A New Algorithm for Finding Minimum-Weight Words in a Linear Code: Application to Primitive Narrow-Sense BCH Codes of Length 511

Anne Canteaut, Florent Chabaud

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

A new algorithm for finding minimum-weight words in a linear code: application to primitive narrow-sense BCH codes of length 511

Anne Canteaut et Florent Chabaud

N° 2685

octobre 1995

PROGRAMME 2

ISSN 0249-6399

A new algorithm for finding minimum-weight words in a linear code: application to primitive narrow-sense BCH codes of length 511

Anne Canteaut * et Florent Chabaud **

Programme 2 — Calcul symbolique, programmation et génie logiciel
Projet Codes

Rapport de recherche n°2685 — octobre 1995 — 19 pages

Abstract: An algorithm for finding small-weight words in large linear codes is developed. It is in particular able to decode random [512,256,57]-linear codes in 9 hours on a DEC alpha computer. We determine with it the minimum distance of some binary BCH codes of length 511, which were not known.

Key-words: error-correcting codes, decoding algorithm, minimum weight, random linear codes, BCH codes.

(Résumé : tsvp)

submitted to IEEE Transactions on Information Theory

*Also with École Nationale Supérieure de Techniques Avancées, laboratoire LEI, 32 boulevard Victor, F-75015 Paris.
**Laboratoire d'Informatique de l'Ecole Normale Supérieure, 45 rue d'Ulm, 75230 Paris Cedex 05

Unité de recherche INRIA Rocquencourt
Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
Téléphone : (33 1) 39 63 55 11 – Télécopie : (33 1) 39 63 53 30

Un nouvel algorithme pour trouver des mots de poids minimum dans un code linéaire : application aux codes BCH primitifs au sens strict de longueur 511

Résumé : Nous développons un algorithme de recherche de mots de poids minimum dans un code linéaire. Cet algorithme nous permet notamment de décoder les codes linéaires aléatoires de paramètres [512,256,57] en 9 heures sur une station DEC alpha. Grâce à lui, nous déterminons la distance minimale de certains codes BCH binaires de longueur 511, lesquelles étaient jusqu'alors inconnues.

Mots-clé : codes correcteurs d'erreurs, algorithme de décodage, poids minimum, codes linéaires aléatoires, codes BCH

1 Introduction

In this paper we present a probabilistic algorithm for finding small-weight words in any linear code. This algorithm could be applied to two important NP-complete problems [2]: computing the minimum distance of a linear code and decoding. It associates an iterative procedure stemming from linear programming for reducing the computational cost of many Gaussian eliminations and a heuristic proposed by Stern. We then give a very precise analysis of the complexity of our algorithm which enables us to optimize the parameters it depends from. Hence our algorithm is to our knowledge the best procedure for decoding without using the structure of the code.

Section 2 describes our algorithm for binary codes but it could be generalized to linear codes over GF(q) (see [4]). Using Markov chain theory we show in section 3 how to compute the number of elementary operations it requires; we give then the parameters which minimize this theoretical running-time and an explicit expression which approximates the number of operations performed for decoding and for finding a minimum-weight word in a random $[n, k]$-binary code. Two applications of our algorithm are then exposed: section 4 presents experimental results for decoding [256,128]-binary linear codes which validate the previous theoretical approach; section 5 gives new results for the true minimum distance of some narrow-sense BCH codes of length 511.

2 Description of the algorithm

As usual wt(x) will denote the Hamming weight of the binary word x.

Let $\mathcal{C}$ be a linear code over GF(2) of length n and dimension k. We try now to find a codeword of weight w where w is small.

2.1 The probabilistic method

Enumerating randomly selecting codewords in hope than one of weight w will be found is obviously not a suitable algorithm because the probability that the weight of a random codeword will be w is very small. It is then necessary to bias this random selection by only examining codewords verifying a given property so that their weight will be *a priori* small, for example codewords which vanish on a randomly chosen coordinate subset. The problem is therefore to find a compromise between the number of operations required by searching such particular codewords and the success probability, *i.e.* the probability that the weight of such a codeword will be w.

All algorithms for finding short codewords use therefore the same method [10], [11], [14]: they only take in account codewords which are particular linear combinations of a small number of rows of a systematic generator matrix. The algorithm proposed by Stern [14] and slightly modified was shown to give the best results [5].

Let $N = \{1, \cdots, n\}$ be the set of all coordinates. For any subset I of N, $G = (V, W)_I$ denotes the decomposition of matrix G onto I, that means $V = (G_i)_{i \in I}$ and $W = (G_j)_{j \in N \setminus I}$, where G_i is the ith column of matrix G.

definition 1 *Let I be a k-element subset of N. I is an information window for code $\mathcal{C}$ iff $G = (Id_k, Z)_I$ is a systematic generator matrix for the code. The complementary set, $J = N \setminus I$, is called a redundancy window.*

From now on we index the rows of Z with I since $G = (Id_k, Z)_I$ is a generator matrix for the code and we denote by Z^i the i-th row of matrix Z.

The idea suggested by Stern is to randomly choose at each iteration an information window I which is split into two parts I_1 and I_2 of same size, and a subset L of J of size ℓ. We only examine codewords c verifying the following property, where p is a fixed parameter for the algorithm:

$$\mathrm{wt}(c_{|I_1}) = \mathrm{wt}(c_{|I_2}) = p \text{ and } \mathrm{wt}(c_{|L}) = 0 \tag{1}$$

until we find such a particular codeword whose restriction on $J \setminus L$ has weight $w - 2p$.

