

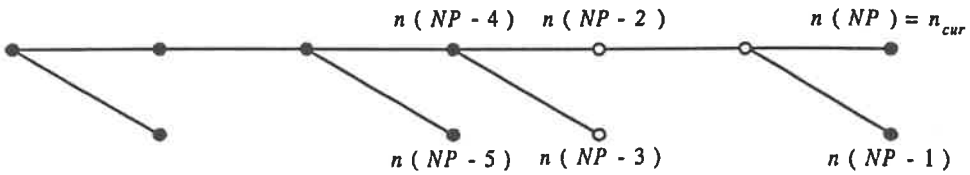


Figure 2: Rule one: move forward and stack both successors as *T*-nodes and stack their μ_Γ values on the μ_Γ stack.

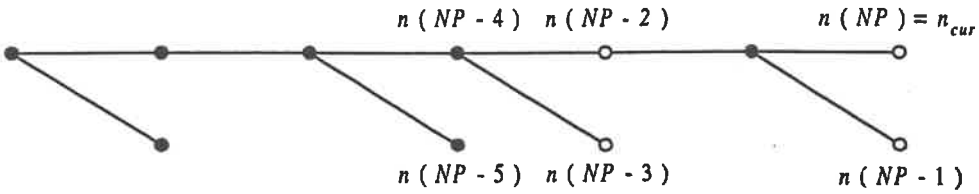


Figure 3: Rule two: move forward and stack both successors as non-*T*-nodes.

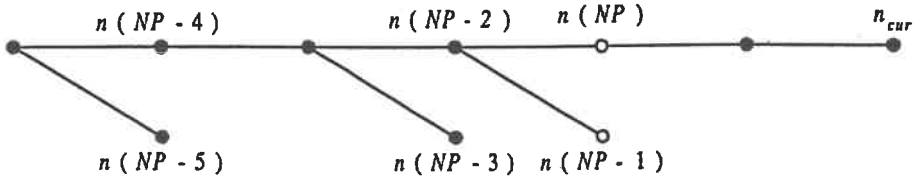


Figure 4: Rule three: move forward.

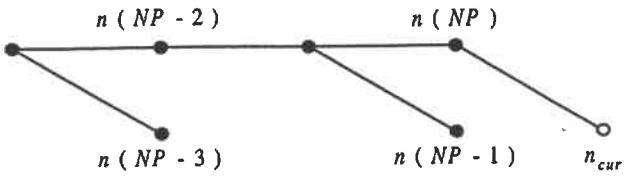


Figure 5: Rule six: move sideways-backwards to the closest (non-*T*-)node and delete the two top elements on the node stack.

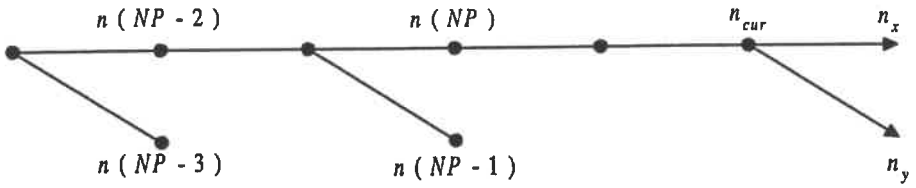


Figure 6: The situation could look like this before either rule four or five is applied.

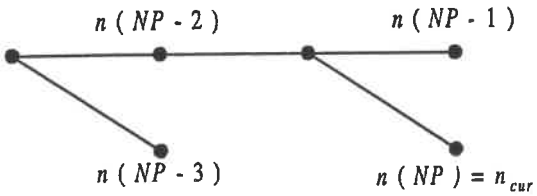


Figure 7: Rule four: move sideways-backwards and exchange the two top elements (*n* (*NP*) and *n*(*NP*-1)) on the node stack.

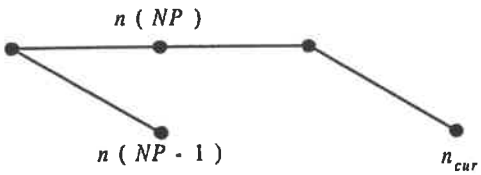


Figure 8: Rule five: move sideways-backwards and delete the top elements on the node stack.

3. Backward coding/decoding.

Every path in the trellis or tree diagram of a convolutional code can be viewed in reverse by starting from the final state (usually the all-zero node) and terminating at the initial state (usually the all-zero node). So from the original forward code, one can generate a backward tree (or trellis) which we call the backward code. A backward code sequence is obtained by simply reversing the original forward code sequence. In addition, the corresponding information sequences are also reversals of each other. The generator matrix of an (*n*, *k*, *m*) backward code can be expressed in terms of the subgenerators $g_i^{(j)}(D)$ of its corresponding forward code by [5]:

$$G_r(D) = D^m \begin{bmatrix} g_1^{(n)}(D^{-1}) & g_1^{(n-1)}(D^{-1}) & \cdots & g_1^{(1)}(D^{-1}) \\ g_2^{(n)}(D^{-1}) & g_2^{(n-1)}(D^{-1}) & \cdots & g_2^{(1)}(D^{-1}) \\ \vdots & \vdots & \ddots & \vdots \\ g_k^{(n)}(D^{-1}) & g_k^{(n-1)}(D^{-1}) & \cdots & g_k^{(1)}(D^{-1}) \end{bmatrix}$$

4. Bidirectional Creeper Algorithm

In this section, we present a novel version of the Creeper Algorithm. The basic idea is to apply the Creeper technique to both the forward and backward trees simultaneously to produce a Bidirectional Creeper Algorithm (BCA).

4.1 Description of the decoder

The system comprises two identical decoders (processors), a forward decoder (FD) and a backward decoder (BD), aided by a high speed control processor, CP. Each decoder uses three stacks, a node stack (NS), a threshold stack (TS) and a supplementary stack (SS). The proposed decoder is shown in Figure 9.

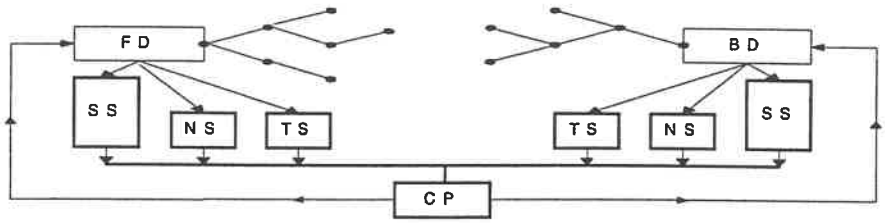


Figure 9: Illustration of a bidirectional Creeper decoding system

- The forward decoder (FD) searches for the correct path in the code tree using the UCA.
- The backward decoder (BD) searches for the correct path in the reverse code tree using the UCA.
- The node stacks serve the same purpose as those used in the UCA..
- The threshold stacks are the same as those used in the UCA.
- The control processor (CP) controls and terminates the decoding process.
- The supplementary stacks store certain good visited nodes.

To explain the operation of the supplementary stacks, let us define d_f as the farthest depth reached by the forward decoder, d_b the farthest depth reached by the backward decoder, d_{fcur} the current depth of the forward decoder, d_{bcur} the current depth of the backward decoder, and β as an adjustable factor. Storing rules applied to the supplementary stack are as follows:

1) In rule three of the conventional Creeper, store n_x if the depth of node n_x, is greater than or equal to $d_f - \beta$ (or $d_b - \beta$).

2) In rule five and six of the conventional Creeper, the top two nodes in the node stack are transferred to the supplementary stack if their depths are greater than or equal to $d_f - \beta$ (or $d_b - \beta$).

4.2 Decoding of a data block

The first decoding steps of both the FD and the BD are the same as the UCA. Both FD and BD are working in parallel at full speed and do not communicate with each other. The CP is always informed about the progress of both the FD and the BD. This information is obtained by the CP from the nodes stored in the NSs and the SSs. The CP is continually checking if both the FD and the BD merge at a common state in the code tree. The CP will occasionally interrupt the operation of the decoders when doing node comparisons. Whenever the CP finds that both decoders merge at a common state in the code tree, it will then send a signal to stop both decoders and terminate the decoding process. Some good nodes may be re-

visited but these must not be stored in the SSs, and this job is also managed by the CP. The three stopping rules used in the system are as follows:

1) If $d_{fcur} + d_b \geq L$, compare the current node being examined by the forward decoder with all the nodes with depth equal to $L - d_{fcur}$ in the NS and the SS of the backward decoder. If one or more merging paths is found, stop decoding. Among all merging paths, select the one with the highest cumulative metric as the decoded path.

2) If $d_{bcur} + d_f \geq L$, compare the current node being examined by the backward decoder with all the nodes with depth equal to $L - d_{bcur}$ in the NS and the SS of the forward decoder. If one or more merging paths is found, stop decoding. Among all merging paths, select the one with the highest cumulative metric as the decoded path.

3) If the computational limit C_{lim} is exceeded, stop decoding and erase the block.

4.3 An example of decoding

To illustrate how the system operates, we give a decoding example. The code used in the example is a small rate-1/2 code with constraint length K = 4 and octal generators (15, 13). The frame length is just 7 bits. Assume that the message sequence is 1010011 so that the corresponding code word is 11 10 10 01 01 00 01 11 10 11 and the received sequence is 11 10 10 01 **00** 00 **1**1 11 10 **0**1 (where error bits are shown in bold).

Table 2: Illustration of how the system proceeds in the example.

Cycle	F_{ncur}	B_{ncur}	T_f	T_b	FNS	BNS	FSS	BSS
0	Root	root	- 20.4	- 10.2	-	-	-	-
1	1	0	- 20.4	- 10.2	1, 0	0, 1	-	-
2	10	1	- 20.4	- 10.2	10, 11, 1, 0	1, 0	-	11
3	101	11	- 20.4	- 10.2	101, 100, 10, 11, 1, 0	1, 0	-	110, 11
4	1010	110	- 20.4	- 10.2	1010, 1011, 101,100, 10, 11, 1, 0	1, 0	-	110, 11
5	10100	0	- 20.4	- 20.4	10100, 10101, 1010, 101, 100, 10, 11, 1, 0	0, 1	-	110, 11
6	101001	1	- 20.4	- 20.4	10100, 10101, 1010, 1011, 101, 100,10,11, 1, 0	1, 0	-	110, 11
7	1010010	11	- 20.4	- 20.4	1010010, 1010011, 10100, 10101, 1010, 1011, 101, 100, 10, 11, 1, 0	1, 0	-	110, 11
8	10100100	110	- 20.4	- 20.4	10100100, 1010010, 1010011, 10100, 10101, 1010, 1011, 101, 100, 10, 11, 1, 0	1, 0	-	110, 11

F_{ncur} = current node of the forward decoder
B_{ncur} = current node of the backward decoder
T_f = threshold used by the forward decoder
T_b = threshold used by the backward decoder
FNS = forward node stack
BNS = backward node stack
FSS = forward supplementary stack
BSS = backward supplementary stack

Table 2 shows the details of how the system proceeds. Both decoders start at the root nodes and the control processor finds the merging point in the eighth step. Figure 10 shows how both decoders proceed by illustrating the same example using a trellis diagram representing of the code trees.

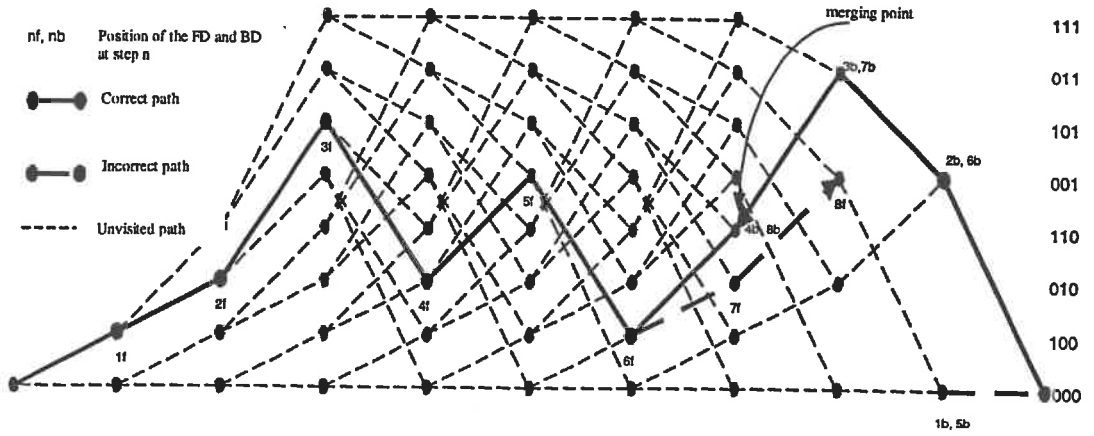


Figure 10: Example of a decoding trellis

5. Simulation results

In this section, we present the results of simulations performed to evaluate the performance of our proposed architecture. Convolutional codes that are most suitable for bidirectional decoding algorithms are systematic almost bidirectional optimum distance profile (SABODP) codes such as those found by Kallel and Li [5]. Our simulations were conducted using a rate-1/2 SABODP convolutional code with constraint length K = 24 and octal generators (55231643, 61346255). The modulation considered was binary phase-shift keying with a post demodulator error probability
$p_e = 0.045$. The frame size was 80 bits. 10000 blocks were run for each simulation.

Figure 11 shows the distribution of the total number of computations per decoded block of the UCA and BCA for four cases: no storage in the supplementary stacks; $\beta = 0$; $\beta = 5$; and $\beta = 10$. The maximum allowed number of computations $C_{\lim}$ used in the BCA is based on both metric computations and node comparisons. The time used to finish one node comparison is assumed to be much faster than that for one metric computation but the exact ratio depends on the speed of the control processor. It can be seen from Figure 11 that the slope of the curve is steeper (better) when β is increased. When the supplementary stacks are not used (no storage case), the slope of the curves of both the UCA and BCA are almost parallel. The BCA clearly results in a significant reduction of the computational variability over the UCA when the supplementary stacks are used.

There is a tradeoff between variability and storage requirements. Table 3 shows the maximum number of nodes stored in the supplementary stacks (for both forward and backward decoders) and the maximum number of operations (metric computations plus node comparisons) used to decode the most noisy block when $C_{\lim}$ and the size of the supplementary stacks were set high enough to allow every block to be decoded successfully.

Table 3: Comparison of the maximum number of nodes stored in supplementary stacks and the maximum number of operations

L	Max. no. of nodes stored in SSs	Max. no of Ops.
$\beta = 0$	131	49,662
$\beta = 5$	369	41,479
$\beta = 10$	501	11,508

6. Conclusion and discussion.

We have applied a three processor architecture to the Creeper Algorithm implementing a bidirectional decoding search. It has been shown that the system gives a significant improvement of computational variability over the unidirectional Creeper algorithm while maintaining the latter's good error performance. The adjustable parameter β was shown to be an important factor for controlling the number of nodes stored in the supplementary stacks. The more nodes stored in the supplementary stacks the faster the decoder. The BCA gives better error performance than the bidirectional stack decoder (BSD) proposed by Kallel and Li [5]. Moreover, the BCA does not have to sort stacks to get the top nodes before performing the next decoding cycle as the BSD does and the use of the high speed control processor, CP, allows both decoders to work in parallel at their full speed without having to communicate with each other yielding a further speed up of the decoding process.

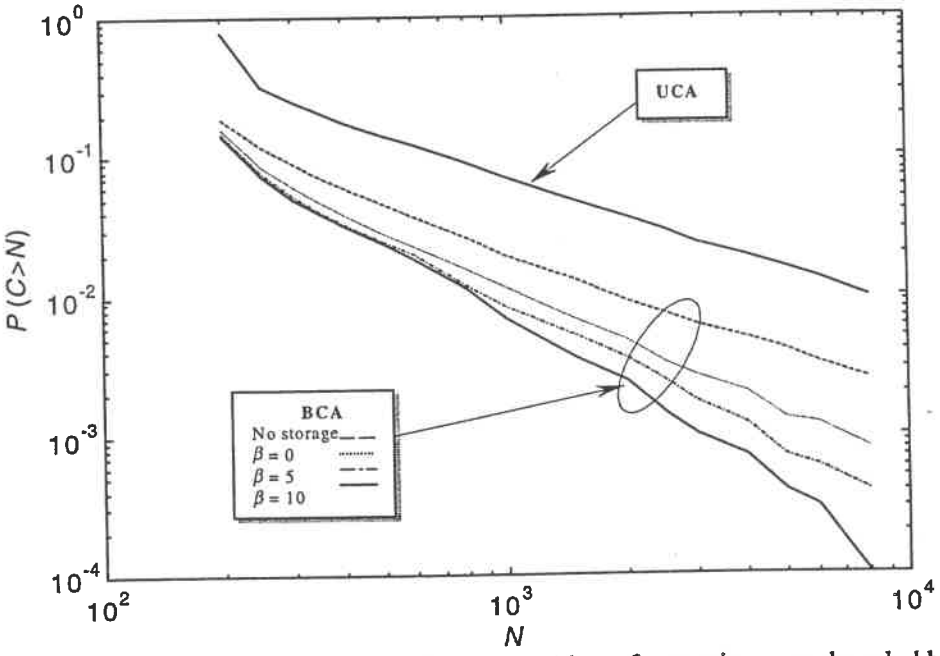


Figure 11: The distribution of the total number of operations per decoded block

References

1. Johan A. Nystrom, Creeper: an algorithm for sequential decoding, Ph.D. thesis, Department of Information Theory, Lund University, Sweden 1993.
2. S. Lin and D. J. Costello, Jr., Error Control Coding: Fundamentals and Applications. Englewood Cliffs, NJ: Prentice-Hall, 1983.
3. V. Imtawil and D. Tait, "Exploiting the BSD technique in a Multiprocessor Decoder", Proc. of IEEE Symposium on Communication Systems and Digital Signal Processing, vol.1, pp. 133-136, 6-8 April 1998, Sheffield UK.
4. N. Belanger, D. Haccoun and Y. Savaria, "A Multiprocessor Architecture for Multiple Path Stack Sequential Decoders", IEEE Trans. Commun., vol.42, no. 2/3/4, pp. 951-957, Feb/March/Apr 1994.
5. S. Kallel and K. Li, "Bidirectional Sequential Decoding", IEEE Trans. Inform. Theory, vol. 43, no. 4, pp. 1319-1326, July 1997.

