Product of linear codes

Nicolas Sendrier

UNITÉ DE RECHERCHE
INRIA-ROCQUENCOURT

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P. 105
78153 Le Chesnay Cedex
France
Tél.:(1) 39 63 55 11

Rapports de Recherche

N° 1286

Programme 1
Programmation, Calcul Symbolique
et Intelligence Artificielle

PRODUCT OF LINEAR CODES

Nicolas SENDRIER

Octobre 1990

Product of linear codes

—

Produit de codes linéaires

Nicolas Sendrier *

INRIA
Domaine de Voluceau, Rocquencourt, B.P. 105
78153 LE CHESNAY Cedex - FRANCE

Abstract

The study of product codes by simulation gives some excellent results, for the code $RS(14,7) \otimes RS(14,7)$, we were able to correct 120 erasures or 50 errors with a very high chance of succes, and up to 147 erasures or 75 errors in some favourable cases. This code however only has a minimum distance of 64. It appears, so, that the minimum distance is not the proper criterion to judge the goodness of product codes.

We introduce the notion of *correcting capability* of an algorithm, this number has the dimension of a distance, and represents the minimum distance of an imaginary code of same rate and length with an algorithm able to correct up to the minimum distance but never beyond, and which has the same performances. Using this measure, we find for the code $RS(14,7) \otimes RS(14,7)$, a *correcting capability* for erasures of 134, and for errors of 123.

These figures are surprisingly high, and we will try to explain them by studying the weight distribution. We are able to describe this distribution for the smaller weights. We are also able to count the

*Travail de recherche effectué sous l'égide de la DRET

smaller uncorrectable patterns. These figures are small: for the code $RS(14,7) \otimes RS(14,7)$, the proportion of words of weight smaller than 72 is $\approx 10^{-44}$.

At last we can say a few words about the decoding complexity. This complexity is low and gives us another argument in favour of product codes.

As a conclusion, we can say that product codes are a good way to construct long codes with a high level of redundancy, that these codes have a low complexity, and that in spite of their poor minimum distance, they have good decoding performances.

Résumé

L'étude des codes produits par simulation donne des résultats excellents, pour le code $RS(14,7) \otimes RS(14,7)$, nous sommes capable de corriger 120 effacements ou 50 erreurs avec une très grande probabilité de succes, et jusqu'à 147 effacements ou 75 erreurs dans certains cas. Ce code n'a pourtant qu'une distance minimum de 64. La distance minimum semble donc ne pas être le meilleur critère d'évaluation des codes produits.

Nous introduisons la notion de *pouvoir de correction* d'un algorithme, ce nombre a la dimension d'une distance, et représente la distance minimum d'un code fictif de même taux et de même longueur ayant un algorithme de decodage corrigeant jusqu'à la distance minimum, mais jamais au delà, et qui aurait les mêmes performances. Nous trouvons pour le code $RS(14,7) \otimes RS(14,7)$, un *pouvoir de correction* egal à 134 pour les effacements et 123 pour les erreurs.

Ces chiffres sont etonnamment élevés, et nous essayons d'en donner une explication par l'étude de la distribution des poids. Nous sommes capable de décrire cette distribution pour les poids les plus faibles. Les valeurs que nous obtenons sont très petites: pour le code $RS(14,7) \otimes RS(14,7)$, la proportion de mots de poids inférieur à 72 est $\approx 10^{-44}$.

Enfin nous disons quelques mots sur la complexité de décodage. Cette complexité est faible, et nous donne un argument supplémentaire en faveur des codes produits.

En conclusion, nous pouvons dire que les codes produits nous donnent un bon shéma de construction de codes longs à haut niveau de redondance, que ces codes ont une faible complexité de décodage, et que malgré une distance minimum faible, ils ont un très bon niveau de performance.