All codewords which verify condition 1 can easily be constructed as soon as the corresponding systematic generator matrix $G = (Id_k, Z)_I$ is known:

- randomly split the rows of Z into two subsets Z_1 et Z_2 corresponding to I_1 and I_2.
- for each linear combination Λ_1 of p rows of matrix Z_1, compute $\Lambda_{1|L}$.
- for each linear combination Λ_2 of p rows of matrix Z_2, compute $\Lambda_{2|L}$.
- if $\Lambda_{1|L} = \Lambda_{2|L}$, check whether $\mathrm{wt}((\Lambda_1 + \Lambda_2)_{|J \setminus L}) = w - 2p$.

2.2 The iterative procedure

The previous algorithm therefore explores a set of randomly selected information windows by performing at each iteration a Gaussian elimination on an $(n \times k)$-generator matrix. In order to avoid this time-consuming procedure, we here propose to choose at each step the new information window by modifying only one element of the previous one. This method is analogous to the one used in the simplex method as suggested in [12] and [15, 3].

definition 2 *Two information windows I and I' are close iff:*

$$\exists \lambda \in I,\ \exists \mu \in N \setminus I,\ \textit{such that } I' = (I \setminus \{\lambda\}) \cup \{\mu\}$$

As any two information windows can be joined by a sequence of close information windows, we use this iterative method in order to find one which enables us to discover a codeword of weight w.

The following proposition shows how choosing λ and μ such that I' is still an information window.

proposition 1 *Let I be an information window such that $G = (Id_k, Z)_I$ is a generator matrix for $\mathcal{C}$. Let be $\lambda \in I$, $\mu \in J$ and $I' = (I \setminus \{\lambda\}) \cup \{\mu\}$.*
I' is an information window iff $z_{\lambda,\mu} = 1$, where $Z = (z_{i,j})_{i \in I, j \in J}$

Proof. Since $G = (Id_k, Z)_I$, we have: $G_\mu = z_{\lambda,\mu} G_\lambda + \sum_{i \in I \setminus \{\lambda\}} z_{i,\mu} G_i$
As the columns indexed by I are linearly independent, G_μ and $(G_i)_{i \in I \setminus \{\lambda\}}$ are linearly independent iff $z_{\lambda,\mu} = 1$.

□

As the probabilistic method procedure only deals with the redundant part of the systematic generator matrix, we only need a procedure able to obtain the redundant matrix Z' corresponding to I' from Z.

proposition 2 *Let I and I' be two close information windows such that $I' = (I \setminus \{\lambda\}) \cup \{\mu\}$. Let $(Id_k, Z)_I$ and $(Id_k, Z')_{I'}$ be the corresponding systematic generator matrices. Then Z' is obtained from Z by:*

- $\forall j \in J', \;\; z'_{\mu,j} = z_{\lambda,j}$
- $\forall i \in I' \setminus \{\mu\}$,
  - $\forall j \in J' \setminus \{\lambda\}, \;\; z'_{i,j} = z_{i,j} + z_{i,\mu} z_{\lambda,j}$
  - $z'_{i,\lambda} = z_{i,\mu}$

Proof. As $I' = (I \setminus \{\lambda\}) \cup \{\mu\}$, $(Id_k, Z')_{I'}$ is obtained by exchanging the λ-th and μ-th columns of $(Id_k, Z)_I$. This can be done by a simple pivoting operation in position (λ, μ), *i.e.* by adding the λ-th row of matrix Z to all other rows Z^i iff the corresponding element $z_{i,\mu}$ is not equal to zero.

□

2.3 Description of the iterative algorithm

The use of this iterative procedure leads then to the following algorithm:

Initialization:
Randomly choose an information window I and apply a Gaussian elimination in order to obtain a systematic generator matrix $(Id_k, Z)_I$.

Until a codeword of weight w will be found:

- randomly split I in two subsets I_1 and I_2 where $|I_1| = \lfloor k/2 \rfloor$ and $|I_2| = \lceil k/2 \rceil$. The rows of Z are then split in two parts Z_1 and Z_2.
- randomly select an ℓ-element subset L of J.

- for each linear combination Λ_1 of p rows of matrix Z_1, compute $\Lambda_{1|L}$.
- for each linear combination Λ_2 of p rows of matrix Z_2, compute $\Lambda_{2|L}$.
- if $\Lambda_{1|L} = \Lambda_{2|L}$, check whether $\mathrm{wt}((\Lambda_1 + \Lambda_2)_{|J\setminus L}) = w - 2p$.
- randomly choose $\lambda \in I$ and $\mu \in J$. Replace I with $(I \setminus \{\lambda\}) \cup \{\mu\}$ by updating matrix Z according to the preceding proposition.

Remark: This algorithm can also be used for decoding. Let $x = c+e$ be a corrupted message where c is a codeword and e an error-vector of weight w such that $w \leq t = \lfloor \frac{d-1}{2} \rfloor$ where d is the minimum distance of the code. Matrix

$$\Gamma = \left(\frac{G}{x} \right)$$

is a generator matrix of an $[n, k+1, t]$-code. Our algorithm then enables us to recover the minimum-weight word of this new code, which is the error-vector e.

3 Theoretical running-time

We give here an explicit and computable expression for the work factor of this algorithm, *i.e.* the average number of elementary operations it requires.