On Low-Density Parity-Check Convolutional Codes[1]

Karin Engdahl and Kamil Sh. Zigangirov
Department of Information Technology, Lund University, Sweden
karin@it.lth.se, kamil@it.lth.se

Abstract – In this paper a formal theory, construction methods, and a random ensemble approach for low-density parity-check convolutional codes (including turbo-codes) are presented. The principles of iterative decoding of low-density parity-check (LDPC) convolutional codes are given, and an iterative algorithm for decoding of homogeneous LDPC convolutional codes is described. Some simulation results and performance bounds are also presented.

1. INTRODUCTION

Low-density parity-check (LDPC) block codes were introduced by Gallager in the early 60's [1]. A generalization of Gallager's codes to convolutional codes with a low-density parity-check matrix was developed in [2,3,4]. These codes form a large family of convolutional codes which includes, in particular, the well known turbo-codes [5].

In this paper we give a mathematical definition of these codes and describe a way to construct such codes. We also present bounds on the free distance and on the error probability for such codes. All these bounds have been derived through the analysis of a special ensemble of LDPC convolutional codes, having Markov properties. The error performance described by these bounds can be reached by applying maximum likelihood decoding on these codes, which is much too complex to use for codes with practically interesting sizes.

Iterative decoding methods, first introduced by Elias [6], have attracted great interest during resent years when applied to turbo-codes [5]. Since the complexity of iterative decoding practically does not depend on the memory of the code, it can be used for decoding very long codes, and very good performance can be achieved even for transmission close to the Shannon limit. Using a graph approach [1,7], we describe and analyze an iterative algorithm for decoding of homogeneous LDPC convolutional codes. For simplicity we consider only the case when the code rate $R = 1/2$. In the end we present the results of computer simulations of the algorithm for different LDPC convolutional codes, and compare these results with the theoretical results for maximum likelihood decoding.

[1]This work was supported in part by Swedish Research Council for Engineering Sciences under Grant 95-164.

2. DEFINITION OF LDPC CONVOLUTIONAL CODES

Let

$$\mathbf{u} = \ldots, \mathbf{u}_{-1}, \mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_t, \ldots, \qquad \mathbf{u}_t = u_{tb}, \ldots, u_{(t+1)b-1}, \tag{1}$$

and

$$\mathbf{v} = \ldots, \mathbf{v}_{-1}, \mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_t, \ldots, \qquad \mathbf{v}_t = v_{tc}, \ldots, v_{(t+1)c-1}, \tag{2}$$

$u_t, v_t \in \mathbf{GF}(2)$, $t \in \mathbf{Z}$, be the information and code sequences respectively of an, in the general case time-varying, rate $R = b/c$, $b < c$, convolutional encoder. Then the code sub-blocks $\mathbf{v}_t$ can be defined recursively by the syndrom former matrix of the code [8],

$$\mathbf{v}_t H_0^T(t) + \mathbf{v}_{t-1} H_1^T(t) + \ldots + \mathbf{v}_{t-m_s} H_{m_s}^T(t) = 0 \tag{3}$$

where m_s is the syndrome former memory, $H_i^T(t)$, $i = 0, 1, \ldots, m_s$, are $c \times (c-b)$ sub-matrices of the infinite syndrom former

$$H^T = \begin{pmatrix} \ddots & \ddots & & \ddots & & & \\ \cdots & H_0^T(-1) & H_1^T(0) & \cdots & H_{m_s}^T(m_s-1) & \cdots & \\ & \cdots & H_0^T(0) & H_1^T(1) & \cdots & H_{m_s}^T(m_s) & \cdots \\ & & & \ddots & \ddots & & \ddots \end{pmatrix}, \tag{4}$$

and where the superscript T indicates transposition. We assume that $H_0^T(t)$ has full rank and that $H_{m_s}^T(t) \neq 0$ for all $t \in \mathbf{Z}$. Furthermore, we assume, without loss of generality, that the last $c-b$ rows of $H_0^T(t)$ are linearly independent. In principle, the condition $H_{m_s}^T(t) \neq 0$ for all $t \in \mathbf{Z}$ can be omitted and the syndrom former memory can be undefined, but in this case some of the following definitions must be changed.

As usual, the code sequence should satisfy the equality $\mathbf{v}H^T = \mathbf{0}$. The rows of the infinite syndrom former can be written as infinite binary vectors $\mathbf{h}_i$, $i \in \mathbf{Z}$.

Definition 1 The rate b/c convolutional code defined by its infinite syndrom former H^T of memory m_s is called a *low-density parity-check (LDPC) convolutional code* if the row vectors $\mathbf{h}_i$ are sparse for all $i \in \mathbf{Z}$, that is if

$$w_H(\mathbf{h}_i) \ll (c-b)\, m_s, \qquad i \in \mathbf{Z}, \tag{5}$$

where $w_H(\cdot)$ denotes Hamming weight. If all rows of the syndrom former have constant Hamming weight, and if the same applies for all columns, then the LDPC convolutional code is called *homogeneous*. □

A constructive procedure for designing homogeneous periodically time-varying LDPC convolutional codes was described in [2]. In this paper we present another and more general method of construction of a large family of LDPC convolutional codes (turbo-codes), that includes homogeneous codes as a partial case.

3. CONVOLUTIONAL SCRAMBLERS

Definition 2 An infinite matrix $S = (s_{i,j})$, $i, j \in \mathbf{Z}$, $s_{i,j} \in \{0, 1\}$, that has one one in each row, one one in each column, and satisfies the causality condition $s_{i,j} = 0$, $i > j$ is called a *(single) convolutional scrambler.* □

If $\mathbf{x}$ denotes the infinite input sequence of a convolutional scrambler and $\mathbf{y} = \mathbf{x}S$ is the corresponding output sequence, then $\mathbf{y}$ is a permutation of the symbols in $\mathbf{x}$. The *identity scrambler* has ones only along the diagonal, i.e. $s_{i,i} = 1$, $i \in \mathbf{Z}$. For the *delay scrambler* with delay δ we have $s_{i,i+\delta} = 1$, $i \in \mathbf{Z}$. If $s_{i,j} = s_{i+T,j+T}$, $i, j \in \mathbf{Z}$, for some T, the scrambler is *periodical* with period T. Most practical scramblers are periodical. Particularly the standard $n \times n$ block interleaver can be represented as a periodical convolutional scrambler with period n^2.

Definition 3 An infinite matrix $S = (s_{i,j})$, $i, j \in \mathbf{Z}$, $s_{i,j} \in \{0, 1\}$, that has one one in each column and at least one one in each row, and that satisfies the causality condition is called a *multiple convolutional scrambler.* □

Multiple convolutional scramblers do not only permute the input symbols, but they also make copies of them. As a natural generalization of the multiple convolutional scrambler, consider the convolutional scrambler $S = (S_{k,l})$, $k, l \in \mathbf{Z}$, whose entries $S_{k,l}$ are $c \times d$, $d \geq c$, sub-matrices, satisfying the causality condition $S_{k,l} = 0$, $k > l$. Represented in element form, $S = (s_{i,j})$, each column of this scrambler has exactly one one, and the rows will have d/c ones on average. This scrambler maps input sequences consisting of c-tuples onto output sequences consisting of d-tuples. The ratio d/c is called the *rate* of the scrambler. If all rows have the same number of ones, then the scrambler is called *homogeneous.*

Definition 4 Let Θ be the set of sub-matrices of S such that $S_{k,l} \neq 0$. Then the *delay* δ of the scrambler is defined as

$$\delta = \max_{k,l:S_{k,l} \in \Theta} (l - k) \tag{6}$$

and the *size* Σ of the scrambler is defined as

$$\Sigma = \max_{p} \sum_{k<p} \sum_{l \geq p} w_H\left(S_{k,l}\right) \tag{7}$$

where $w_H\left(S_{k,l}\right)$ denotes the Hamming weight (the number of ones) of the sub-matrix $S_{k,l}$. □

The size of the scrambler is equal to the maximal number of input symbols that the scrambler must keep in its memory. If the diagonal sub-matrices, $S_{k,k}$, of the homogeneous scrambler all have the same Hamming weight, then the scrambler always keeps the same number of input symbols in its memory. Although non-periodical scramblers with infinite delay can be considered in theory, all practical scramblers have finite delays.

Now we will define two operations - *column interleaving* of matrices and *row-column interleaving* of matrices - that can be used to construct multiple convolutional scramblers from more simple scramblers.

Definition 5 The column interleaving $S = S^{(1)} \boxplus S^{(2)}$ of two infinite matrices $S^{(1)} = (S^{(1)}_{k,l})$ and $S^{(2)} = (S^{(2)}_{k,l})$, where $S^{(i)}_{k,l}$, $i = 1, 2$, are $c \times d^{(i)}$ sub-matrices, is an infinite matrix $S = (S_{k,l})$ such that

$$S_{k,2l} = S^{(1)}_{k,l}, \qquad S_{k,2l+1} = S^{(2)}_{k,l}. \tag{8}$$

Since the entries $S_{k,2l}$ and $S_{k,2l+1}$ of the matrix S have sizes $c \times d^{(1)}$ and $c \times d^{(2)}$ respectively, we can merge these two entries into one entry of size $c \times \left(d^{(1)} + d^{(2)}\right)$. □

The column interleaving of two scramblers of rates $d^{(1)}/c$ and $d^{(2)}/c$ gives a rate $\left(d^{(1)} + d^{(2)}\right)/c$ scrambler.

Definition 6 The row-column interleaving $S = S^{(1)} \boxplus S^{(2)}$ of two infinite matrices $S^{(1)} = (S^{(1)}_{k,l})$ and $S^{(2)} = (S^{(2)}_{k,l})$, where $S^{(i)}_{k,l}$, $i = 1, 2$, are $c^{(i)} \times d^{(i)}$ sub-matrices, is an infinite matrix $S = (S_{k,l})$ such that

$$\begin{aligned} S_{2k,2l} &= S^{(1)}_{k,l}, & S_{2k,2l+1} &= 0_{\left(c^{(1)} \times d^{(2)}\right)}, \\ S_{2k+1,2l} &= 0_{\left(c^{(2)} \times d^{(1)}\right)}, & S_{2k+1,2l+1} &= S^{(2)}_{k,l}. \end{aligned} \tag{9}$$

Here $0_{(c \times d)}$ means the all zero $c \times d$ matrix. If we merge these four sub-matrices, we can regard S as a matrix whose sub-matrices are of size $\left(c^{(1)} + c^{(2)}\right) \times \left(d^{(1)} + d^{(2)}\right)$. □

The row-column interleaving of two scramblers of rates $d^{(1)}/c^{(1)}$ and $d^{(2)}/c^{(2)}$ gives a rate $\left(d^{(1)} + d^{(2)}\right) / \left(c^{(1)} + c^{(2)}\right)$ scrambler. The generalization of Definitions 5 and 6 to more than two matrices is straight forward.

4. TURBO-CODES

The following definition of turbo-codes, that differs from conventional definitions, includes a larger family of LDC codes than the usual definitions.

Definition 7 An LDPC convolutional code is called a *turbo-code* if its syndrom former can be represented as a product of a multiple convolutional scrambler S and the syndrom former matrix H_b^T of the basic (component) code, $H^T = SH_b^T$. □

If $R_S = d/c$ is the rate of the scrambler, and $R_b = 1-(c-b)/d$ is the rate of the basic code, then $R = 1 - R_S(1 - R_b) = b/c$ is the rate of the turbo-code. It can be shown [8] that the scrambler matrix S of the standard, rate 1/3, turbo-code [5] is a rate 4/3 scrambler, and that it can be represented as the row-column interleaving of three scramblers $S^{(1)}$, $S^{(2)}$ and $S^{(3)}$, where $S^{(1)}$ is the column interleaving of the identity scrambler and a convolutional scrambler, that describes the interleaver used, and $S^{(2)}$ and $S^{(3)}$ are identity scramblers. The basic code is a rate 2/4 convolutional code, whose syndrom former can be constructed by row-column interleaving the syndrom formers of two rate 1/2 convolutional codes.

To construct the syndrom former for rate 1/2 homogeneous LDPC convolutional codes, having ν ones in each row and 2ν ones in each column, $\nu = 2, 3, \ldots$, we proceed as follows. First, using column interleaving, we construct two rate $\nu/1$ scramblers. Then, using row-column interleaving of these scramblers, we create a rate $2\nu/2$ scrambler S. As basic code we can choose the trivial, memory zero, rate $(2\nu-1)/2\nu$ convolutional code with parity-check matrix

$$H_b = \begin{pmatrix} \ddots & \ddots & \ddots & \ddots & & & & & \\ & 1 & 1 & \ldots & 1 & & & & \\ & & & & & 1 & 1 & \ldots & 1 \\ & & & & & & \ddots & \ddots & \ddots & \ddots \end{pmatrix}. \tag{10}$$

The product SH_b^T is the syndrom former of the resulting rate 1/2 homogeneous LDPC convolutional code. A homogeneous LDPC convolutional code with syndrom former memory m_s, ν ones in each row of the syndrom former and $\nu/(1-R)$ ones in each column of the syndrom former is denoted a homogeneous LDPC convolutional $(m_s, \nu, \nu/(1-R))$-code.

5. ENSEMBLE PERFORMANCE

The most difficult problem with LDPC convolutional codes is to find theoretical bounds for their performance. Our approach to this problem consists in introducing a special ensemble of convolutional scramblers, Markov scramblers, for which we can develop a

strong mathematical theory, and get tight bounds for the performance of the code, such as free distance, burst and bit error probabilities for maximum likelihood decoding. We will explain how to define this ensemble in the case of a homogeneous LDPC convolutional $(m_s, \nu, 2\nu)$-code, $\nu = 2, 3, \ldots$

Let $S = (S_{k,l})$ be a rate $2\nu/2$ convolutional scrambler of size Σ, such that

$$S_{k,l} = \begin{pmatrix} s_{2k,2\nu l} & s_{2k,2\nu l+1} & \cdots & s_{2k,2\nu(l+1)-1} \\ s_{2k+1,2\nu l} & s_{2k+1,2\nu l+1} & \cdots & s_{2k+1,2\nu(l+1)-1} \end{pmatrix}. \tag{11}$$

Let the matrix $S_{k,k}$ be such that $s_{2k,2\nu k} = 1$, $s_{2k,2\nu k+p} = 0$, $p = 1, 2, \ldots, 2\nu-1$, $s_{2k+1,2\nu k+\nu} = 1$ and $s_{2k+1,2\nu k+p} = 0$, $p = 0, 1, \ldots, \nu-1, \nu+1, \ldots, 2\nu-1$. The matrices $S_{k,l}$, $k < l$, are chosen randomly from the ensemble of scramblers according to the following rule. Consider an arbitrary row

$$\mathbf{s}_i = \ldots \quad s_{i,0} \quad s_{i,1} \quad \ldots \quad s_{i,j} \quad \ldots \tag{12}$$

of the matrix S. Clearly the Hamming weight of this row is equal to ν, and $s_{i,j} = 0$ if $j < \nu i$. Let

$$\sum_{p=\nu i}^{j} w_H(s_{i,p}), \qquad j \geq \nu(i+1) \tag{13}$$

be the accumulated Hamming weight of $\mathbf{s}_i$ at moment j, and let

$$w_{i,j} = \nu - \sum_{p=\nu i}^{j} w_H(s_{i,p}) \tag{14}$$

be the residual Hamming weight of $\mathbf{s}_i$ at moment j. According to the definition of a scrambler, only one of the jth components $s_{i,j}$, $i \in \mathbf{Z}$, of the vectors $\mathbf{s}_i$ can be nonzero. Since $s_{i,\nu i} = 1$, we have that the nonzero element for column $j = \nu p$, $p \in \mathbf{Z}$, is the one on row $i = p$. For the rest of the columns we have to choose the nonzero jth component among the jth components of the vectors $\mathbf{s}_i$, $i < 2\lfloor j/2\nu \rfloor$, that have nonzero residual Hamming weight $w_{i,j}$. We suppose that for any j a special random mechanism first chooses i, $i < 2\lfloor j/2\nu \rfloor$, with probability

$$P_j(i) = \frac{w_{i,j}}{\sum_{i<2\lfloor j/2\nu \rfloor} w_{i,j}}, \tag{15}$$

and then sets $s_{i,j} = 1$ and $s_{p,j} = 0$, $p \neq i$. This defines an ensemble of Markov scramblers with rate $2\nu/2$. We use the trivial, rate $(2\nu-1)/2\nu$, memory zero convolutional code with parity-check matrix (10) as basic code.