Introduction

In order to obtain codes with high level of correction, we will need codes of great length. The main drawback of the known classes of block codes (BCH, Reed-Solomon, ...) is that their decoding complexity is very high for great lengths. A solution to avoid this inconvenience is to combine some short codes together. Such a scheme would be for instance a concatenated scheme, the code recommended by the European Space Agency is a concatenated code. An other solution can be the product codes.

Product codes are quite well known in coding theory [1, pages 568-575], but they usualy are not considered as good block codes, because they are poor in terms of minimum distance, so very few people did study their decoding properties in detail, though some good decoding algorithm are known, see [2].

In a first part of this work we evaluate the performances of the product codes. The simulations we ran showed that even the simplest decoding algorithm was able to correct far beyond the minimum distance. We develloped some tools that would permit us to built a more acute criterion to judge the goodness or the badness of a code.

In a second part we try by a more theoretical study of the weight distribution and of the correctable error patterns to find some hints on how to construct good product codes, and why our simulations were much better than what we expected.

Part I
Simulating product codes

1 Definitions

Let C_1 and C_2 be two linear codes over $GF(q)$:

$$C_1(q; n_1, k_1, d_1),$$

$$C_2(q; n_2, k_2, d_2).$$

1.1 Product code

Definition 1 *The product code $C_1 \otimes C_2$ of C_1 and C_2 is the set of all matrices M of size $n_2 \times n_1$ such that:*

- *every row of M is a codeword of C_1,*
- *every column of M is a codeword of C_2.*

This defines a linear code over $GF(q)$ of length $N = n_1 n_2$.

1.2 A very simple coding scheme

Let C_1 and C_2 be systematic, and let

$$G_i = \left(\; I_{k_i} \; \middle| \; P_i \; \right), \quad i = 1, 2$$

be the generating matrix of C_i. X is any $k_2 \times k_1$ matrix over $GF(q)$. Then the matrix:

$$\begin{pmatrix} X & XP_1 \\ {}^tP_2X & {}^tP_2XP_1 \end{pmatrix}$$

is a codeword of $C_1 \otimes C_2$. Furthermore, any word of the product code can obviously be produced that way, thus the dimension of the product code is $K = k_1 k_2$.

1.3 The minimum distance is $D = d_1 d_2$

Proposition 1 *$C_1 \otimes C_2$ is a linear code over $GF(q)$, with parameters:*

$$N = n_1 n_2, \quad K = k_1 k_2, \quad D = d_1 d_2.$$

proof:

1. any word of $C_1 \otimes C_2$ has at least one non-zero row, then at least d_1 non-zero columns, and each of these columns have a weight of at least d_2, so $D \geq d_1 d_2$.

2. and $D \leq d_1 d_2$ because there exist words of weight $d_1 d_2$, let:

$$\bar{x}_1 = (x_{11}, x_{12}, \ldots, x_{1n_1}) \in C_1,$$
$$\bar{x}_2 = (x_{21}, x_{22}, \ldots, x_{2n_2}) \in C_2.$$

The following matrix M is a codeword of $C_1 \otimes C_2$

$$M = \begin{pmatrix} x_{11}{}^t\bar{x}_2 & x_{12}{}^t\bar{x}_2 & \ldots & x_{1n_1}{}^t\bar{x}_2 \end{pmatrix} = \begin{pmatrix} x_{21}\bar{x}_1 \\ x_{22}\bar{x}_1 \\ \vdots \\ x_{2n_2}\bar{x}_1 \end{pmatrix} \in C_1 \otimes C_2,$$

and the weight of M is:

$$w(M) = w(\bar{x}_1) w(\bar{x}_2)$$

so, if we choose $w(\bar{x}_1) = d_1$ and $w(\bar{x}_2) = d_2$, then

$$w(M) = d_1 d_2.$$

□

2 The decoding algorithm

2.1 The basic algorithm

We assume that we have a decoding algorithm for both C_1 and C_2. Decoding a row or a column, consists in:

- correct the row or column in the matrix-codeword if correction is possible,
- do nothing in all other cases.