3.1 Average number of operations by iteration

1. There are exactly $\binom{k/2}{p}$ linear combinations of p rows of matrix Z_1 (resp. Z_2); computing each of them on an ℓ-bit selection requires $p\ell$ binary additions.
2. The average number of collisions *i.e.* the average number of pairs (Λ_1, Λ_2) such that $(\Lambda_1 + \Lambda_2)_{|L} = 0$ is equal to $\frac{\binom{k/2}{p}^2}{2^\ell}$. For each collision we must perform $2p-1$ additions of $(n-k-\ell)$-bit words for computing $(\Lambda_1 + \Lambda_2)_{|J\setminus L}$ and a weight-checking.
3. We need $K(p\binom{k/2}{p} + 2^\ell)$ more operations to perform the dynamic memory allocation where K is the size of a computer word (K=32 or 64).
4. For updating matrix Z according to proposition 2 we have to add row Z^λ to all other rows Z^i when $z_{i,\mu} = 1$. Assuming that the average weight of column Z_μ is $k/2$, the work factor involved in this procedure is $\frac{1}{2}k(n-k)$.

Hence the average number of elementary operations performed at each iteration is:

$$\Omega_{p,\ell} = 2p\ell\binom{k/2}{p} + 2p(n-k-\ell)\frac{\binom{k/2}{p}^2}{2^\ell} + K(p\binom{k/2}{p} + 2^\ell) + \frac{k(n-k)}{2} \qquad (2)$$

3.2 Expected number of iterations

The average number of iterations performed by the algorithm is not the same as the one performed by the initial Stern's algorithm since the successive information windows are not independent anymore. Hence the algorithm must be modeled by a discrete-time stochastic process.

Let c be the codeword of weight w to recover and $\mathrm{supp}(c)$ its support. Let I be the information window and I_1, I_2 and L the other selections corresponding to the i-th iteration.

The i-th iteration can then be associated with the random variable X_i whose state space is $\mathcal{E} = \{0, \ldots, 2p-1\} \cup \{(2p)_S, (2p)_F\} \cup \{2p+1, \ldots, w\}$ where

$$\begin{array}{lll} X_i = u & \text{iff} & |I \cap \mathrm{supp}(c)| = u, \ \ \forall u \in \{0, \ldots, 2p-1\} \cup \{2p+1, \ldots, w\} \\ X_i = (2p)_F & \text{iff} & |I \cap \mathrm{supp}(c)| = 2p \text{ and } (|I_1 \cap \mathrm{supp}(c)| \neq p \\ & & \text{or } |I_2 \cap \mathrm{supp}(c)| \neq p \text{ or } |L \cap \mathrm{supp}(c)| \neq 0) \\ X_i = (2p)_S & \text{iff} & |I_1 \cap \mathrm{supp}(c)| = |I_2 \cap \mathrm{supp}(c)| = p \text{ and } |L \cap \mathrm{supp}(c)| = 0 \end{array}$$

The success space is then $\mathcal{S} = \{(2p)_S\}$ and the failure space is $\mathcal{F} = \{0, \ldots, (2p)_F, \ldots, w\}$.

proposition 3 *The stochastic process $\{X_i\}_{i \in \mathbb{N}}$ associated with the algorithm is an homogeneous Markov chain.*

Proof. The selections I, I_1, I_2 and L corresponding to the i-th iteration only depend on the previous information window since I_1, I_2 and L are randomly chosen. Then we have for all i and for all $(u_0, u_1, \cdots, u_i) \in \mathcal{E}$,

$$Pr[X_i = u_i / X_{i-1} = u_{i-1}, \ X_{i-2} = u_{i-2}, \cdots, X_0 = u_0] = Pr[X_i = u_i / X_{i-1} = u_{i-1}].$$

Furthermore this probability does not depend on the iteration. Hence there exists a matrix P such that :

$$\forall i \in \mathbb{N}, \ \forall (u, v) \in \mathcal{E}^2, \ \ Pr[X_i = v / X_{i-1} = u] = P_{u,v}$$

□

$\{X_i\}_{i \in \mathbb{N}}$ is therefore completely determined by its starting distribution π_0 and its transition matrix P.

proposition 4 *The transition matrix P associated with the homogeneous Markov chain representing the algorithm is given by:*

$$\begin{aligned} P_{u,u} &= \frac{k-u}{k} \times \frac{n-k-(w-u)}{n-k} + \frac{u}{k} \times \frac{w-u}{n-k} \ \textit{for all } u \in \mathcal{E} \setminus \{(2p)_S, (2p)_F\} \\ P_{u,u-1} &= \frac{u}{k} \times \frac{n-k-(w-u)}{n-k} \ \textit{for all } u \neq 2p+1 \\ P_{u,u+1} &= \frac{k-u}{k} \times \frac{w-u}{n-k} \ \textit{for all } u \neq 2p-1 \end{aligned}$$

$$
\begin{aligned}
P_{u,v} &= 0 \text{ for all } v \notin \{u-1, u, u+1\} \\
P_{(2p)_F,(2p)_F} &= (1-\beta)\left[\frac{k-2p}{k} \times \frac{n-k-(w-2p)}{n-k} + \frac{2p}{k} \times \frac{w-2p}{n-k}\right] \\
P_{2p+1,(2p)_F} &= (1-\beta)\left[\frac{2p+1}{k} \times \frac{n-k-(w-(2p+1))}{n-k}\right] \\
P_{2p-1,(2p)_F} &= (1-\beta)\left[\frac{k-(2p-1)}{k} \times \frac{w-(2p-1)}{n-k}\right] \\
P_{2p+1,(2p)_S} &= \beta\left[\frac{2p+1}{k} \times \frac{n-k-(w-(2p+1))}{n-k}\right] \\
P_{2p-1,(2p)_S} &= \beta\left[\frac{k-(2p-1)}{k} \times \frac{w-(2p-1)}{n-k}\right] \\
P_{(2p)_F,(2p)_S} &= \beta\left[\frac{k-2p}{k} \times \frac{n-k-(w-2p)}{n-k} + \frac{2p}{k} \times \frac{w-2p}{n-k}\right] \\
P_{(2p)_S,(2p)_S} &= 1 \\
P_{(2p)_S,u} &= 0 \text{ for all } u \neq (2p)_S
\end{aligned}
$$