In Figure 1 a bound for the free distance of LDPC convolutional codes, analogous to the Costello bound for "usual" convolutional codes, is given. At least half of the codes

in the ensemble derived above have free distance greater than the curves shown in Figure 1, for different ν. The curve for $\nu = 2.5$ corresponds to the semi-homogeneous LDPC convolutional code, whose syndrom former has three ones in each even row, two ones in each odd row, and five ones in each column. The upper bounds for the burst error probabilities are given in Figure 3, where the union and expurgation bounds are presented.

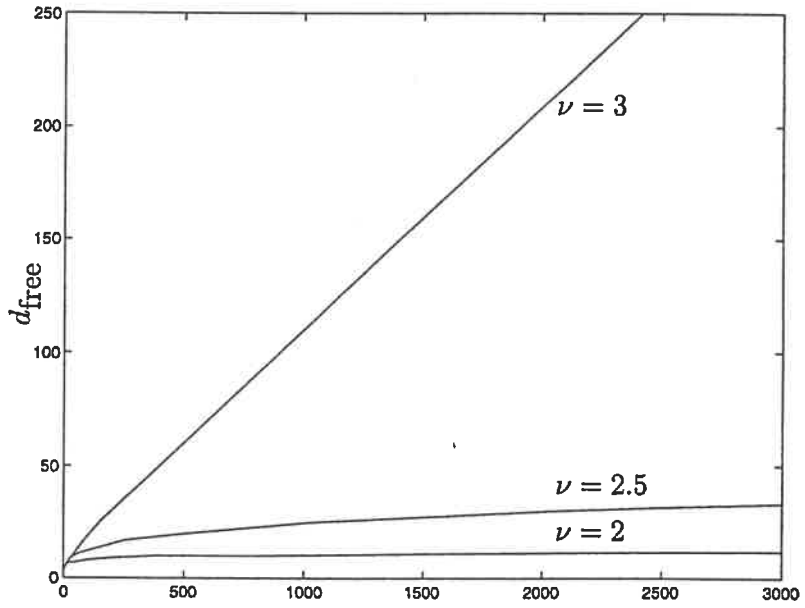


Figure 1: Lower bound on the free distance.

6. PRINCIPLES OF ITERATIVE DECODING

Consider the homogeneous LDPC convolutional $(m_s, \nu, \nu/(1-R))$-code. Suppose that the code sequence

$$\mathbf{v} = v_0, v_1, \ldots, v_i, \ldots, \qquad v_i \in \mathrm{GF}(2), \tag{16}$$

of this code is transmitted, by antipodal signaling, over an AWGN channel with one-sided power spectral density N_0. Let the received sequence be

$$\mathbf{r} = r_0, r_1, \ldots, r_i, \ldots, \tag{17}$$

where r_i are independent Gaussian variables, such that $E(r_i) = \sqrt{E_s}(1 - 2v_i)$ and $Var(r_i) = N_0/2$, where $E_s = E_b R$ is the energy per symbol. Let $\pi_i^{(0)}$, $i = 0, 1, \ldots$ be the apriori probability that $v_i = 0$, i.e. $\pi_i^{(0)} = P(v_i = 0)$, and let $f(r_i \mid v_i)$ be the probability density function of r_i conditioned on that the transmitted symbol is v_i. The

goal of the decoding is to calculate, for all symbols v_i, the aposteriori probabilities (APP) that $v_i = 0$ given that the transmitted code sequence of the convolutional code $\mathcal{C}$ is $\mathbf{v}$ and that the received sequence is $\mathbf{r}$, $\pi_i^{apost} = P(v_i = 0 \mid \mathbf{v} \in \mathcal{C}, \mathbf{r})$. The calculation of the sequence of aposteriori probabilities π_i^{apost}, $i = 0, 1, \ldots$, for each symbol v_i in the code sequence, is called APP or soft decoding. Application of the trivial decision rule

$$\hat{v}_i = \begin{cases} 0, & \text{if } \pi_i^{apost} > 1/2 \\ 1, & \text{otherwise,} \end{cases} \tag{18}$$

yields maximum aposteriori probability (MAP) or hard decoding. Unfortunately, the complexity of such decoding, if all aposteriori probabilities should be calculated correctly, exceeds the computational possibilities of a realistic decoder. Therefore, for each i, the decoder calculates, not the statistic π_i^{apost}, but an approximation of it, $\tilde{\pi}_i^{apost}$. The calculation of $\tilde{\pi}_i^{apost}$ is iterative, such that each new value defines π_i^{apost} more accurately.

Definition 8 Let $\pi_i^{(k)} = \tilde{\pi}_i^{apost}$, calculated in the kth iteration step satisfy Bayes formula

$$\pi_i^{(k)}\left(\{\mathcal{R}_{i,1}\}, \ldots \{\mathcal{R}_{i,\nu}\}\right) = \frac{\pi_i^{(k-1)}\left(\{\mathcal{R}_{i,1}\}, \ldots, \{\mathcal{R}_{i,\nu-1}\}\right) P\left(\{\mathcal{R}_{i,\nu}\} \mid v_i = 0\right)}{P\left(\{\mathcal{R}_{i,\nu}\}\right)}, \tag{19}$$

where $\{\mathcal{R}_{i,1}\}, \ldots \{\mathcal{R}_{i,\nu}\}$ are different sub-sets of received symbols. The probability $\pi_i^{(k)}$ is called *correct*, if $\cap_{k=1}^{\nu} \{\mathcal{R}_{i,k}\} = \emptyset$. □

We will show that it is possible to organize the iterative procedure such that, at least in the first iteration steps, the calculated statistics are correct, and the condition of correctness is violated only in later iteration steps.

Below, we consider decoding of homogeneous LDPC convolutional $(m_s, \nu, \nu/(1-R))$-codes. Then each symbol enters into ν parity-check equations and each equation contain $\nu/(1-R)$ symbols.

Example 1 Let us consider the decoding of a rate $R = 1/3$ homogeneous LDPC convolutional code with $\nu = 2$. Each symbol is included in two parity-check equations, and each equation contains three symbols. We can represent this code as a set of triangles, Figure 2, where each triangle corresponds to a parity-check equation, and each apex corresponds to a symbol v_i. We enumerate the apexes i_k, $k = 0, 1, \ldots$, and the triangles j_l, $l = 0, 1, \ldots$, such that each apex is part of two triangles, one with even number and one with odd. Let the apriori probabilities $\pi_{i_k}^{(0)}$ and the received sequence $\mathbf{r}_{i_k}$ be known. In the first iteration step we calculate two aposteriori probabilities, $\pi_{i_0,j_0}^{(1)}$ and $\pi_{i_0,j_1}^{(1)}$, for symbol v_{i_0}. The one

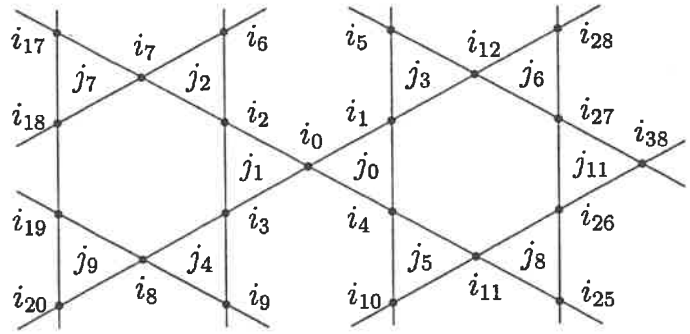


Figure 2: A graph describing the $(m_s, 2, 3)$-code in Example 1.

corresponding to a parity-check equation with *even* number is calculated by using the symbols in the parity-check equation with *odd* number, i.e.

$$\pi_{i_0,\text{even}}^{(1)} = \pi_{i_0,j_0}^{(1)} = P\left(v_{i_0} = 0 \mid r_{i_0}, r_{i_2}, r_{i_3}\right), \tag{20}$$

and vice versa,

$$\pi_{i_0,\text{odd}}^{(1)} = \pi_{i_0,j_1}^{(1)} = P\left(v_{i_0} = 0 \mid r_{i_0}, r_{i_1}, r_{i_4}\right). \tag{21}$$

Obviously, we use the apriori probabilities $\pi_{i_0}^{(0)}$, $\pi_{i_2}^{(0)}$ and $\pi_{i_3}^{(0)}$ in the calculation of $\pi_{i_0,\text{even}}^{(1)}$, and $\pi_{i_0}^{(0)}$, $\pi_{i_1}^{(0)}$ and $\pi_{i_4}^{(0)}$ in the calculation of $\pi_{i_0,\text{odd}}^{(1)}$. Analogous calculations are carried out for all symbols v_{i_k}, $k = 0, 1, \ldots$. Then, as result of the first iteration we get, for each symbol v_{i_k}, one even and one odd aposteriori probability, each corresponding to a parity-check equation that includes v_{i_k}. These statistics will be used as apriori probabilities in the second iteration step. Continuing the iteration process, we get, after the nth step, the aposteriori probabilities $\pi_{i_k,\text{even}}^{(n)}$ and $\pi_{i_k,\text{odd}}^{(n)}$ for each symbol v_{i_k}. These statistics will, together with $\pi_{i_k}^{(0)}$, be used in the $(n+1)$th iteration step. For the description of the decoding process we need the following additional definitions, here given for symbol v_{i_0} but the generalization to v_{i_k}, $k = 1, 2, \ldots$ is trivial.

$$F_{i_0} \stackrel{def}{=} \frac{f\left(r_{i_0} \mid v_{i_0} = 0\right)}{f\left(r_{i_0} \mid v_{i_0} = 1\right)} = \exp\left(\frac{4\sqrt{E_s} r_{i_0}}{N_0}\right), \tag{22}$$

$$\Lambda_{i_0}^{(0)} \stackrel{def}{=} \frac{L_{i_0}^{(0)} F_{i_0} - 1}{L_{i_0}^{(0)} F_{i_0} + 1} = \tanh\left(\frac{1}{2} \ln L_{i_0}^{(0)}\right), \qquad L_{i_0}^{(0)} \stackrel{def}{=} \frac{\pi_{i_0}^{(0)}}{1 - \pi_{i_0}^{(0)}} \tag{23}$$

Following Gallager [1], we get

$$L_{i_0,\text{odd}}^{(1)} = L_{i_0}^{(0)} F_{i_0} \frac{1 + \Lambda_{i_1}^{(0)} \Lambda_{i_4}^{(0)}}{1 - \Lambda_{i_1}^{(0)} \Lambda_{i_4}^{(0)}}, \tag{24}$$

and $L^{(1)}_{i_0,\text{even}}$ is calculated analogously. In the following iteration steps we have

$$\Lambda^{(n)}_{i_0,\text{odd}} \stackrel{def}{=} \frac{L^{(n)}_{i_0,\text{odd}} - 1}{L^{(n)}_{i_0,\text{odd}} + 1} = \tanh\left(\frac{1}{2}\ln L^{(n)}_{i_0,\text{odd}}\right), \qquad n = 1, 2, \ldots \tag{25}$$

where

$$L^{(n)}_{i_0,\text{odd}} \stackrel{def}{=} \frac{\pi^{(n)}_{i_0,\text{odd}}}{1 - \pi^{(n)}_{i_0,\text{odd}}} = L^{(0)}_{i_0} F_{i_0} \frac{1 + \Lambda^{(n-1)}_{i_1,\text{even}} \Lambda^{(n-1)}_{i_4,\text{even}}}{1 - \Lambda^{(n-1)}_{i_1,\text{even}} \Lambda^{(n-1)}_{i_4,\text{even}}}. \tag{26}$$

In the same way, the likelihood ratio $L^{(n)}_{i_0,\text{even}}$ depends on the statistics $\Lambda^{(n-1)}_{i_2}$ and $\Lambda^{(n-1)}_{i_3}$ calculated in the previous iteration step, and on the original statistic $\Lambda^{(0)}_{i_0}$. It follows from Figure 2 that the likelihood ratio $L^{(1)}_{i_0,\text{odd}}$ depends on $\Lambda^{(0)}_{i_0}$, $\Lambda^{(0)}_{i_1}$ and $\Lambda^{(0)}_{i_4}$, and that the likelihood ratio $L^{(2)}_{i_0,\text{odd}}$ depends on $\Lambda^{(0)}_{i_0}$, $\Lambda^{(0)}_{i_1}$, $\Lambda^{(0)}_{i_4}$, $\Lambda^{(0)}_{i_5}$, $\Lambda^{(0)}_{i_{12}}$, $\Lambda^{(0)}_{i_{10}}$ and $\Lambda^{(0)}_{i_{11}}$. We can also see in Figure 2 that the condition for correct calculation of the odd aposteriori probability is violated in the fourth iteration step. In fact, both the statistics $\Lambda^{(3)}_{i_1}$ and $\Lambda^{(3)}_{i_4}$ depend, in particular, on $\Lambda^{(0)}_{i_{26}}$, $\Lambda^{(0)}_{i_{27}}$ and $\Lambda^{(0)}_{i_{38}}$. Therefore, when we calculate $L^{(4)}_{i_0,\text{odd}}$, the result depends on these statistics twice, which unwarrantably intensifies their significance. This is a corollary of the fact that the graph describing this LDPC convolutional code has a loop of length 4. As a matter of fact, the introduction of the relatively complicated rule of separate calculation of even and odd aposteriori probabilities, is motivated by the desire to have correct aposteriori probabilities at least in the first iteration steps. For the same reason, we use the original value of the statistic $\Lambda^{(0)}_{i_0}$ when we calculate the likelihood ratio according to formula (26). In practice, when realizing the algorithm, it is convenient to operate with the statistics $\Lambda^{(n)}_{i_0,\text{even}}$ and $\Lambda^{(n)}_{i_0,\text{odd}}$, defined by (25). Then (26) is equivalent to

$$\Lambda^{(n)}_{i_0,\text{odd}} = \frac{\Lambda^{(0)}_{i_0} + \Lambda^{(n-1)}_{i_1,\text{even}} \Lambda^{(n-1)}_{i_4,\text{even}}}{1 + \Lambda^{(0)}_{i_0} \Lambda^{(n-1)}_{i_1,\text{even}} \Lambda^{(n-1)}_{i_4,\text{even}}}, \tag{27}$$

and $\Lambda^{(n)}_{i_0,\text{even}}$ is calculated analogously. In the last, Ith, iteration step the calculation of $\Lambda^{(I)}_{i_0}$ is done by using both parity-check equations that includes v_{i_0}. Namely,

$$\Lambda^{(I)}_{i_0,\text{odd}} = \frac{\Lambda^{(0)}_{i_0} + \Lambda^{(I-1)}_{i_1,\text{even}} \Lambda^{(I-1)}_{i_4,\text{even}}}{1 + \Lambda^{(0)}_{i_0} \Lambda^{(I-1)}_{i_1,\text{even}} \Lambda^{(I-1)}_{i_4,\text{even}}}, \qquad \Lambda^{(I)}_{i_0} = \frac{\Lambda^{(I)}_{i_0,\text{odd}} + \Lambda^{(I-1)}_{i_2,\text{even}} \Lambda^{(I-1)}_{i_3,\text{even}}}{1 + \Lambda^{(I)}_{i_0,\text{odd}} \Lambda^{(I-1)}_{i_2,\text{even}} \Lambda^{(I-1)}_{i_3,\text{even}}}. \tag{28}$$

The rule

$$\hat{v}_{i_0} = \begin{cases} 0, & \text{if } \Lambda^{(I)}_{i_0} > 0 \\ 1, & \text{otherwise,} \end{cases} \tag{29}$$

is used to make a decision about v_{i_0}. □

We have described the iterative decoding process in the simple case, when each symbol enters into two parity-check equations and each equation contains three symbols. Although the graph describing the code can get complicated, the generalization to other cases is straight forward.

We will now consider the decoding algorithm of (semi-) homogeneous LDPC convolutional $(m_s, \nu, 2\nu)$-codes, which are the codes that we used in all simulations and analysis. Let $\mathcal{I}(j)$ be the set of all i, such that v_i enters into the jth parity-check equation, and let $\mathcal{J}(i)$ be the set of parity-check equations j, that include symbol v_i. $\mathcal{I}\mathcal{J}(i)$ is the set of i', such that $v_{i'}$ participates in one of the parity-check equations that include v_i. In the nth iteration step, $n = 1, 2, \ldots, I-1$, we calculate, for each i, ν statistics $\Lambda_{i,j}^{(n)}$. In the calculation of $\Lambda_{i,j}^{(n)}$ we use the statistics $\Lambda_i^{(0)}$ and $\Lambda_{i',j'}^{(n-1)}$ such that $i' \in \mathcal{I}\mathcal{J}(i)$, $i' \neq i$ and $j' \in \mathcal{J}(i)$, $j' \neq j$. Only in the last, Ith, iteration step we calculate the statistic $\Lambda_i^{(I)}$, using $\Lambda_{i',j'}^{(I-1)}$ such that $i' \in \mathcal{I}\mathcal{J}(i)$, $i' \neq i$ and $j' \in \mathcal{J}(i)$, and make a decision according to (29). Clearly, for any finite syndrom former of memory m_s, the condition of correctness of the calculated aposteriori probabilities will be violated in some iteration step. The later it will be violated, the better, and the scrambler design problem is to construct a scrambler having as long loops as possible for a given size Σ.

Definition 9 An iterative aposteriori probability (APP) decoding procedure for LDPC convolutional codes is called *asymptotically correct* if, for any fixed number of iterations I, there exists an LDPC convolutional code with sufficiently large syndrom former memory m_s, such that the calculated APP values are correct. □

The procedure of decoding rate 1/2 LDPC convolutional codes, described above, is asymptotically correct.

7. DECODING ALGORITHM

A formal description of the algorithm for rate $R = 1/2$ homogeneous LDPC convolutional codes is given below. For each $i = 0, 1, \ldots$

- $$\Lambda_i^{(0)} = \frac{L_i^{(0)} F_i - 1}{L_i^{(0)} F_i + 1}, \qquad L_i^{(0)} = \frac{\pi_i^{(0)}}{1 - \pi_i^{(0)}}, \qquad F_i = \exp\left(\frac{4\sqrt{E_s} r_i}{N_0}\right)$$

- For each processor $k = 1, 2, \ldots, I$
  - $i_k = i - 2m_s(k-1)$

– For each $q = 1, 2, \ldots, \nu$ and for each $j_p \in \mathcal{J}(i_k)$, $p = 1, 2, \ldots, \nu$

$$\Lambda_{i_k,j_p}^{(k,q)} = \begin{cases} \frac{\Lambda_{i_k,j_p}^{(k'',q'')} + \prod_{i' \in \mathcal{I}(j_q), i' \neq i_k} \Lambda_{i',j_q}^{(k')}}{1 + \Lambda_{i_k,j_p}^{(k'',q'')} \prod_{i' \in \mathcal{I}(j_q), i' \neq i_k} \Lambda_{i',j_q}^{(k')}} & p \neq q \\ \Lambda_{i_k,j_p}^{(k'',q'')} & p = q \end{cases} \tag{30}$$

where

$$k' = \begin{cases} k-1 & \text{if } i' > i_k \\ k & \text{otherwise} \end{cases} \qquad (k'', q'') = \begin{cases} (k-1, \nu) & \text{if } q = 1 \\ (k, q-1) & \text{otherwise} \end{cases}$$

and

$$\Lambda_{i_k,j_p}^{(k'',(p+1) \bmod \nu)} = \Lambda_{i_k}^{(0)}, \qquad \Lambda_{i_k,j_p}^{(k,\nu)} = \Lambda_{i_k,j_p}^{(k)}.$$

- Make a decision according to

$$\hat{v}_{i_I} = \begin{cases} 0 & \text{if } \Lambda_{i_I}^{(I)} \geq 0 \\ 1 & \text{otherwise,} \end{cases}$$

where

$$\Lambda_{i_I}^{(I)} \stackrel{def}{=} \Lambda_{i_I,j_\nu}^{(I,\nu)} = \frac{\Lambda_{i_I,j_\nu}^{(I,q-1)} + \prod_{i' \in \mathcal{I}(j_\nu), i' \neq i_I} \Lambda_{i',j_\nu}^{(k')}}{1 + \Lambda_{i_I,j_\nu}^{(I,q-1)} \prod_{i' \in \mathcal{I}(j_\nu), i' \neq i_I} \Lambda_{i',j_\nu}^{(k')}} \qquad k' = \begin{cases} I-1 & \text{if } i' > i_I \\ I & \text{otherwise} \end{cases}$$

8. SIMULATION RESULTS

We performed computer simulations of the given algorithm for (semi-) homogeneous LDPC convolutional codes with $\nu = 2, 2.5, 3, 4$ some different syndrom former memory m_s, and $I = 60$. The results are shown in Figure 3. In this figure we have also plotted some error bounds, namely the union and expurgation bounds, for maximum likelihood decoding of the random ensemble of codes that we get if we use the ensemble of Markov scramblers described in Section 5. The expurgation bound is the union bound over the sub-ensemble, that contains the codes with a free distance larger than the bound shown in Figure 1.

9. CONCLUSION

In this paper, we presented a formal theory of LDPC convolutional codes, described a method of their construction and gave preliminary results of a probabilistic analysis of such codes. We discussed iterative decoding of LDPC convolutional codes, and simulated the presented iterative decoding algorithm.

ACKNOWLEDGEMENT

The authors are grateful to R. Johannesson for his help in the formulation of some basic definitions.

REFERENCES

[1] R. G. Gallager, *Low-Density Parity-Check Codes*, M.I.T. Press, Cambridge, Massachusetts, 1963.

[2] A. Jimenez and K. Sh. Zigangirov, "Periodic Time-Varying Convolutional Codes with Low-Density Parity-Check Matrices", submitted to *IEEE Trans. on Inform. Theory.*

[3] A. Jimenez and K. Sh. Zigangirov, "Periodic Time-Varying Convolutional Codes with Low-Density Parity-Check Matrices", *Proceedings ISIT-98*, p.305.

[4] K. Engdahl and K. Sh. Zigangirov, "On the Statistical Theory of Turbo-Codes", *Proceedings ACCT-98*, pp. 108-111.

[5] C. Berrou, A. Glavieux and P. Thitimajshima, "Near Shannon Limit Error Correcting Coding and Decoding: Turbo-Codes", *Proceedings ICC-93*, pp. 1064-1070.

[6] P. Elias, "Error-free coding", *IRE Transactions on Information Theory*, vol. PGIT-4, pp. 29-37, Sept. 1954.

[7] R.M. Tanner, "A Recursive Approach to Low Complexity Codes", *IEEE Transactions on Information Theory*, vol. IT-27, pp. 533-547, Sept. 1981.

[8] R. Johannesson and K. Sh. Zigangirov, *Fundamentals of Convolutional Codes*, to be published by IEEE Press, 1999.

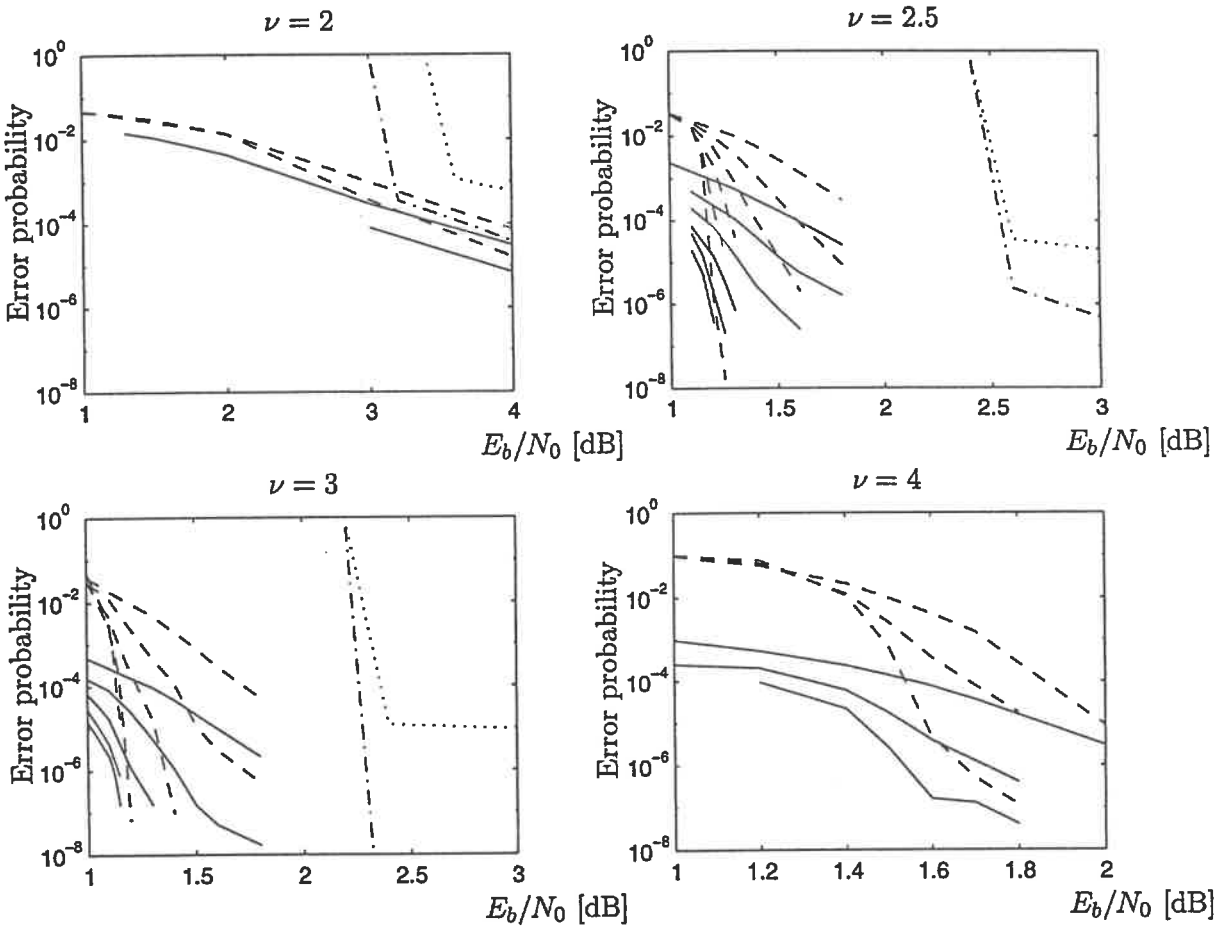


Figure 3: Simulation results and bounds for (semi-) homogeneous LDPC convolutional codes. The dashed lines are simulated bit error probabilities, the solid lines are simulated burst error probabilities, and the bounds described in Section 5 are the dotted (union bound) and dash-dotted (expurgation bound) lines. For all the plots the syndrom former memory m_s are, starting from the top and continuing for as many solid lines as there is in the plot; 129, 257, 513, 1025, 2049 and 4097. The bounds are plotted only for the case when $m_s = 129$.

A Gallager-Tanner construction based on convolutional codes

Sandrine VIALLE, Joseph BOUTROS

Motorola Research Center, Paris
ENST, Communications & Electronics Department
Email : *NAME*@com.enst.fr

December 1, 1998

Keywords : Low density codes, Turbo codes, Asymptotically good codes.

Abstract

Generalized low density codes are built by applying a Tanner-like construction to binary recursive systematic convolutional codes. The Gallager-Tanner construction is restricted to 2 levels only. We describe the structure of a GLD code and show how to compute its ensemble performance. We also prove that RSC based GLD codes are asymptotically good. A parity-check interpretation of turbo codes is given for both parallel and serial concatenations.

1 Introduction

An efficient channel coding scheme has to imitate random codes. To make it feasible, such a scheme is generally based on simple structured elementary codes linked via a pseudo-random interleaver. Low density parity-check (LDPC) codes developed by Gallager [1] are a good example of error-correcting codes imitating random coding.
A binary LDPC code (N, K) of length N and dimension K is defined by a set of $N-K$ interleaved parity-check equations (PCEs) making the $(N-K) \times N$ matrix H. The low density of H is due to the limited number of 1's in each PCE and the limited weight of its columns. It is also proved [1] that the low density of H reduces significantly the complexity of the LDPC iterative decoder. Figure 1 shows an LDPC matrix of size 9×12. This matrix is obtained from the concatenation of $J = 3$ submatrices H_i, $i = 1, 2, 3$. The first submatrix H_1 contains 3 disjoint PCEs and the other ones are given by applying a random column permutation π to H_1, i.e. $H_2 = \pi_1(H_1)$ and $H_3 = \pi_2(H_1)$. The PCEs weight is $n = 4$ and the columns weight is $J = 3$. Gallager's codes are asymptotically good in the sense of the minimum distance criterion if $J \geq 3$.

The matrix representation of an LDPC code is equivalent to a bipartite graph showing the structure of the code. The left part of the graph has N nodes (bit nodes) and the right one has $N-K$ nodes (subcode nodes). The graph is regular and the degrees of the bit nodes and the subcode nodes are J and n respectively. For example, the code of Figure 1 can be graphically represented with a bipartite graph having 12 bit nodes and 9 subcode nodes.
The generalization of the graphical representation described above generates a Tanner code [2]. In fact, each subcode node of an LDPC code is associated to an $(n = 4, k = 3, 2)$ single parity-check (SPC) code. A Tanner code is built from a graph where the subcode nodes are associated to a more general linear (n, k, d_{Hmin}) code, e.g. BCH codes.

In this paper, we apply a Tanner-like construction to binary recursive systematic convolutional (RSC) codes [3] to build a generalized low density (GLD) code. The matrix representation of a GLD code is similar to an LDPC representation where the PCEs derived from the SPC code are replaced by non-disjoint PCEs derived from an RSC code. In the sequel, we restrict the Gallager-Tanner construction to $J = 2$ levels only. We describe the structure of a GLD code and show how to compute its average weight distribution, i.e. its ensemble performance. We also prove that RSC based GLD codes (with 2 levels) are asymptotically good. A parity-check interpretation of turbo codes [4] is given for both parallel and serial concatenations, where turbo codes are described as a special case of Low Density Parity Check codes.

The structure of the generalized low density code is given in section 2 and its performance is analysed in section 3. A Gallager-Tanner interpretation of turbo codes is made in section 4 before sketching the simulation results on a gaussian channel.

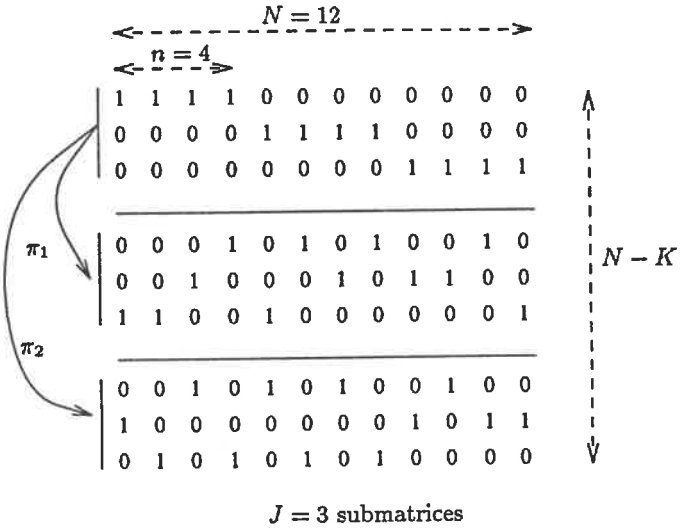


Figure 1: Example of an LDPC matrix H with $J = 3$ levels.

2 The GLD code structure

For simplicity reasons, we consider only RSC codes of rate $r = \frac{k}{k+1}$. The RSC encoder reads k information bits and generates an additional parity bit. The total number of coded bits at its output is denoted by $n = k + 1$. The constraint length is $L = \nu + 1$ and the RSC code trellis has 2^ν states. This convolutional code is defined by n generator polynomials, $g_0(x), g_1(x), \ldots, g_k(x)$. The n output sequences $s_i(x)$ and the k input sequences $e_i(x)$ are related by the following equations

$$s_i(x) = e_i(x) \quad for\ i = 1, \ldots, k \qquad and \quad s_0(x) = \sum_{i=1}^{k} \frac{g_i(x)}{g_0(x)} e_i(x) \tag{1}$$

The PCEs of the RSC code are defined by the syndrome equation easily derived from (1)

$$g_0(x)s_0(x) + g_1(x)s_1(x) + \ldots + g_k(x)s_k(x) = 0 \tag{2}$$

The above equation produces the parity-check matrix H_{RSC} of the convolutional code. As an example, the matrix below is associated to a four-state RSC of rate $r = 1/2$ with generators $g_0 = 7$ and $g_1 = 5$ in octal notation,

$$H_{RSC} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \tag{3}$$

Note that this convolutional code ($g_0 = 7, g_1 = 5$) has been converted to a ($N = 16, K_1 = 6$) linear block code. The trellis termination needs 2 branches ($\lceil \frac{\nu}{k} \rceil$ branches in general) and it occupies the last 4

columns and the last 2 rows of H_{RSC}. The even columns are associated to information bits and the odd columns to parity bits. Note also that the PCEs of a convolutional code are not disjoint.
The weight distribution of an RSC code (viewed as an (N, K_1) block code) is computed using the transfer function method described in [5]. The effect of the trellis termination phase can be neglected, i.e. $\frac{K_1}{N} \approx r$. The number of codewords of weight ℓ is denoted $N_1(\ell)$, $\ell = 0 \ldots N$.

Let C_1 be an (N, K_1, d_1) linear binary block code built from an RSC code. A second (N, K_1, d_1) block code $C_2 = \pi(C_1)$ is constructed by a random interleaving of C_1.

Definition (GLD code with two identical constituents)
A GLD code C is an (N, K, d_{Hmin}) linear block code equal to the intersection of C_1 and C_2.
The above definition is similar to that of GLD construction based on block codes [6] such as primitive, extended or shortened BCH codes. It can be easily shown that $R = \frac{K}{N} = 2r - 1$ when the permutation π is random. The average minimum Hamming distance d_{Hmin} is obtained from the average weight distribution given in theorem 1. The structure of the GLD parity-check matrix is illustrated in Figure 2.

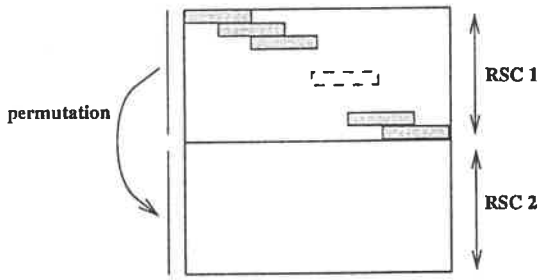


Figure 2: Structure of a GLD parity-check matrix based on two convolutional codes.