The initial probability vector is

$$\pi_0 = \left(\frac{\binom{w}{u}\binom{n-w}{k-u}}{\binom{n}{k}}\right)_{0 \leq u \leq w}.$$

Since the single success state is an absorbing state, $\{X_i\}_{i \in \mathbb{N}}$ is a transient chain. The following theorem can then be applied for computing the excepted number of iterations performed by the algorithm.

proposition 5 [9] *If $\{X_i\}_{i \in \mathbb{N}}$ is a transient chain with transition matrix P, and Q is the sub-stochastic matrix corresponding to transitions among the transient states,* i.e. $Q = (P_{u,v})_{\substack{u \in \mathcal{F} \\ v \in \mathcal{F}}}$ *then $(Id - Q)$ has an inverse R called the fundamental matrix of the chain and*

$$R = \sum_{m=0}^{\infty} Q^m = (Id - Q)^{-1}.$$

theorem 1 *The expectation $\bar{N}$ of the number of iterations required until X_n reaches a success state is given by:*

$$\bar{N} = \sum_{u \in \mathcal{F}} \pi_0(u) \sum_{v \in \mathcal{F}} R_{u,v}$$

where R is the corresponding fundamental matrix.

Proof.

$$\begin{aligned}\bar{N} &= \sum_{n\geq 0} nPr[N=n] \\ &= \sum_{n\geq 1} Pr[N\geq n] \\ &= \sum_{n\geq 0} Pr[X_n \in \mathcal{F}].\end{aligned}$$

As the initial probability distribution is vector π_0, we have:

$$\bar{N} = \sum_{n\geq 0}\sum_{u\in\mathcal{F}} Pr[X_n \in \mathcal{F}/X_0 = u]\pi_0(u)$$

Let $Q = (P_{u,v})_{\substack{u\in\mathcal{F}\\ v\in\mathcal{F}}}$ then we have

$$Pr[X_i \in \mathcal{F}/X_0 = u] = \sum_{v\in\mathcal{F}} (Q^i)_{u,v}.$$

Thanks to the preceding proposition we finally obtain

$$\bar{N} = \sum_{u\in\mathcal{F}} \pi_0(u) \sum_{v\in\mathcal{F}} R_{u,v}$$

□

proposition 6 *Suppose that the number of codewords of weight w is $\mathcal{A}_w$. The overall work factor required by the algorithm is:*

$$W_{p,l} = \frac{\Omega_{p,\ell}\bar{N}}{\mathcal{A}_w} \tag{3}$$

where $\bar{N}$ is given by theorem 1 and $\Omega_{p,\ell}$ by equation 2.

3.3 Theoretical running-time for the general decoding and minimum-weight problems

Using the implicit formula 3 we now give the optimal parameters p and ℓ for some basic problems. The following work factors are computed for some $[n,k]$-random binary codes whose minimum distance is obtained with the Gilbert-Varshamov's bound. Comparisons with other decoding algorithms [10, 11, 14] are made in [4]; the work factor required by the initial Stern's algorithm is shown to be at least 4 times higher for the following problems.

3.3.1 Decoding random linear codes

We here give the optimal parameters and the work factors required for recovering an error-vector of weight $t = \lfloor \frac{d-1}{2} \rfloor$. We recall that p corresponds to the number of rows of each part of the generator matrix we use for the linear combinations and that ℓ is the size of the selection L on which the examined words vanish. Note that the code we consider for computing the work factor is an $[n, k+1]$-code according to the remark in section 2.3.

code	[64,32,7]	[128,64,15]	[256,128,29]	[512,256,57]	[768,384,85]
t	3	7	14	28	42
optimal parameters	$p=1$ $\ell=4$	$p=1$ $\ell=6$	$p=1$ $\ell=7$	$p=1$ $\ell=9$	$p=2$ $\ell=17$
work factor	$2^{15.39}$	$2^{19.36}$	$2^{26.51}$	$2^{40.48}$	$2^{54.55}$

[1024,512,113]	[1536,768,170]	[2048,1024,226]
56	84	112
$p=2$ $\ell=18$	$p=2$ $\ell=19$	$p=2$ $\ell=20$
$2^{68.51}$	$2^{96.87}$	$2^{125.50}$

Table 1: Optimal parameters for decoding random $[n, n/2]$ binary codes

We see in figure 1 that, for a fixed rate $r = k/n$, $\log_2(W)$ linearly depends on n when parameters p and ℓ are optimized and that the work factor can be written in the form $W_{opt} = 2^{na(r)+b}$.

Figure 2 shows how parameters p and ℓ act on the work factor for decoding a random $[n, n/2]$ code: if they are not optimized, $\log_2(W)$ does not linearly depend on n anymore.