The GLD code $C = C_1 \bigcap C_2$ has a simple graphical representation. Each convolutional code C_i is drawn as a chain [7] where a supernode includes the encoder state, the n coded bits and the corresponding channel output (observation). Thus, the Bayesian network [7] of C is obtained by linking the supernodes of the two chains via the interleaver. For example, if $r = 3/4$, each supernode is linked to the opposite code via 4 branches (see Figure 3). The iterative decoding of C is done by propagating the belief in the network (practically by a forward-backward algorithm [8] applied successively on each chain).

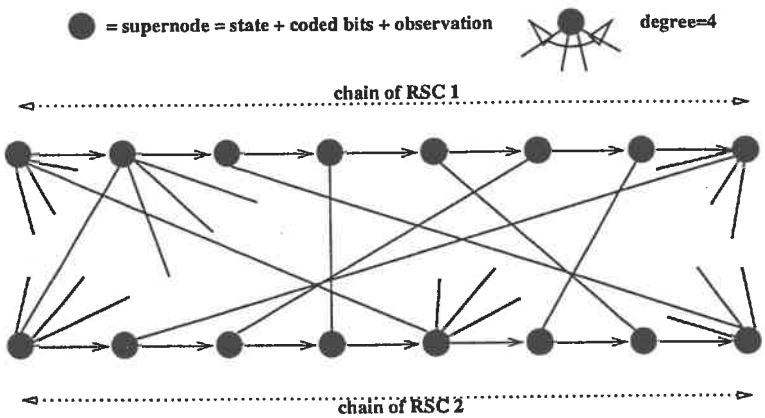


Figure 3: The Bayesian network of a GLD code based on two RSC rate 3/4 codes.

3 Performance analysis of GLD codes

Given a fixed RSC constituent code C_1, the average weight distribution over the whole ensemble of random interleavers is stated by the following theorem.

Theorem 1 (weight distribution)
Let C be an (N, K) GLD code. Then, the average number $N(\ell)$ of codewords of C with weight ℓ is

$$N(\ell) = \frac{N_1(\ell)^2}{\binom{N}{\ell}} \tag{4}$$

where $N_1(\ell)$ is the weight distribution of the constituent RSC code.

Proof. We have $N(\ell) = \binom{N}{\ell} \times P(\ell)$, where $P(\ell)$ is the probability that a weight-ℓ word c chosen at random belongs to C. But if $P_i(\ell)$ is the probability that $c \in C_i$, then $P(\ell) = P_1(\ell)P_2(\ell)$ since $C = C_1 \bigcap C_2$.
By replacing $P_1(\ell) = P_2(\ell) = \frac{N_1(\ell)}{\binom{N}{\ell}}$ we obtain the announced result. *QED.*

Formula (4) has been applied to a rate 1/2 GLD code C based on an 8-state RSC code. The starting tail of the distribution is shown in Table 1. It can be easily shown that $Prob(d_{Hmin} \leq D) \leq \sum_{\ell=d_1}^{D} N(\ell)$. By taking the right hand side of the previous relation equal to 1, we can compute an upper bound for the minimum Hamming distance of C

$$\sum_{\ell=d_1}^{\Delta} N(\ell) = 1 \qquad and \qquad d_{Hmin} \leq \Delta \tag{5}$$

As an example, from Table 1 we obtain $d_{Hmin} \leq 18$.

Weight ℓ	Coefficient $N(\ell)$	Weight ℓ	Coefficient $N(\ell)$
4	3.4E-3	17	0.37
5	1.4E-3	18	0.91
6	8.9E-4	19	2.3
7	6.3E-4	20	6.1
8	1.3E-3	21	16.6
9	2.0E-3	22	46.5
10	3.1E-3	23	134.0
11	4.8E-3	24	396.3
12	8.6E-3	25	1.19E+3
13	1.6E-2	26	3.71E+3
14	3.4E-2	27	1.16E+4
15	7.2E-2	28	3.75E+4
16	0.16	29	1.22E+5

Table 1: The starting tail of the average weight distribution, C_1 is an 8-state (13,7,15,17) RSC.

The output weight distribution is sufficient for computing the bit error probability when Maximum Likelihood (ML) decoding of C. Actually, the interleaver acts on all coded bits, so that they are equally

protected. Thus, the input-output enumerating function [9][10] is not needed to evaluate the ML bound. Finally, we can write the following union bound

$$P_{eb} \leq \sum_{\ell=d_1}^{N} \frac{\ell}{N} \times N(\ell) \times Q(\sqrt{R\ell \frac{2E_b}{N_0}}) \tag{6}$$

where E_b/N_0 is the signal-to-noise ratio per bit and $Q(x)$ is the error function.
Figure 4 shows the average ML bound for two values of the code length, $N = 200$ and $N = 800$, when C_1 has an 8-state trellis.

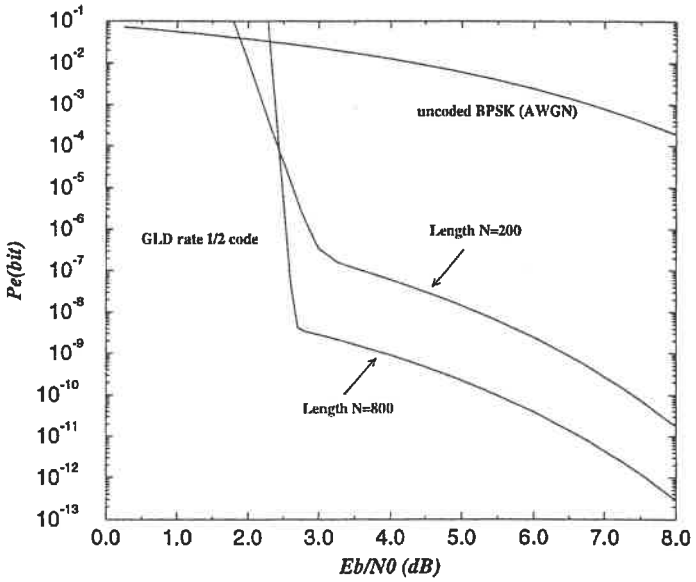


Figure 4: ML Performance of the GLD code built from the 8-state rate 3/4 RSC (13,7,15,17). The GLD code length is $N = 200$ and $N = 800$, total rate $R = 0.5$, on AWGN channel.

Gallager [1] showed that LDPC codes based on simple SPC equations are asymptotically good, i.e. $d_{Hmin} \geq \delta N$ where δ is a positive constant, when $J = 3$ levels. It has been recently proved [6] that Tanner codes based on bipartite graphs and BCH codes are asymptotically good with $J = 2$ levels only. The theorem stated below proves that GLD construction with convolutional codes satisfies the same minimum distance property. Notice that theorems 1 & 2 are not limited to RSC codes and are also valid for non-systematic non-recursive convolutional (NRNSC) codes. In the SISO decoding of a systematic code, the a posteriori probability (APP) depends on the a priori probabilities of information bits and their channel observation. Thus, RSC codes exhibit a slightly better performance than NRNSC codes, when iterative decoding is applied to the whole concatenation.

Theorem 2 (asymptotically good)
The GLD code C built from the rate k/n convolutional code $C_1(N, K_1, d_1)$ is asymptotically good. When N is large enough, the normalized minimum distance $\delta_{min} = d_{Hmin}/N$ is lower bounded by a positive constant δ. The constant δ is the smallest positive (non-zero) root of the equation $B(\lambda) = 0$, where $B(\lambda) > 0$ for $\lambda \in]0 \ldots \delta[$ and

$$B(\lambda) = H(\lambda) - 2\Big(\frac{1}{n} - \frac{\lambda}{n} + \frac{\lambda}{d_1}\Big) H\Big(\frac{\lambda}{\frac{d_1}{n} - \beta\lambda d_1}\Big) \tag{7}$$

$H(x) = -xlog(x) - (1-x)log(1-x)$ is the entropy function and β is a positive constant depending on the convolutional code transfer function.

Proof. We first compute an asymptotic upper bound for $N(\ell)$ from formula (4). When the code length N is large enough, the Stirling approximation gives

$$\frac{1}{\sqrt{2\pi N\lambda(1-\lambda)}} \exp\left(NH(\lambda) - \frac{1}{12N\lambda(1-\lambda)}\right) \leq \binom{N}{\ell} \leq \frac{1}{\sqrt{2\pi N\lambda(1-\lambda)}} \exp(NH(\lambda)) \tag{8}$$

where $\lambda = \frac{\ell}{N}$ is the normalized Hamming weight, $0 \leq \lambda < 0.5$.
By introducing the number $N_1(\ell, e)$ of codewords of C_1 of weight ℓ formed from the concatenation of e consecutive simple error events in the trellis of C_1, we can write the weight distribution of C_1 as a sum over all possible combinations of error events, i.e.

$$N_1(\ell) = \sum_{e=1}^{e_{max}} \binom{\frac{N}{n} - \rho(e,\ell) + e}{e} N_1(\ell, e) \tag{9}$$

where $\rho(e, \ell)$ is the total number of branches in the e error events of total weight ℓ. Moreover $\binom{\frac{N}{n} - \rho(e,\ell) + e}{e}$ is maximal for $e = e_0$, consequently we obtain:

$$N_1(\ell) \leq \binom{\frac{N}{n} - \rho(e_0,\ell) + e_0}{e_0} \sum_{e=1}^{e_{max}} N_1(\ell, e)$$

Using (8), the above equation becomes

$$N_1(\ell) \leq A(N,l) \exp\left(\Big(\frac{N}{n} - \rho(e_0,\ell) + e_0\Big) H\Big(\frac{e_0}{\frac{N}{n} - \rho(e_0,\ell) + e_0}\Big)\right)$$

where $A(N,l) = \frac{1}{\sqrt{2\pi e_0\left(1 - \frac{e_0}{\frac{N}{n} - \rho(e_0,\ell) + e_0}\right)}} \sum_{e=1}^{e_{max}} N_1(\ell, e)$.

Similarly, by substituting equation (9) in (4) we have:

$$N(\ell) \leq C(N,\lambda) \exp(-Nf(\lambda)) \tag{10}$$

where the expression of the exponent function is

$$f(\lambda) = H(\lambda) - 2\Big(\frac{1}{n} - \frac{\rho}{N} + \frac{e_0}{N}\Big) H\Big(\frac{e_0}{\frac{N}{n} - \rho(e_0,\ell) + e_0}\Big)$$

According to inequality (10), $f(\lambda)$ can be lowerbounded while keeping the inequality satisfied. The positive part $\frac{1}{n} - \frac{\rho}{N} + \frac{e_0}{N}$ can be maximized as well $\frac{e_0}{\frac{N}{n} - \rho + e_0}$. We need then to bound ρ and to upperbound e_0.

For a given ℓ, the maximal number of simple error events can be upper bounded by $\frac{\ell}{d_1}$, as d_1 is the minimal weight of an error event. The number of branches ρ is simply lowerbounded by $\rho \geq \frac{\ell}{n}$. On the other hand, we can upperbound ρ by $\beta\ell$, where β is a constant given by the trellis (or state diagram) cycle maximizing the ratio ρ/ℓ for C_1. Figures 5, 6 and 7 illustrate this special cycle for three different codes and the resulting value of β.

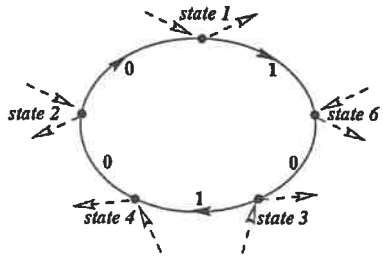


Figure 5: The RSC (13,7,15,17) cycle maximizing ρ/ℓ: cycle length=5, weight=2, $\beta = 5/2$.

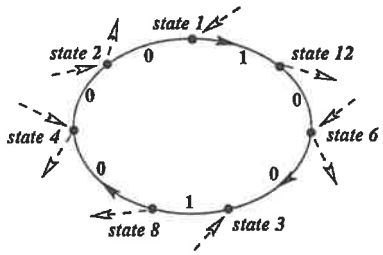


Figure 6: The RSC (23,31,35,37) cycle maximizing ρ/ℓ: cycle length=7, weight=2, $\beta = 7/2$.

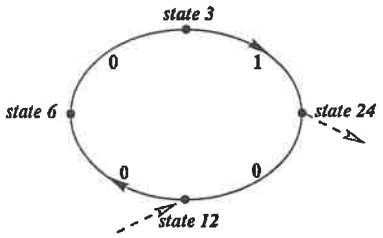


Figure 7: The RSC (45,63,67,75) cycle maximizing ρ/ℓ: cycle length=4, weight=1, $\beta = 4$.

Consequently, we obtain a lower bound for $f(\lambda)$:

$$f(\lambda) \geq H(\lambda) - 2\left(\frac{1}{n} - \frac{\lambda}{n} + \frac{\lambda}{d_1}\right) H\left(\frac{\lambda}{\frac{d_1}{n} - \beta\lambda d_1 + e_0\frac{d_1}{N}}\right) \tag{11}$$

When N is large, $e_0\frac{d_1}{N}$ can be neglected (this has no influence on the starting tail of the weight distribution) and finally we have

$$N(\ell) \leq C(N,\lambda)\exp(-NB(\lambda)) \tag{12}$$

where $B(\lambda)$ is defined by expression (7). Q.E.D.

Expressions of $B(\lambda)$ are given in Table 3 for different constituent RSC codes. $B(\lambda)$ is also sketched on figure 8. The values of δ are listed in the last column of Table 3. Notice that the Gilbert-Varshamov bound produces a minimum distance of $\delta_0 = H_2^{-1}(1-R) = H_2^{-1}(1/2) = 0.11$. We also computed the upperbound Δ from equation (5) for the GLD code based on (13,7,15,17) and for different values of N. Both bounds are shown in Figure (9) where Δ has a linear behavior similar to δN.

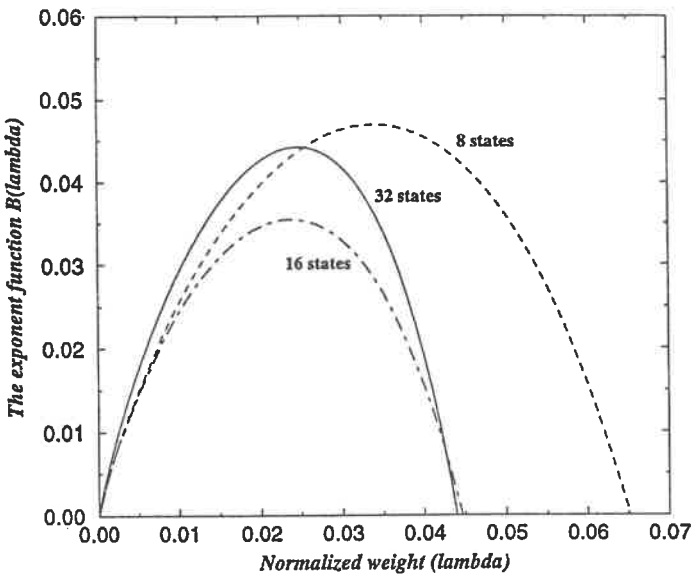


Figure 8: The exponent function $B(\lambda)$ versus the normalized weight λ.

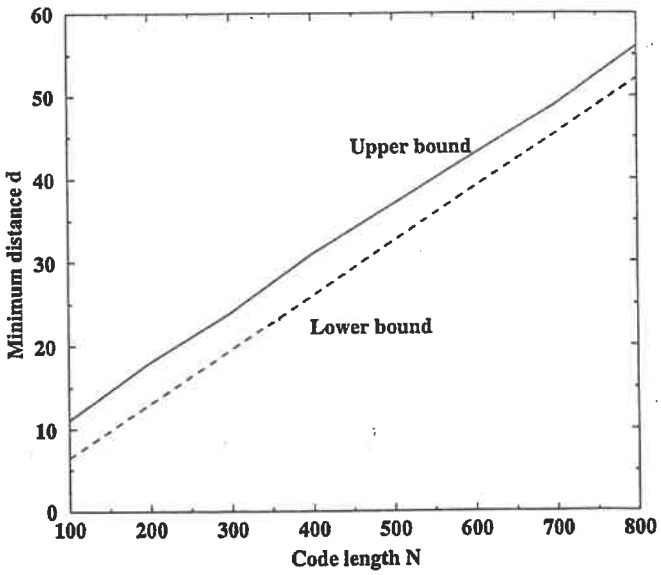


Figure 9: Upper bound and lower bound on the minimum distance.

Number of States	Generator polynomials	$B(\lambda)$	$\delta = \frac{d_{Hmin}}{N}$
8	$13, 7, 15, 17$	$H(\lambda) - \frac{1}{2}H(\frac{\lambda}{1-10\lambda})$	0.0652
16	$23, 31, 35, 37$	$H(\lambda) - \frac{1}{2}H(\frac{\lambda}{1-14\lambda})$	0.0449
32	$45, 63, 67, 75$	$H(\lambda) - \frac{1}{2}(1-\frac{\lambda}{5})H(\frac{\lambda}{\frac{5}{4}-20\lambda})$	0.0439

Table 2: Lower bound on the minimum distance of three GLD codes.

4 A Gallager-Tanner interpretation of PCCCs and SCCCs

Parallel concatenated convolutional codes (PCCCs) [4] and serial concatenated convolutional codes (SCCCs) [11] can be described as the intersection of two (or more) interleaved convolutional codes.

Let us consider a classical (PCCC) turbo code C with an interleaver of size K and two constituents. A parity-check matrix H_1 can be defined for the first constituent $C_1(N_1, K)$. The $(N_1 - K) \times N_1$ matrix H_1 is similar to that given by equation (3), but the columns associated to information bits are now grouped together on the left side. Next, extend C_1 by adding $N_1 - K$ zero columns to H_1 (at the right side). This extended code is denoted C_{1ext}. The parity-check matrix H_{1ext} is written horizontally as 3 blocks : the first block of size $(N_1 - K) \times K$ defines the part of the PCEs associated to information bits, the second block of size $(N_1 - K) \times (N_1 - K)$ corresponds to parity check bits and the last block is null.
The turbo code C is equal to the GLD code obtained from the intersection of C_{1ext} and $C_2 = \pi(C_{1ext})$, where the special interleaver π acts randomly on the K columns of the first block of H_{1ext} and permutes the second and the third blocks.

Let us now consider a serial (SCCC) turbo code C with an interleaver of size N_1 and two constituents. The parity-check matrix H_1 of C_1 (the outer code) is of size $(N_1 - K) \times N_1$. Extend H_1 by adding $N - N_1$ zero columns and denote the extended constituent by C_{1ext}. H_{1ext} has the same structure as described above for the parallel turbo code. The inner code $C_2(N, N_1)$ has a parity-check matrix H_2 with no zero columns, where the N_1 columns associated to information bits are grouped in the left side.
The serial turbo code C is equal to the GLD code obtained from the intersection of C_{1ext} and $\pi(C_2)$, where the special interleaver π acts on the N_1 first columns of H_2.

In both cases, parallel and serial concatenations, the interleaver of a GLD code equivalent to a turbo code does not act on all coded bits. Thus, the formula $R = 2r - 1$ is no more valid. We have $R = r/(2 - r)$ and $R = r_1 r_2$ for PCCCs and SCCCs respectively.
PCCCs exhibit an interleaver gain of $\frac{1}{N}$ when both constituents are RSC codes [9]. SCCCs have an interleaver gain of $\frac{1}{N^{(1+d_1)/2}}$ when the inner code is RSC. Unfortunately, using the proof of theorem 2, it can be easily shown that the bit error probability of a GLD code with a random interleaving of all coded bits is proportionnal to $\sqrt{N}$ at least, i.e. GLD codes do not exhibit an interleaving gain.

On the other hand, the specific structure of the interleaver makes the results on GLD code performance inappropriate for turbo codes. Thus, GLD codes are good for the minimum distance criterion, but it is well known that Turbo codes are not asymptotically good in this sense [9][10][11].

5 Simulation results

The iterative decoding of GLD codes is similar to the SISO decoding of turbo codes. Simulation results presented here are obtained by applying the modified forward-backward algorithm to the constituent code C_1 and its interleaved version. Figures 10, 11, 12, 13 show the bit error rate function of the signal-to-noise ratio E_b/N_0 on a gaussian channel. Two code lengths have been tested, $N = 800$ and $N = 2000$. For each length, two different constituent codes have been compared. The first code C_1 has a rate $r = 3/4$

and is obtained by puncturing the rate 1/2 16 state (23,35) RSC code. The second code C_1 is a true rate 3/4 8 state (13,7,15,17) RSC code.

6 Conclusions

We built generalized low density parity-check codes from the intersection of two randomly interleaved convolutional codes. These codes belong to the Tanner family based on bipartite graphs. It has been proved that such GLD codes are asymptotically good but do not have an interleaving gain. Their average minimum Hamming distance is relatively high, e.g. $52 \leq d_{Hmin} \leq 56$ for a code length $N = 800$ based on an 8 state RSC code. We also showed that parallel and serial turbo codes can be viewed as a special case of GLD codes.

References

[1] R.G. Gallager : Low-density parity-check codes, MIT Press, 1963.

[2] R.M. Tanner : "A recursive approach to low complexity codes," *IEEE Trans. on Information Theory*, Vol. IT-27, Sept 1981.

[3] L.H. Charles Lee : Convolutional coding, fundamentals and applications, Artech House, 1997.

[4] C. Berrou, A. Glavieux, P. Thitimajshima : "Near Shannon limit error-correcting coding and decoding : turbo-codes," *Proceedings of ICC'93*, Genève, pp. 1064-1070, Mai 1993.

[5] D. Divsalar, S. Dolinar, F. Pollara, R.J. McEliece: "Transfer function bounds on the performance of Turbo codes," *TDA Progress Report* 42-122, August 1995.

[6] J. Boutros, O. Pothier, G. Zémor : "Generalized Low Density (Tanner) Codes : Approaching the channel capacity with simple and easily decodable block codes," *ENST - Philips Research Report*, January 1998, also to appear in ICC'99.

[7] B.J. Frey and F.R. Kschischang : "Probability propagation and iterative decoding," *Allerton Conference*, October 1996.

[8] L.R. Bahl, J. Cocke, F. Jelinek, J. Raviv : "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. on Inf. Theory*, vol. 20, pp. 284–287, March 1974.

[9] S. Benedetto, G. Montorsi : "Design of parallel concatenated convolutional codes," *IEEE Trans. Com.*, vol. 44, no. 5, pp. 591-600, May 1996.

[10] L.C. Perez, J. Seghers, D.J. Costello : "A distance spectrum interpretation of turbo codes," *IEEE Trans. on Inf. Theory*, vol. 42, no. 6, pp. 1698-1709, November 1996.

[11] S. Benedetto, G. Montorsi, D. Divsalar, F. Pollara : "Serial concatenation of interleaved codes : Performance analysis, design and iterative decoding," *TDA Progress Report 42-126*, JPL, August 1995.

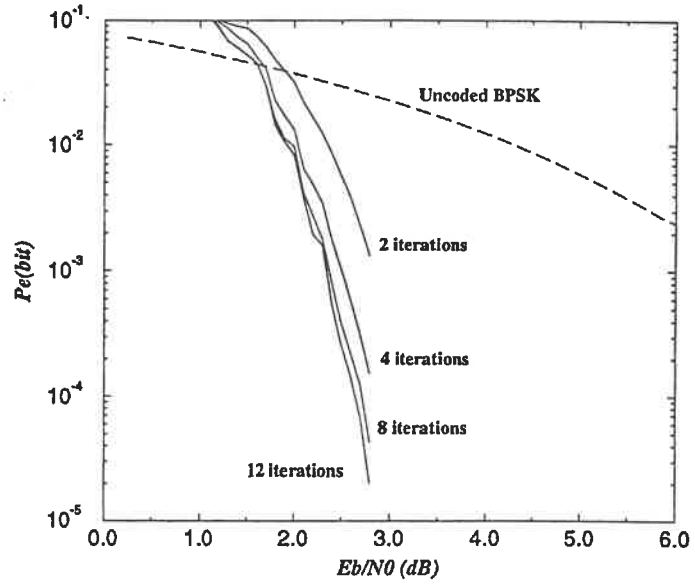


Figure 10: Iterative decoding of a GLD code, $R = 0.5$, C_1 is a 16 state punctured (23,35) RSC, N=800.

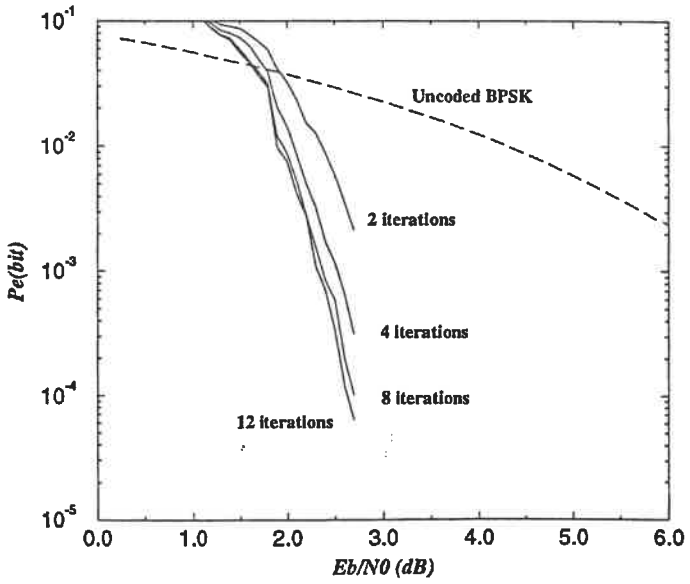


Figure 11: Iterative decoding of a GLD code, $R = 0.5$, C_1 is an 8 state (13,7,15,17) RSC, N=800.

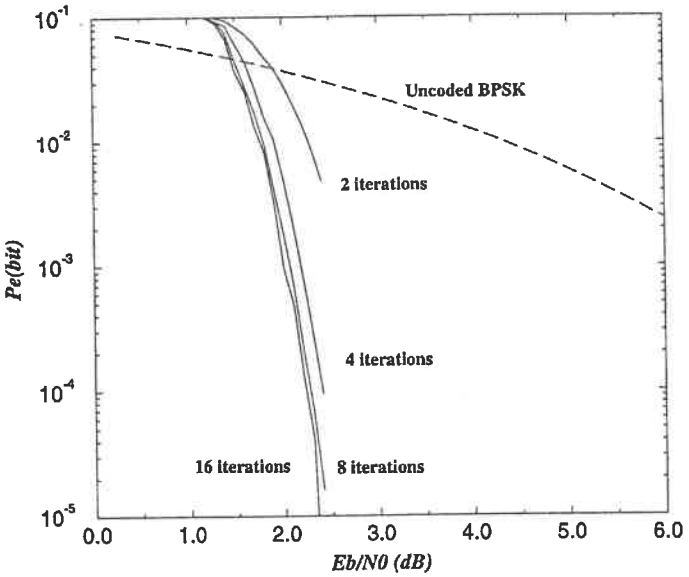


Figure 12: Iterative decoding of a GLD code, $R = 0.5$, C_1 is a 16 state punctured (23,35) RSC, N=2000.

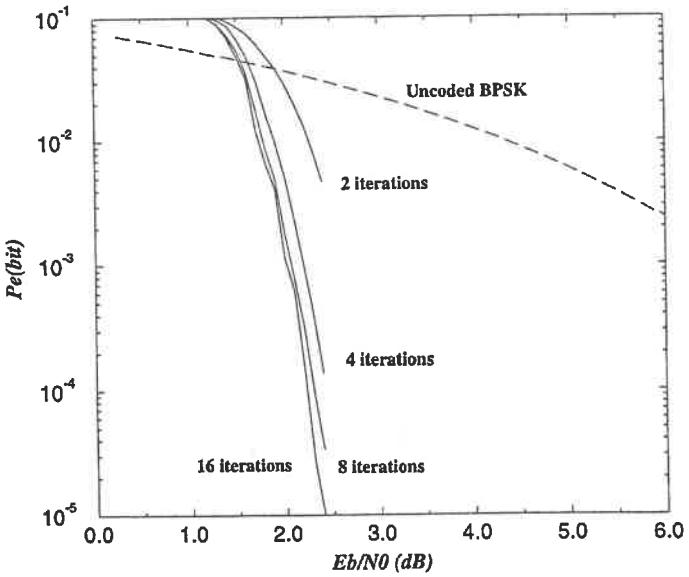


Figure 13: Iterative decoding of a GLD code, $R = 0.5$, C_1 is an 8 state (13,7,15,17) RSC, N=2000.

New Algorithm to Identify Rate k/n Catastrophic Punctured Convolutional Encoders

Conor O'Donoghue and Cyril Burkley,
Department of Electronic Engineering,
University of Limerick, Limerick, Ireland.
E-mail: conor.odonoghue@ul.ie.

Abstract

In this paper we derive a new catastrophicity test for rate k/n punctured convolutional codes. Based on this test we present a computationally efficient algorithm to determine whether or not a punctured convolutional encoder is catastrophic. The algorithm has a simple software implementation and has a broader range of application than previously published algorithms.

I. Introduction

Punctured convolutional codes were introduced by Cain *et al.* [1] as a means of significantly simplifying both Viterbi and sequential decoding of high rate convolutional codes at the expense of a relatively small performance penalty. A punctured convolutional code $\mathcal{C}$ is obtained by periodically deleting output symbols from a convolutional encoder for some low rate $1/n_0$ code which we call the *base code* and denote by $\mathcal{C}_b$. The base code has a canonical generator matrix

$$B(D) = \sum_{i=0}^{\alpha} \boldsymbol{b}_i(D) D^i \qquad \boldsymbol{b}_i \in \mathrm{F}_2^{n_0}$$

where α is the Kronecker index of $\mathcal{C}_b$. Unless stated otherwise $\mathcal{C}_b$ will be assumed to be *antipodal*, that is,

$$\boldsymbol{b}_0 = \boldsymbol{b}_\alpha = (1, 1, \ldots, 1)$$

Symbols from $\mathcal{C}_b$ are deleted according to a periodic puncturing pattern which is described by a puncturing matrix

$$P = \begin{bmatrix} p_{11} & \cdots & p_{1k} \\ \vdots & & \vdots \\ p_{n_0 1} & \cdots & p_{n_0 k} \end{bmatrix} \qquad p_{ij} \in \mathrm{F}_2$$

Consider a sequence of k trellis sections in the trellis diagram of the base code beginning at time $t = 0$. Then $p_{ij} = 0$ indicates that the ith symbol of every branch in the jth trellis section is to be deleted. In this paper we place no restrictions on P except that it has no zero columns. This condition guarantees that at least one symbol is transmitted from each branch in the trellis of $\mathcal{C}$.

Like ordinary convolutional codes, good punctured codes are usually found by computer search [2]-[5]. One of the problems encountered in searching for good punctured codes is that, even if the encoder for the base code is non-catastrophic, the punctured encoder may not be. Such encoders must be identified and eliminated during the search procedure. In view of the large number of codes to be examined it is essential that the algorithm employed be computationally efficient. In Section II we briefly review three algorithms that appear in the literature. In

Section III we show how a canonicity test reported in [6] can be used as a catastrophicity test for rate k/n punctured convolutional encoders. Finally, in Section IV we show how this test can be used as the basis of a fast and simple algorithm to test rate k/n punctured encoders for catastrophicity.

II. Previous Work

Previously published algorithms have for the most part dealt with rate $(n-1)/n$ punctured convolutional codes. In all cases the base code is assumed to be antipodal and the puncturing matrix P is assumed to have no zero columns.

Hole [9] has shown that every rate $(n-1)/n$ punctured encoder can be regarded as a rate $(n-1)/n$ ordinary convolutional encoder. In the same paper he shows how to obtain the corresponding generator matrix $G(D)$. Hence, one method of testing a punctured encoder for catastrophicity, is to compute the greatest common divisor of the full size minors of $G(D)$. The punctured encoder is catastrophic if, and only if, GCD $\neq D^d$ (cf. [10]).

A second method, also due to Hole [8], is based on the fact that the state diagram of a catastrophic encoder contains at least one zero-weight cycle. The search is considerably simplified if the state diagram of the base code is used. The computational complexity of this algorithm is estimated to be $O(n2^{\alpha})$ and the algorithm is easily generalised to rate k/n punctured codes derived from antipodal base codes. Furthermore a software implementation of the algorithm is very straightforward. Its main disadvantage is that the computational complexity increases exponentially with α.

Recently Sun and Vinck [13] have proposed a third algorithm which they claim is more efficient for longer constraint lengths. This algorithm is based on the fact that the dual of a rate $(n-1)/n$ punctured code is a rate $1/n$ code. Hence given any codeword in $\mathcal{C}^{\perp}$ it is easy to find the Kronecker index of the punctured code. If it is less than α then the code is catastrophic. The computational complexity is estimated to be $O(\alpha^3)$. The algorithm is more complex than Hole's algorithm and is restricted to rate $(n-1)/n$ punctured codes derived from antipodal base codes.

The most general of these approaches is the GCD method, the simplest to implement in software is Hole's algorithm, and the most computationally efficient is Sun and Vinck's algorithm. In this paper we derive a new algorithm which combines the benefits of all three approaches.

III. Catastrophicity Test

Let $\mathcal{C}_b$ be a rate $1/n_0$ base code with Kronecker index α and canonical generator matrix $B(D)$ and let $P \in \mathrm{F}_2^{n_0 \times k}$ be any puncturing matrix with n non-zero elements. Let $\mathcal{C}$ denote the corresponding punctured code with Kronecker index $\nu \leq \alpha$. $\mathcal{C}$ has a generator matrix

$$G(D) = \sum_{i=0}^{v_m} G_i D^i \qquad G_i \in \mathrm{F}_2^{k \times n}$$

where v_m denotes the largest constraint length. We propose the following simple method to obtain $G(D)$ by inspection: Form the matrices

$$\hat{G}_i = \begin{bmatrix} b_{ik} & \cdots & b_{(i+1)k-1} \\ \vdots & & \vdots \\ b_{(i-1)k+1} & \cdots & b_{ik} \end{bmatrix} \qquad 1 \leq i \leq v_m \tag{1}$$

The coefficient matrices $\{G_i\}$ are obtained by deleting the columns of the matrices $\{\hat{G}_i\}$ indicated by the zero elements of the puncturing matrix P.