Furthermore we see in figure 3 that $a(r)$ is closed to the entropy function $H_2(r)$ multiplied by a fixed coefficient, where $H_2(x) = -x\log_2(x) - (1-x)\log_2(1-x)$.

proposition 7 *The theoretical work factor required for decoding a random $[n, k]$-binary code can be approximated by the following formula:*

$$W_{opt} = 2^{naH_2(k/n)+b} \text{ where } a = 5.511\ 10^{-2} \text{ and } b = 12$$

3.3.2 Finding minimum-weight codewords

We now give the theoretical complexity of the algorithm for recovering a word of weight d in a random $[n, k]$ binary code, where d is given by the Gilbert-Varshamov's bound. We assume that all these codes contain exactly one word of weight d.

code	[64,32,7]	[128,64,15]	[256,128,29]
optimal parameters	$p=1$ $\ell=4$	$p=1$ $\ell=5$	$p=1$ $\ell=7$
work factor	$2^{17.93}$	$2^{25.55}$	$2^{40.65}$

[384,192,43]	[512,256,57]	[640,320,71]
$p=2$ $\ell=14$	$p=2$ $\ell=15$	$p=2$ $\ell=16$
$2^{55.77}$	$2^{70.72}$	$2^{85.78}$

Table 2: Optimal parameters for finding a minimum-weight word in random $[n, n/2]$-binary codes

As for the general decoding problem $\log_2(W_{opt})$ linearly depends on n for a fixed rate $r = k/n$ (see figure 4).

If this work factor is written in the form $W_{opt} = 2^{nc(r)n+d}$, figure 5 shows that $c(r)$ is closed to the translated entropy function $cH_2(r + r_0)$.

proposition 8 *The theoretical work factor required for finding a minimum-weight word in a random $[n, k]$-binary code can be approximated by the following formula:*

$$W_{opt} = 2^{ncH_2(\frac{k}{n}+r_0)+d} \text{ where } c = 0.12,\ d = 10 \text{ and } r_0 = 3.125\ 10^{-2}$$

4 Experimental results for decoding random $[256, 128]$-binary codes

In order to check the correctness of our optimization we have made a great number of simulations for a small problem: decoding a random [256,128,29]-binary code, *i.e.* recovering an error-vector of weight 14. For each set of parameters 1000 computations have been made on a DEC alpha computer running at 175 MHz. The results given in table 3 confirm the validity of the previous theory. We see that decoding a random [256,128,29]-code requires around 2 seconds. Thus decoding a [512,256,57]-one requires around 9 hours on our computer.

5 True minimum distance of primitive binary narrow-sense BCH codes of length 511

Let $q = 2^m$ and $n = 2^m - 1$. Let α be a primitive n-th root of unity in GF(q).

parameters	theoretical work factor $\log_2(W)$	theoretical average iteration number	experimental average iteration number	deviation %	average CPU time (s)	corrected CPU time (s)
$p=1$, $\ell=5$	27.37	3961	4072	+2.80	2.76	2.68
$p=1$, $\ell=6$	26.80	4045	3985	-1.48	2.11	2.14
$p=1$, $\ell=7$	26.51	4139	4190	+1.23	2.07	2.04
$p=1$, $\ell=8$	26.56	4244	4338	+2.21	2.13	2.09
$p=1$, $\ell=9$	26.95	4362	4417	+1.26	2.90	2.87
$p=2$, $\ell=10$	29.80	432	433	+0.35	13.84	13.79
$p=2$, $\ell=11$	29.04	442	470	+6.51	13.00	12.21
$p=2$, $\ell=12$	28.51	454	446	-1.76	12.13	12.35
$p=2$, $\ell=13$	28.37	466	487	+4.51	17.70	16.91
$p=2$, $\ell=14$	28.68	480	508	+5.83	29.40	27.72

Table 3: Decoding random $[256, 128, 29]$-binary codes

definition 3 *The primitive binary narrow-sense BCH code of length $n = 2^m - 1$ and designed distance δ, denoted by $B(n, \delta)$, is the largest binary cyclic code of length n having zeros*

$$\alpha, \alpha^2, \ldots, \alpha^{\delta-1}$$

The minimum distance d of $B(n, \delta)$ satisfies the BCH bound: $d \geq \delta$.

There is no general method for finding the true minimum distance of a BCH code although several infinite classes are known to have minimum distance equal to their designed distance [13, 7, 1]. The true minimum distance has been determined for all narrow-sense BCH codes of length less than 511. For the length 511 the true minimum distance of 12 of them is still unknown [1]. Using the previously presented algorithm we show that the designed distance is reached for 5 of these codes.

It is well-known that the automorphism group of an extended primitive BCH code contains the affine group on $\mathrm{GF}(2^m)$; then we have the following proposition:

proposition 9 *If $B(2^m - 1, \delta)$ contains a word of even weight w then it contains a word of weight $w - 1$.*

Thus we obtain:

theorem 2 *$B(511, 29)$, $B(511, 37)$, $B(511, 41)$, $B(511, 43)$ and $B(511, 87)$ have minimum distance equal to their designed distance.*

Proof. Let us define $\mathrm{GF}(2^9)$ by $\alpha^9 + \alpha^4 + 1 = 0$. For each of these $B(511, \delta)$ a word of weight $\delta + 1$ was found. We here list all the corresponding exponents.

- $\delta = 29$**:**
(26, 31, 38, 51, 64, 72, 112, 126, 139, 142, 157, 188, 222, 227 , 265, 270, 301, 306, 307, 317, 347, 354, 368, 369, 412, 415, 423, 431, 494, 498)

- $\delta = 37$**:**
(4, 13, 27, 48, 56, 94, 102, 103, 115, 118, 132, 149, 152, 159, 197, 202, 215, 232, 240, 249, 250, 251, 290, 324, 327, 349, 359 , 360, 367, 383, 396, 423, 461, 493, 494, 499, 504, 509)

- $\delta = 41$**:**
(9, 20, 30, 37, 38, 42, 43, 53, 66, 68, 83, 93, 95, 106, 108, 110, 111, 175, 185, 202, 234, 250, 262, 270, 321, 342, 362, 363 , 379, 382, 385, 401, 402, 410, 426, 436, 462, 467, 478, 482, 499, 507)