If $\mathcal{C}_b$ is antipodal and P has no zero columns then it is easy to show that $G(D)$ has the following properties:-

(1) $v_m = \lceil \frac{\alpha}{k} \rceil$.

(2) The overall constraint length of $G(D)$ is $v = \alpha$.
(3) The columns of $G(D)$ can be permuted such that $G_0 = [G_{00}|G_{01}]$ where $G_{00} \in \mathrm{F}_2^{k\times k}$ is a nonsingular upper triangular matrix.
(4) The high-order coefficient matrix G_h has full row rank.

Using these properties we can generalise a result for rate $(n-1)/n$ punctured codes due to Hole [7] as follows.

Lemma 1: Let $G(D)$ be the generator matrix for some rate k/n punctured code derived from a rate $1/n_0$ base code with puncturing matrix P. If the base code is antipodal and P has no zero columns then $G(D)$ is canonical if, and only if, $G(D)$ is non-catastrophic. Furthermore $\nu = \alpha$ if, and only if, $G(D)$ is canonical. □

Thus we may determine whether or not $G(D)$ is catastrophic using the following canonicity test [6, Theorem 6].

Theorem 2: Let $G(D)$ be any generator matrix with overall constraint length v. Then $G(D)$ is canonical if, and only if,

$$\operatorname{rank} \mathcal{H}_\circ^{(v)} = (k+1)v$$

where $\mathcal{H}_\circ^{(\ell)}$ is the $k(\ell + v_m) \times \ell n$ matrix

$$\mathcal{H}_\circ^{(\ell)} = \begin{bmatrix} G_0 & 0 & \dots & 0 \\ G_1 & G_0 & & \vdots \\ \vdots & G_1 & \ddots & 0 \\ G_{v_m} & \vdots & \ddots & G_0 \\ 0 & G_{v_m} & & G_1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & & G_{v_m} \end{bmatrix}$$

□

In order to minimise the number of computations required to compute the rank of $\mathcal{H}_\circ^{(v)}$ it is necessary to analyse the rank properties in more detail. To do this we require the following lemma due to Forney [11].

Lemma 3: Let $\mathcal{C}$ be a rate k/n convolutional code with Kronecker indices $\{\nu_i\}_1^k$ and let $\mathcal{C}_\ell$ denote set of polynomial codewords in $\mathcal{C}$ with degree strictly less than ℓ.

$$\mathcal{C}_\ell := \{y(D) \in \mathcal{C} | \deg y(D) < \ell,\ \mathrm{del}\, y(D) \geq 0\}$$

Then $\mathcal{C}_\ell$ is a subspace of $\mathcal{C}$ over F_q with dimension

$$\dim_{\mathrm{F}_q} \mathcal{C}_\ell = k\ell - \nu + \sum_{i:\nu_i \geq \ell} (\nu_i - \ell)$$

□

The rank of the matrix $\mathcal{H}_\circ^{(\ell)}$ is given by the following theorem.

Theorem 4: Let $\mathcal{C}$ be a rate k/n convolutional code and let $G(D)$ be any generator matrix for $\mathcal{C}$ with constraint lengths $\{v_i\}_1^k$. Then

$$\operatorname{rank} \mathcal{H}_\circ^{(\ell)} = k\ell + \nu - \sum_{i:\nu_i^\perp \geq \ell} (\nu_i^\perp - \ell) \tag{2}$$

where $\{\nu_i^\perp\}_1^{n-k}$ are the Kronecker indices of $\mathcal{C}^\perp$. □

Proof: Let $H(D)$ be any canonical polynomial generator matrix for $\mathcal{C}^\perp$. Since $\mathcal{C}$ and $\mathcal{C}^\perp$ are dual subspaces of $\mathrm{F}_q((D))^n$ it follows that

$$H(D)G^T(D) = 0$$

Equivalently we can write $H\bar{G} = 0$ where H and $\bar{G}$ are the semi-infinite block matrices

$$H = \begin{bmatrix} H_0 & H_1 & H_2 & H_3 & \dots \\ 0 & H_0 & H_1 & H_2 & \dots \\ 0 & 0 & H_0 & H_1 & \dots \\ \vdots & \vdots & \vdots & \vdots & \end{bmatrix} \qquad \bar{G} = \begin{bmatrix} G_0^T & G_1^T & G_2^T & G_3^T & \dots \\ 0 & G_0^T & G_1^T & G_2^T & \dots \\ 0 & 0 & G_0^T & G_1^T & \dots \\ \vdots & \vdots & \vdots & \vdots & \end{bmatrix}$$

Now let $\bar{G}^{(\ell)}$ denote the matrix consisting of the first ℓ block rows and $\ell + v_m$ block columns of $\bar{G}$

$$\bar{G}^{(\ell)} = \begin{bmatrix} G_0^T & G_1^T & \dots & G_{v_m}^T & 0 & 0 \\ & \ddots & \ddots & & \ddots & \\ 0 & 0 & G_0^T & G_1^T & \dots & G_{v_m}^T \end{bmatrix} \tag{3}$$

The kernel of $\bar{G}^{(\ell)}$ is spanned by those rows of H that are zero in all but the first $n\ell$ columns. Since $H(D)$ is a canonical polynomial generator matrix these rows are also a basis for $\mathcal{C}_\ell^\perp$, the set of polynomial codewords in $\mathcal{C}^\perp$ with degree less than ℓ. Therefore $\ker \bar{G}^{(\ell)} = \mathcal{C}_\ell^\perp$ and hence from Lemma 3

$$\dim \ker \bar{G}^{(\ell)} = (n-k)\ell - \nu + \sum_{i:\nu_i^\perp \geq \ell} (\nu_i^\perp - \ell) \tag{4}$$

We may also write $\dim \ker \bar{G}^{(\ell)} = n\ell - \operatorname{rank} \bar{G}^{(\ell)}$. Substituting into (4) and re-arranging terms yields

$$\operatorname{rank} \bar{G}^{(\ell)} = k\ell + \nu - \sum_{i:\nu_i^\perp \geq \ell} (\nu_i^\perp - \ell) \tag{5}$$

But inspection of (3) reveals that $\bar{G}^{(\ell)}$ is the transpose of $\mathcal{H}_\circ^{(\ell)}$. Therefore

$$\operatorname{rank} \bar{G}^{(\ell)} = \operatorname{rank} \mathcal{H}_\circ^{(\ell)}$$

Substituting the expression for $\operatorname{rank} \bar{G}^{(\ell)}$ given by (5) yields the desired result. ■

Therefore we can re-state Theorem 2 as follows

Theorem 5: Let $G(D)$ be a generator matrix for $\mathcal{C}$ with overall constraint length v. Then $G(D)$ is canonical if, and only if,

$$\operatorname{rank} \mathcal{H}_\circ^{(\ell)} = k\ell + v \qquad \ell \geq \nu_m^\perp$$

where $\nu_m^\perp$ is defined as the largest Kronecker index of $\mathcal{C}^\perp$. □

We note that $\frac{\nu}{n-k} \leq \nu_m^\perp \leq \nu$ and hence application of Theorem 5 may involve significantly fewer computations than computing the rank of $\mathcal{H}_\circ^{(v)}$. Furthermore if $G(D)$ is catastrophic then $\nu < v$ and hence further savings are made. In the next section we will use Theorem 5 as the basis of a fast algorithm to test $G(D)$ for catastrophicity.

IV. The Algorithm

A computationally efficient algorithm to implement the canonicity test of Theorem 5 can be obtained by exploiting the banded and block Toeplitz structure of the matrix $\mathcal{H}_o^{(\ell)}$. We assume, without loss of generality, that $G_0 = [G_{00}|G_{01}]$ where $G_{00} \in \mathrm{F}_2^{k\times k}$ is a nonsingular upper triangular matrix. If this is not true then the columns of $G(D)$ can be permuted to ensure that this condition is met. Multiplying the block rows of $\mathcal{H}_o^{(\ell)}$ by G_{00}^{-1} and re-arranging columns yields the matrix

$$\hat{\mathcal{H}}_o^{(\ell)} = \left[\begin{array}{ccc|ccc} I & & 0 & S_0 & & 0 \\ R_1 & \ddots & & S_1 & \ddots & \\ \vdots & & I & \vdots & & S_0 \\ R_{v_m} & & R_1 & S_{v_m} & & S_1 \\ & \ddots & \vdots & & \ddots & \vdots \\ 0 & & R_{v_m} & 0 & & S_{v_m} \end{array}\right]$$

where $[R_i \mid S_i] = G_{00}^{-1}G_i$, $0 \le i \le v_m$. Using elementary row operations $\hat{\mathcal{H}}_o^{(\ell)}$ can be put in the form

$$\left[\begin{array}{ccc|ccc} I & & 0 & \bar{S}_0 & & 0 \\ & \ddots & & \vdots & \ddots & \\ 0 & & I & \bar{S}_{\ell-1} & \dots & \bar{S}_0 \\ \hline 0 & \dots & 0 & \bar{S}_\ell & \dots & \bar{S}_1 \\ \vdots & & \vdots & \vdots & & \vdots \\ 0 & \dots & 0 & \bar{S}_{\ell+v_m-1} & \dots & \bar{S}_{v_m} \end{array}\right] \tag{6}$$

where $\bar{S}_i \in \mathrm{F}_2^{k\times(n-k)}$, $0 \le i < \ell + v_m$. It can be shown that the matrices $\{\bar{S}_i\}$ are given by the recursion formula

$$\bar{S}_i = S_i + \sum_{j=0}^{i-1} R_{i-j}\bar{S}_j \tag{7}$$

Note that $R_i = 0$, $\forall\ i > v_m$ and hence the computation of $\bar{S}_i$ requires at most v_m matrix products.

Once the $\{\bar{S}_i\}$ have been computed we form the matrix

$$W^{(\ell)} = [W_1 \quad W_2 \quad \dots \quad W_\ell]$$

where

$$W_i = \begin{bmatrix} \bar{S}_i \\ \vdots \\ \bar{S}_{i+v_m+1} \end{bmatrix} \tag{8}$$

From (6) it is easily seen that $\operatorname{rank} W^{(\ell)} = \operatorname{rank} \mathcal{H}_o^{(\ell)} - k\ell$. Substituting the expression for $\operatorname{rank} \mathcal{H}_o^{(\ell)}$ given by (2) yields

$$\operatorname{rank} W^{(\ell)} = \nu - \sum_{i:\nu_i^\perp \ge \ell} (\nu_i^\perp - \ell)$$

(1) Construct the coefficient matrices $\{G_i\}_0^{v_m}$.
(2) Compute G_{00}^{-1} and the matrices $\{R_i\}_1^{v_m}$ and $\{S_i\}_0^{v_m}$.
(3) FOR $\ell = 1$ TO α
Compute W_ℓ using the recursion formula (7).
Using elementary column operations obtain $W_c^{(\ell)}$ from $W_c^{(\ell-1)}$ and W_ℓ.
IF $(\operatorname{rank} W_c^{(\ell)} - \operatorname{rank} W_c^{(\ell-1)} = 0)$ OR $(\operatorname{rank} W_c^{(\ell)} = v)$ THEN GOTO END
NEXT ℓ
(4) END

TABLE I
ALGORITHM FOR CATASTROPHICITY TEST.

and hence by Lemma 1 and Theorem 5 $G(D)$ is non-catastrophic if, and only if,

$$\operatorname{rank} W^{(\ell)} = v \qquad \ell \geq \nu_m^{\perp}$$

However we have no a priori knowledge of $\nu_m^{\perp}$. This difficulty may be circumvented by noting that

$$\operatorname{rank} W^{(\ell+1)} - \operatorname{rank} W^{(\ell)} \geq 0$$

where the equality holds if, and only if, $\ell \geq \nu_m^{\perp}$. Hence we compute $\operatorname{rank} W^{(\ell)}$ for $\ell = 1, 2, \ldots$ until either (i) $\operatorname{rank} W^{(\ell)} = v$ or (ii) $\operatorname{rank} W^{(\ell)} - \operatorname{rank} W^{(\ell-1)} = 0$. In both cases $G(D)$ is non-catastrophic if, and only if, $\operatorname{rank} W^{(\ell)} = v$.

Finally, we may compute $\operatorname{rank} W^{(\ell)}$, $\ell = 1, 2, \ldots$, as follows. Using elementary column operations put $W^{(\ell)}$ in column echelon form which we denote by $W_c^{(\ell)}$. The rank of $W_c^{(\ell)}$ is determined by inspection. $W_c^{(\ell+1)}$ is easily found from the augmented matrix $[W_c^{(\ell)} \mid W_{\ell+1}]$ where $W_{\ell+1}$ is given by (8). The complete algorithm is summarised in Table I.

A. Computational complexity

Computing the matrices $\{\bar{S}_i\}$ requires $O(k(n-k)\alpha\nu_m^{\perp})$ binary operations. Computation of the rank of $W^{(\nu_m^{\perp})}$ requires $O(\nu_m^{\perp}(n-k)\alpha^2)$ binary operations. Typically α will be greater than k in order to ensure that the punctured code has good distance properties and consequently this latter step dominates the overall computational complexity of the algorithm. For rate $(n-1)/n$ punctured codes the computational complexity is $O(\nu\alpha^2)$ which is the same as Sun and Vinck's algorithm when $G(D)$ is non-catastrophic.

B. Interpretation of the algorithm

Let $G(D)$ be a generator matrix for some rate k/n code $\mathcal{C}$ with Kronecker index ν. We assume, without loss of generality, that $G(D) = [G_0(D) \mid G_1(D)]$ where $G_0(D) \in \mathrm{F}_2[D]^{k\times k}$ is nonsingular and $\det G_0(D)$ is delay-free. Then the systematic generator matrix

$$G_s(D) = G_0^{-1}(D)G(D) = [\ I \mid S(D)\ ]$$

is also a generator matrix for $\mathcal{C}$ [10]. It can be shown that

$$S(D) = \sum_{i=0}^{\infty} \bar{S}_i D^i$$

where the coefficient matrices $\{\bar{S}_i\}$ are given by (7) and hence the first part of the algorithm is equivalent to computing the coefficient matrices of D^i in an equivalent systematic generator matrix.

It is well known [10], [12] that a systematic generator matrix for $\mathcal{C}$ is minimal and hence has a ν-dimensional state space realisation. It is also well known [14] that the McMillan degree of a transfer function $S(D)$ is equal to the rank of the Hankel matrix

$$S = \begin{bmatrix} \bar{S}_1 & \bar{S}_2 & \cdots & \bar{S}_v \\ \bar{S}_2 & \bar{S}_3 & \cdots & \bar{S}_{v+1} \\ \vdots & \vdots & & \ddots \\ \bar{S}_v & \bar{S}_{v+1} & \cdots & \bar{S}_{2v-1} \end{bmatrix}$$

It follows that $\operatorname{rank} S = \nu$. If $\mathcal{C}$ is punctured code derived from a base code with Kronecker index α then by Lemma 1 $G(D)$ is non-catastrophic if, and only if, $\operatorname{rank} S = \alpha$.

V. Conclusions

We have presented a new algorithm for identifying rate k/n catastrophic punctured convolutional encoders. The algorithm is simple to implement in software and requires no polynomial operations. The computational complexity is $O(\nu_m^{\perp}(n-k)\alpha^2)$ which is considerably better than Hole's algorithm when α is large. For rate $(n-1)/n$ punctured codes the computational complexity is the same as Sun and Vinck's algorithm. However their algorithm does not generalise to rate k/n codes.

As with other algorithms reported in the literature, we have assumed punctured codes derived from antipodal base codes. However Bocharova and Kudryashov [5] have recently shown to be false the hypothesis that the best punctured codes are derived from antipodal base codes. A significant advantage, therefore, of the algorithm presented in this paper is that it easily generalised to non-antipodal base codes. In this case the algorithm identifies non-catastrophic punctured codes with Kronecker index equal to that of the base code.

Finally, our algorithm may also used for codes defined over an arbitrary field F_q. With suitable modifications it can also be applied to punctured codes derived from base codes with rational generator matrices.

References

[1] J.B. Cain, G.C. Clark, Jr., and J.M Geist, "Punctured convolutional codes of rate $(n-1)/n$ and simplified maximum likelihood decoding," *IEEE Trans. Inform. Theory*, vol. IT-25, No. 1, pp. 97-100, January 1979.

[2] Y. Yasuda, K. Kashiki, and Y. Hirata, "High-rate punctured convolutional codes for soft decision Viterbi decoding," *IEEE Trans. Commun.*, vol. COM-32, No.2 , pp. 315-319, March 1984.

[3] D. Haccoun, and G. Begin, "High-rate punctured convolutional codes for Viterbi and sequential decoding," *IEEE Trans. on Commun.*, vol. COM-37, No. 11, pp. 1113-1125, November 1989.

[4] M.-G. Kim, "On systematic punctured convolutional codes," *IEEE Trans. on Commun.*, vol. COM-45, No. 2, pp. 133-139, February 1997.

[5] I.E. Bocharova, and B.D. Kudryashov, "Rational rate punctured convolutional codes for soft-decision Viterbi decoding," *IEEE Trans. Inform. Theory*, vol. IT-43, No. 4, pp. 1305-1313, July 1997.