- $\delta = 43$**:**
(0, 16, 35, 38, 56, 57, 58, 80, 82, 87, 115, 134, 147, 148, 156 , 165, 167, 190, 196, 206, 229, 240, 242, 258, 269, 284, 295, 296, 309, 317, 321, 322, 324, 325, 326, 361, 375, 394, 405, 418 , 429, 444, 460, 492)

- $\delta = 87$**:**
(18, 19, 23, 25, 27, 43, 50, 51, 64, 70, 73, 77, 81, 88, 96, 101 , 102, 116, 117, 143, 146, 152, 158, 163, 165, 166, 173, 179, 192, 193, 195, 197, 199, 203, 210, 212, 225, 230, 240, 244, 252 , 263, 272, 283, 287, 290, 292, 293, 295, 297, 301, 306, 320, 323, 327, 330, 339, 353, 361, 382, 385, 392, 394, 400, 411, 414 , 417, 421, 432, 436, 446, 453, 459, 461, 466, 474, 475, 476, 480, 481, 483, 487, 489, 492, 503, 504, 505, 510)

Table 4 gives the list of all narrow-sense BCH codes of length 511, their minimum distance and the way they were found.

Acknowledgements: We wish to thank Hervé Chabanne who initiated this work.

References

[1] D. Augot, P. Charpin, and N. Sendrier. Studying the locator polynomials of minimum weight codewords of BCH codes. *IEEE Trans. Inform. Theory*, 38(3):960–973, 1992.

[2] E.R. Berlekamp, R.J. McEliece, and H.C.A. Van Tilborg. On the inherent intractability of certain coding problems. *IEEE Trans. Inform. Theory*, IT-24(3):384–386, May 1978.

[3] A. Canteaut and H. Chabanne. A further improvement of the work factor in an attempt at breaking McEliece's cryptosystem. In P. Charpin, editor, *EUROCODE 94*, pages 163–167. INRIA, 1994.

[4] A. Canteaut and F. Chabaud. Improvements of the attacks on cryptosystems based on error-correcting codes. Rapport interne du Departement Mathematiques et Informatique LIENS-95-21, Ecole Normale Supérieure, Paris, July 1995.

[5] F. Chabaud. On the security of some cryptosystems based on error-correcting codes. In *pre-proceedings of EUROCRYPT '94*. Springer-Verlag, 1994.

[6] H.J. Helgert and R.D. Stinaff. Shortened BCH codes. *IEEE Trans. Inform. Theory*, pages 818–820, November 1973.

[7] T. Kasami and S. Lin. Some results on the minimum weight of primitive BCH codes. *IEEE Trans. Inform. Theory*, pages 824–825, November 1972.

[8] T. Kasami and N. Tokura. Some remarks on BCH bounds and minimum weights of binary primitive BCH codes. *IEEE Trans. Inform. Theory*, pages 408–413, May 1969.

[9] J.G. Kemeny and J.L. Snell. *Finite Markov chains*. D.Van Nostrand, Princeton, New Jersey, 1960.

[10] P.J. Lee and E.F. Brickell. An observation on the security of McEliece's public-key cryptosystem. In C.G. Günther, editor, *Advances in Cryptology – EUROCRYPT '88*, number 330 in Lecture Notes in Computer Science, pages 275–280. Springer-Verlag, 1988.

[11] J.S. Leon. A probabilistic algorithm for computing minimum weights of large error-correcting codes. *IEEE Trans. Inform. Theory*, IT-34(5):1354–1359, September 1988.

[12] J.K. Omura. Iterative decoding of linear codes by a modulo-2 linear program. *Discrete Math*, 3:193–208, 1972.

[13] W.W. Peterson and E.J. Weldon. *Error-Correcting Codes*. MIT Press, 1961.

[14] J. Stern. A method for finding codewords of small weight. In G. Cohen and J. Wolfmann, editors, *Coding Theory and Applications*, number 388 in Lecture Notes in Computer Science, pages 106–113. Springer-Verlag, 1989.

[15] J. van Tilburg. On the McEliece public-key cryptosystem. In S. Goldwasser, editor, *Advances in Cryptology – CRYPT0 '88*, number 403 in Lecture Notes in Computer Science, pages 119–131. Springer-Verlag, 1990.

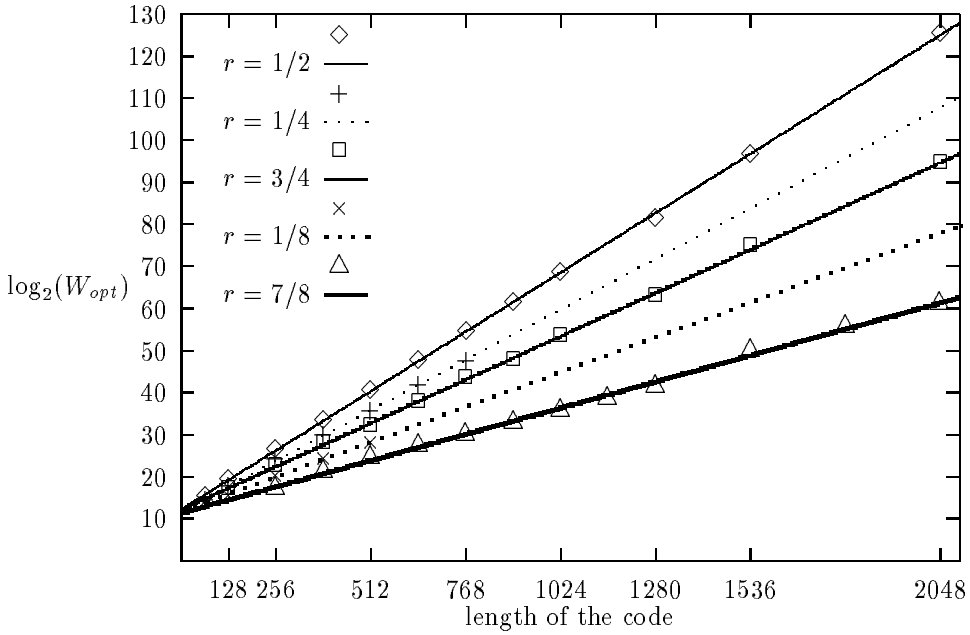

Figure 1: Evolution of the theoretical work factor with optimized parameters for decoding random $[n, nr]$-binary codes

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 46 avenue Félix Viallet, 38031 GRENOBLE Cedex 1
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
ISSN 0249-6399

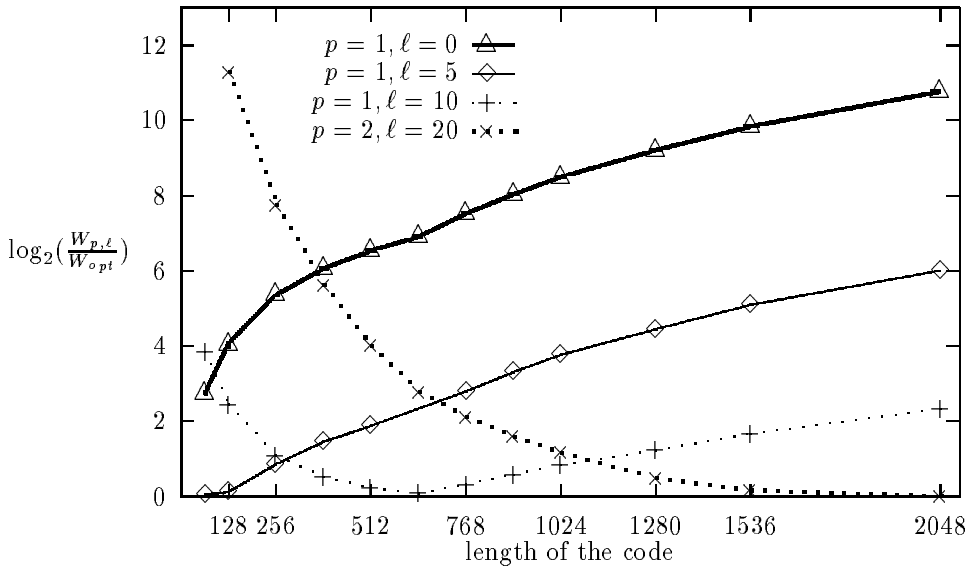

Figure 2: Influence of parameters p and ℓ on the work factor for decoding $[n, n/2]$ random codes