[6] C. O'Donoghue, and C.J. Burkley, "Minimality and canonicity tests for rational generator matrices for convolutional codes," in *Proc. 1998 IEEE Information Theory Workshop*, pp. 112-114, Killarney, 22-26 June, 1998.

[7] K.J. Hole, "Rate $k/(k+1)$ minimal punctured convolutional encoders," *IEEE Trans. Inform. Theory*, vol. IT-37, No. 3, pp. 653-655, May 1991.

[8] K.J. Hole, "An algorithm for determining if a rate (n-1)/n punctured convolutional encoder is catastrophic," *IEEE Trans. on Commun.*, vol. COM-39, No. 3, pp. 386-389, March 1991.

[9] K.J. Hole, "Punctured convolutional codes for the 1-D partial response channel," *IEEE Trans. Inform. Theory*, vol. IT-37, No. 3, pp. 808-817, May 1991.

[10] G.D. Forney, Jr., "Convolutional codes I: Algebraic structure," *IEEE Trans. Inform. Theory*, vol. IT-16, pp. 720-738, Nov. 1970. Correction in *IEEE Trans. Inform. Theory*, vol. IT-17, No. 3, p. 360, May 1971.

[11] G.D. Forney, Jr., "Minimal bases of rational vector spaces, with applications to multivariable linear systems," *SIAM J. Control*, vol. 13, pp.493-520, May 1975.

[12] R. Johannesson and Z.-X. Wan, "A linear algebra approach to convolutional encoders," *IEEE Trans. Inform. Theory*, vol. IT-39, No. 4, pp. 1219-1233, July 1993.

[13] F.-W. Sun, and A.J. Vinck, "An algorithm for identifying rate (n-1)/n catastrophic punctured convolutional encoders," *IEEE Trans. Inform. Theory*, vol. IT-42, No. 3, pp. 1010-1013, May 1996.

[14] T. Kailath, *Linear Systems Theory*, Wiley, New York, 1980.

Convolutional-like codes over discrete valuation rings and an application to 2-adic codes

Nicolas Lagorce
Laboratoire d'Arithmétique, Calcul formel et Optimisation,
Université de Limoges.
nicolas.lagorce@unilim.fr

Introduction

Classical convolutional codes are defined over fields of Laurent series with coefficients over a finite field. The main justification of this article is that these fields belong to the class of "fields of fractions of complete discrete valuation rings with finite residue field". In particular, extensions of p-adic fields also belong to this class. So, we will see that it is possible to define codes over the field of fraction of any complete discrete valuation ring with finite residue field in a way similar to the one used to define convolutional codes. Codes in this new class will be named "convolutional-like codes".
This abstract is organized in the following way: the first section is a small reminder about discrete valuation rings and classification of complete discrete valuation rings with finite residue field. In the second section, we will see the definition of convolutional-like codes and in the third we will specialize this definition for codes defined over the field of 2-adic numbers.
Note also that we use the term "2-adic code" to mean "convolutional-like code over the 2-adic field" and that they don't have the same meaning as the codes defined by A.R. Calderbank and N.J.A. Sloane in [3] and P. Solé in [13].

1 Discrete valuation rings

Definition 1.1. Let A be a commutative integral ring. A is a *discrete valuation ring* if it is principal and has an unique maximal ideal.

Proposition 1.2. *Let A be a discrete valuation ring and M its maximal ideal. The quotient A/M is a field called the residue field of A.*

Proposition 1.3. *Let A be a discrete valuation ring with M its maximal ideal, $\mathbb{F}$ its residue field and K its fraction field. As M is a principal ideal,*

it has a generator element, say π, called an uniformizer of A. Then

$$\begin{array}{rcl} A & = & \{\sum_{i\geq 0} x_i\pi^i \mid x_i \in \mathbb{F}\} \\ M & = & \{\sum_{i>0} x_i\pi^i \mid x_i \in \mathbb{F}\} \\ K & = & \{\sum_{i\geq d} x_i\pi^i \mid x_i \in \mathbb{F},\ d \in \mathbb{Z}\} \end{array}$$

Proposition 1.4. *The application*

$$\begin{array}{rccc} v: & K & \longrightarrow & \mathbb{Z}\cup\{+\infty\} \\ & \sum_{i\geq d} x_i\pi^i & \longmapsto & \min\{i \in \mathbb{Z} \mid x_i \neq 0\} \end{array}$$

is a valuation over K. Moreover, if $a > 1$ is a real number, the function $x \mapsto |x|$ defined by $|0| = 0$ and $|x| = a^{-v(x)}$ is an absolute value on K. This absolute value is extended to a distance over K by

$$\begin{array}{rccc} d: & K\times K & \longrightarrow & \mathbb{R}^+ \\ & (x,y) & \longmapsto & d(x,y) = |x-y|. \end{array}$$

Definition 1.5. The field K is said *complete* if, for every sequence (a_n) of elements of $\mathbb{F}$, the series $\sum_{i\geq 0} a_i\pi^i$ converge in K.

The following theorem completely classify the complete discrete valuation rings with finite residue field.

Theorem 1.6. *Let $\mathbb{F}$ be a finite field having characteristic p and A a complete discrete valuation ring with residue field $\mathbb{F}$. Only two cases can arise:*

(i) If A has characteristic p then A is isomorphic to the ring $\mathbb{F}[[T]]$ of formal series over $\mathbb{F}$. Its fraction field is isomorphic to the field $\mathbb{F}((T))$ of Laurent series over $\mathbb{F}$.

(ii) If A has characteristic zero then A is isomorphic to an extension of the ring $\mathbb{Z}_p$ of p-adic integers. Its fraction field is isomorphic to an extension of the field $\mathbb{Q}_p$ of p-adic numbers.

Definition 1.7. For an element $x = \sum_{i\geq r} x_i\pi^i$ of K, the *weight* of x, denoted by $w(x)$, is the number of its non-zero coefficients, that is $w(x) = \#\{i \in \mathbb{Z} \mid x_i \neq 0\}$.

2 Convolutional-like codes

Let $\mathbb{F}$ be a finite field having characteristic p. Let A be a complete discrete valuation ring with residue field $\mathbb{F}$ and K its field of fractions. We denote by A_f the sub-semiring of A containing the series of A whose general term is ultimately null (finite weight series).

Definition 2.1. A (n,k) *convolutional-like code* over K is a k dimensional subspace of the vector space K^n with a basis consisting entirely of vectors from A_f^n.

Definition 2.2. If $\mathcal{C}$ is a (n,k) convolutional-like code over K, an *encoder* of $\mathcal{C}$ is a $k \times n$ matrix over K whose rows form a basis of $\mathcal{C}$. This encoder is said *finite* if its coefficients are all in A_f.

Definition 2.3. Let $\mathcal{C}$ be a (n,k) convolutional-like code over K and G a finite encoder for $\mathcal{C}$. Let u be an element of K^k. The *codeword* corresponding to u in $\mathcal{C}$ is the element $x = uG$ in K^n.

According to the classification of field of fractions of complete discrete valuation rings (theorem 1.6), two types of codes are enclosed in definition 2.1. If K has characteristic p then we find again the classical definition of convolutional codes as it can be seen in, for example, [9]. If K has characteristic zero, then this definition identify a new class of codes with coefficients over extensions of p-adic fields.
Although some properties can be obtained directly, without distinguishing between these two cases (notably a notion of "state space"), we will not talk about that here. However, in the next section, we focus on 2-adic codes and we will see some basic properties of their encoders. (Note that the choice of 2-adic was driven by practical constraints and that theoretical results are generalized to p-adic codes in a straightforward way.)

3 2-adic codes

Definition 3.1. A 2-adic code with parameters (n,k) is a k dimensional subspace of the $\mathbb{Q}_2$ vector space $\mathbb{Q}_2^n$ with a basis consisting entirely of vectors from $\mathbb{N}^n$.

Remark. In the context of 2-adic codes, an encoder is said finite if all its components belong to $\mathbb{N}$.

3.1 Finite encoders

In this section, we denote by $\widehat{\mathbb{N}}$ the set $\{x/2^t \mid x \in \mathbb{N},\ t \geq 0\}$ and by $\widehat{\mathbb{Z}}$ the set $\{x/2^t \mid x \in \mathbb{Z},\ t \geq 0\}$. Let $\mathcal{C}$ be an (n,k) 2-adic code.
In the encoding process, we try to avoid the case where the image of a finite weight word is an infinite weight word as, in this case, a finite number of errors during the transmission can cause an infinite number of errors after the decoding process. Only the elements of $\widehat{\mathbb{N}}$ have a finite weight and there exists two types of infinite weight words in $\mathbb{Q}_2$, the elements of $\widehat{\mathbb{Z}} - \widehat{\mathbb{N}}$ and the elements of $\mathbb{Q}_2 - \widehat{\mathbb{Z}}$. (Note that negative elements of $\mathbb{Z}$ have an infinite weight as $-1 = \sum_{i \geq 0} 2^i$.) Conditions on the encoders can be given to eliminate these two types of series (theorem 3.2 and definition 3.4 respectively).
Moreover theorem 3.7 give an even stronger property insuring that, for every word, there isn't any time delay between encoding and transmission.

Theorem 3.2. *Let G be a finite encoder for C. The three following conditions are equivalent :*

(i) If $x = uG$ is an element of $\widehat{\mathbb{N}}^n$ then u is necessarily in $\widehat{\mathbb{Z}}^k$,

(ii) The gcd of the k-th order minors (determinants of $k \times k$ sub-matrices) of G is a power of 2,

(iii) G has a right inverse with coefficients over $\widehat{\mathbb{Z}}$.

An encoder satisfying these conditions is said weakly non-catastrophic

Definition 3.3. An $k \times n$ encoder G is said *quasi-monomial* if, for every $i = 1, \ldots, k$, there exists an index $1 \leq j \leq n$ such that $g_{l,j} = 0$ for all $l \neq i$ and $g_{i,j} \neq 0$.

Definition 3.4. Let G be a weakly non-catastrophic encoder. G is said *(strongly) non-catastrophic* if, when $x = uG$ is an element of $\widehat{\mathbb{N}}^n$, then necessarily u is an element of $\widehat{\mathbb{N}}^k$.

Theorem 3.5. *A weakly non-catastrophic encoder is strongly non-catastrophic if and only if it is quasi-monomial.*

Definition 3.6. Given a $k \times n$ matrix M over a field E. We denote by Δ_i the gcd of its i-th order minors, for $i = 1, \ldots, k$, and, by convention, $\Delta_0 = 1$. Then, for $i = 1, \ldots, k$, the i-th *invariant factor* of G is the element of E defined by $\gamma_i = \Delta_i / \Delta_{i-1}$.

Theorem 3.7. *A finite quasi-monomial encoder G is said basic if it satisfies one of the following five equivalent conditions:*

(i) The invariant factors of G are all 1,

(ii) The gcd of the k-th order minors of G is 1,

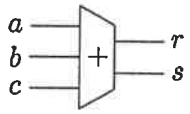
(iii) G has a right inverse with coefficients over $\mathbb{Z}$,

(iv) $x = uG$ is an element of $\mathbb{N}^n$ if and only if u is an element of $\mathbb{N}^k$,

(v) G is a sub-matrix of an invertible $n \times n$ matrix over $\mathbb{Z}$.

3.2 Physical realization and decoding

As for convolutional codes, it is possible to draw a blueprint for an actual physical device directly from a finite encoder of a given 2-adic code. In fact, the only difference between convolutional codes and 2-adic codes is the structure of binary adders. Adders for convolutional codes are XOR gates while adders for 2-adic codes are constituted by a three entries modulo 2 adder with carry and a delay flip-flop which re-inject the carry at the next



a	b	c	s	r
0	0	0	0	0
1	0	0	1	0
0	1	0	1	0
1	1	0	0	1
0	0	1	1	0
1	0	1	0	1
0	1	1	0	1
1	1	1	1	1

Figure 1: Binary adder with carry

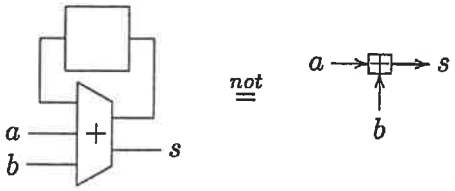


Figure 2: 2-adic adder

time index (figures 1 and 2). The blueprint for a simple 2-adic code is given in example in figure 3).
Another similarity between convolutional codes and 2-adic codes is the existence of a trellis structure, corresponding to a given finite encoder, in which the set of paths starting and ending in the zero state is equivalent to the set of codewords. The existence of this trellis permits the use of any trellis-oriented decoding algorithms (such as the Viterbi flow-calculus algorithm) to decode any received sequence. The figure 4 shows an example of trellis-module for the 2-adic code generated by $G = (3, 5)$. (In this figure an arrow denotes a state transition; it is plain if the input is 0 and dashed if the input is 1; its is the corresponding output vector.)
However, the number of states for the trellis-module of 2-adic codes is not as easy to compute as the one of a convolutional code. An upper bound can be given but it isn't very accurate. In fact, a 2-adic encoder contains two types of memory registers: the ones which hold past message bits and the ones which are enclosed in 2-adic adders. The reason for which the number of states isn't easy to know *a priory* is that, whereas the values of the registers of the first kind are free, the values of the registers of the second kind are driven by the global state of the encoder. So, as some configurations aren't reached from the zero state, the number of states isn't necessarily a power of 2.

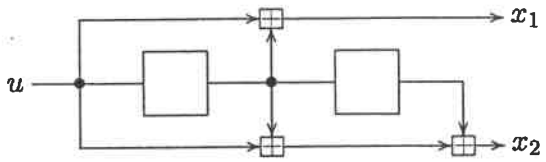


Figure 3: Physical realization of the 2-adic code generated by $G = (3, 7)$

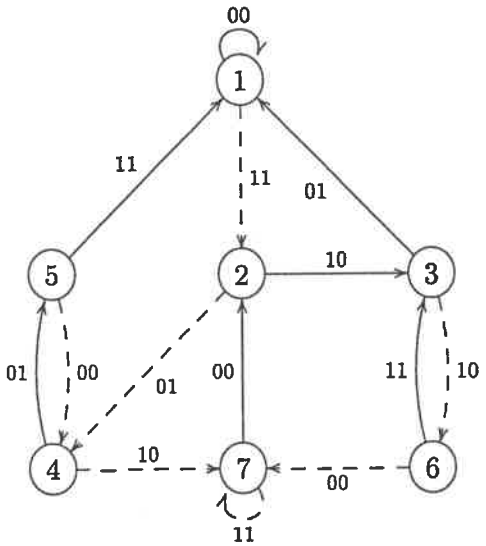


Figure 4: Trellis-module of the 2-adic code $(3, 5)$

Conclusion

As we have seen in this paper, the definition of convolutional codes can easily be generalized to other fields and notably to p-adic fields. Resulting codes still keep a great part of the properties that made convolutional codes so widely used (physical realization, soft decoding, ...). However, some problems arise and, in particular, the fact that the number of states of the trellis is not easily calculable a priori, and the fact that we don't know, for the moment, if the Hamming "distance" is appropriate to evaluate the quality of a 2-adic code.

Using the mapping from $\mathbb{F}_2((T))$ to $\mathbb{Q}_2$ sending one uniformizer to the other, we have made some simulations to compare the correcting power of the two versions of some $(2, 1)$ codes. We obtained that the 2-adic version give a slightly better bit error rate at the expense of an higher decoding work. Nevertheless, it must be noted that we only used convolutional codes which

are known to be extremal and that we don't even know if their 2-adic version are "good". Moreover, the fact that, for a given 2-adic code, the sequence of carries is only dependent of the message (in fact, of the repartition of the ones in the message) may be exploited to obtain a substantial correcting power gain.

References

[1] Y. Amice. *Les nombres p-adiques.* Presses Universitaires de France, 1975.

[2] I. Blake. Codes over certain rings. *Information and control*, 1972.

[3] A.R. Calderbank and N.J.A. Sloane. Modular and p-adic cyclic codes. *Designs, Codes and Cryptography*, vol. 6:pp. 21–35, 1995.

[4] H. Cohen. *A course in computational algebraic number theory.* Graduate texts in Mathematics. Springer, 1995.

[5] F.Q. Gouvea. *p-adic Numbers.* Springer-Verlag, 1991.

[6] R. Hammons, V. Kumar, A.R. Calderbank, N.J.A. Sloane, and P. Solé. The $\mathbb{Z}_4$-linearity of Kerdock, Preparata, Goethals and related codes. *IEEE Transactions on Information theory*, 1994.

[7] H. Hasse. *Number thoery.* Springer-Verlag, 1980.

[8] S. Lang. *Algebra.* Addison-Wesley, 1995.

[9] R.J. McEliece. *The algebraic theory of convolutional codes.* in Handbook of Coding Theory, 1996.

[10] P. Piret. *Convolutional codes : an algebraic approach.* The MIT Press, 1988.

[11] P. Ribenboim. *L'Arithmétique des corps.* Hermann Paris, 1972.

[12] J.P. Serre. *Corps locaux.* Hermann Paris, 1962.

[13] P. Solé. Open problem 2: cyclic codes over rings and p-adic fields. In G. Cohen and J. Wolfmann, editors, *Coding theory and applications*, number 388 in Lecture Notes in Computer Science, page 329. Springer-Verlag, New York, 1988.

[14] J. Vuillemin. On circuits and numbers. *Rapport Digital Equipment Corp. (PRL)*, 1993.

Index des auteurs – *Author index*