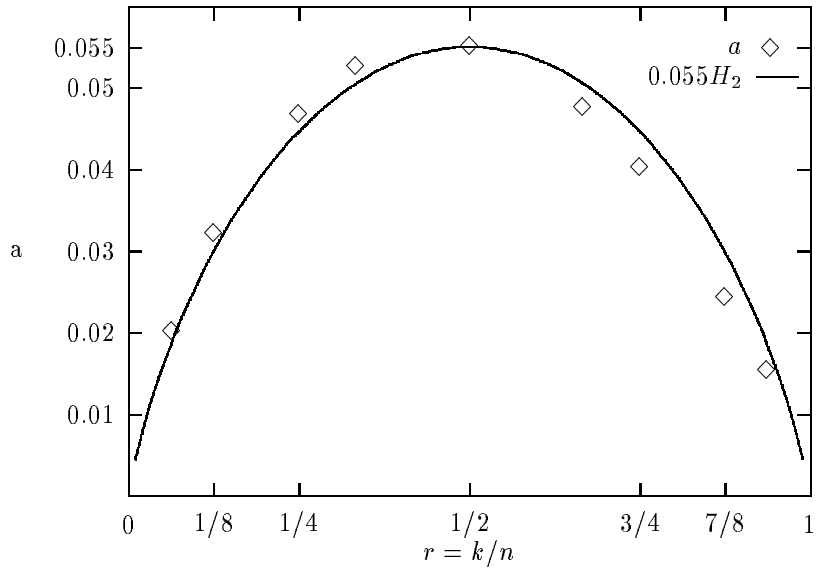

Figure 3: Evolution of coefficient a vs. $r = k/n$

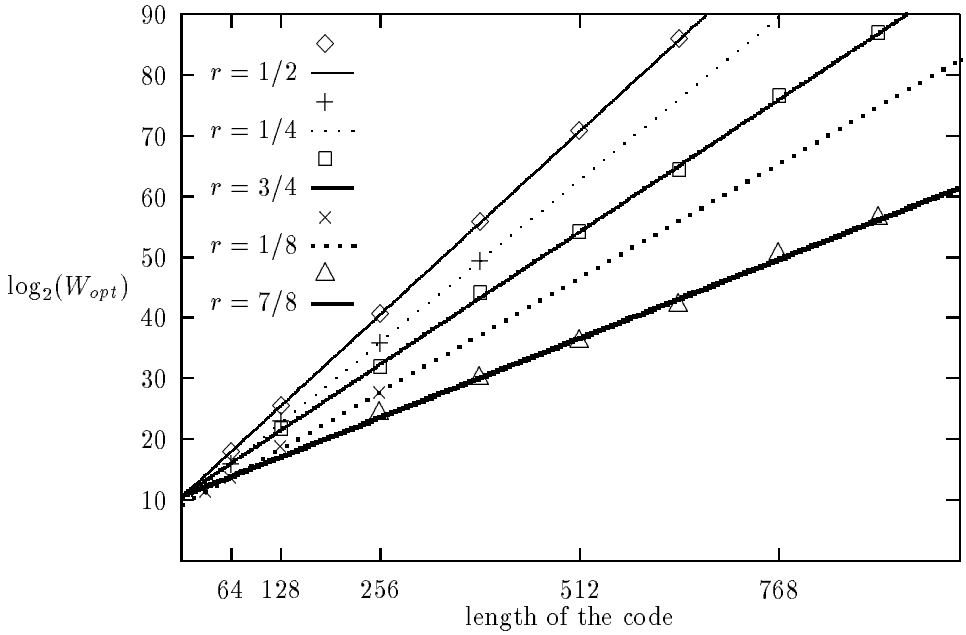

Figure 4: Evolution of the theoretical work factor with optimized parameters for finding a minimum-weight word in random $[n, nr]$-binary codes

n	k	δ	d	argument	in	n	k	δ	d		argument	in
511	502	3	3	RM(7)	[7]	511	241	73	73		PS(7,1)	[13]
	493	5	5	RM(7)	[7]		238	75	≥ 75		-	-
	484	7	7	RM(6)	[7]		229	77	≥ 77		-	-
	475	9	9	ES	[1]		220	79	79		ID	[1]
	466	11	11	RM(6)	[7]		211	83	83		ID	[1]
	457	13	13	S	[6]		202	85	≥ 85		-	-
	448	15	15	RM(15)	[7]		193	87	87		**	**
	439	17	17	ES	[1]		184	91	91		ID	[1]
	430	19	19	ID	[1]		175	93	95	#	4-DI	[8]
	421	21	21	PS(73,3)	[13]		166	95	95		RM(3)	[7]
	412	23	23	RM(5)	[7]		157	103	103		ID	[1]
	403	25	25	S	[6]		148	107	≥ 107		-	-
	394	27	27	RM(5)	[7]		139	109	111	#	4-DI	[8]
	385	29	29	**	**		130	111	111		RM(3)	[7]
	376	31	31	RM(4)	[7]		121	117	119	#	4-DI	[8]
	367	35	35	PS(73,5)	[13]		112	119	119		RM(3)	[7]
	358	37	37	**	**		103	123	127	##	NI	[1]
	349	39	39	ID	[1]		94	125	127	#	4-DI	[8]
	340	41	41	**	**		85	127	127		RM(2)	[7]
	331	43	43	**	**		76	171	171		ES	[1]
	322	45	45	ID	[1]		67	175	175		ES	[1]
	313	47	47	RM(4)	[7]		58	183	183		ES	[1]
	304	51	≥ 51	-	-		49	187	187		ES	[1]
	295	53	53	NI	[1]		40	191	191		RM(2)	[7]
	286	55	55	RM(4)	[7]		31	219	219		PS(7,3)	[13]
	277	57	57	ID	[1]		28	223	223		RM(2)	[7]
	268	59	≥ 59	-	-		19	239	239		RM(2)	[7]
	259	61	≥ 61	-	-		10	255	255		RM(1)	[7]
	250	63	63	RM(3)	[7]							

\# $d = \delta + 2$
\#\# $d = \delta + 4$

** new result
ES exhaustive search
ID idempotent
NI Newton's identities
RM(s) intersection with the shortened sth-order Reed-Muller code
S code shortening
PS(n_2, δ_2) product subcode: $n = n_1 n_2$ and $\delta = n_1 \delta_2$
where BCH(n_2, δ_2) has minimum distance equal to δ_2
4-DI 4-divisibility of RM(4)
since $B(511, \delta)$ is included in the punctured Reed-Muller code of order 4 for all $\delta > 85$

Table 4: BCH codes of length 511

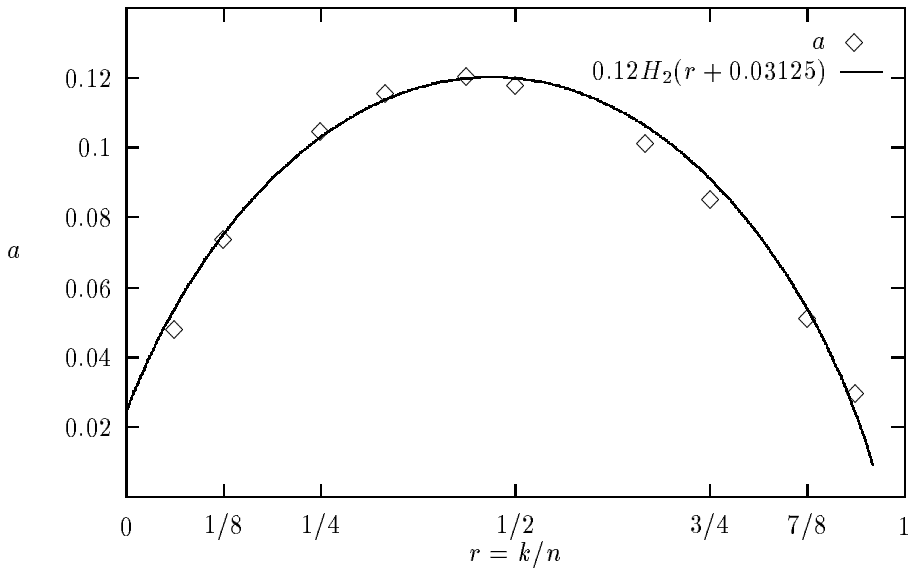

Figure 5: Evolution of coefficient c vs. $r = k/n$