We can then describe a decoding algorithm as follow:

Step 1. decode successively each row then each column

Step 2. if the resulting matrix is a codeword $\rightarrow$ **SUCCES**

Step 3. if the resulting matrix has already been obtained $\rightarrow$ **FAILURE**

Step 4. $\rightarrow$ **Step 1.**

Of course, Step 3 needs to be different in practice. In our simulations, we limit the number of row-column decoding.

2.2 Erasure decoding

The model we will use is the memoryless erasure channel: every symbol transmitted has the probability p to be erased.

2.2.1 What are we going to compute?

Definition 2 *We consider a code of length N,*

1. *Let e_i, $i = 0 \ldots N$, be the ratio of erasure patterns of weight i that can be corrected:*

$$e_i = \frac{S_i}{\left(\begin{array}{c} N \\ i \end{array} \right)}, \tag{1}$$

where S_i is the number of correctable patterns of weight i.

2. *if the erasure probability of each symbol is p, then the probability for a codeword to be corrected is:*

$$P_{cor}(p) = \sum_{i=0}^{N} p^i (1-p)^{N-i} S_i, \tag{2}$$

$$P_{cor}(p) = \sum_{i=0}^{N} \left(\begin{array}{c} N \\ i \end{array} \right) p^i (1-p)^{N-i} e_i, \tag{3}$$

3. *and we define the probability of decoding failure as:*

$$P_{fail}(p) = 1 - P_{cor}(p) = \sum_{i=0}^{N} \binom{N}{i} p^i (1-p)^{N-i}(1-e_i). \qquad (4)$$

We are going to compute by simulation the values of the e_i. For each i:

1. we generate a great number M of random erasure patterns of weight i,
2. we try to decode each of them, let's say $\tilde{S}_i$ of the M are decoded,
3. this gives us an evaluation of e_i:

$$\tilde{e}_i = \frac{\tilde{S}_i}{M} \approx e_i. \qquad (5)$$

We won't need to run the programme for each weight (obviously $e_0 = 1$ and $e_N = 0$), the following proposition will limit with precision the field of investigation for a product code. We consider the codes defined in section 1.

Proposition 2 *We assume that we have for C_1 (resp. C_2), a decoding algorithm able to correct $d_1 - 1$ (resp. $d_2 - 1$) erasures but never more, then, using the algorithm described above for the product code $C_1 \otimes C_2$, we have the following properties for the ratio of correctable patterns:*

$$0 \leq i < D, \quad e_i = 1, \qquad (6)$$
$$D \leq i \leq U, \quad 0 < e_i < 1, \qquad (7)$$
$$U < i \leq N, \quad e_i = 0, \qquad (8)$$

where $U = N - (n_1 - d_1 + 1)(n_2 - d_2 + 1)$.

proof:

1. $i < D$, we first correct the rows, there is at most $d_2 - 1$ uncorrectables rows (or we would have $i \geq d_1 d_2 = D$), then all the columns have at most $d_2 - 1$ erasures and can be decoded.
2. $i = D$, an erasure pattern of weight d_1 cannot be decoded by the row decoder, the same yields for columns. So an erasure pattern whose support is a $d_2 \times d_1$ submatrix is uncorrectable.

3. $i = U$, let's consider the following pattern:

   - the first $d_2 - 1$ rows are erasures,
   - the first $d_1 - 1$ columns are erasures,

   this pattern has weight $n_1 d_2 + n_2 d_1 - d_1 d_2 = N - (n_1 - d_1 + 1)(n_2 - d_2 + 1) = U$ and is correctable.

4. $i > U$,

   (a) if C_1 and C_2 are MDS, then $U = N - K$ and if $i > N - K$ we have less than K remaining symbols, which cannot be enough for a code of dimension K.

   (b) from the assumptions made on the decoding algorithms of C_1 and C_2, the performances of the decoder is the same than in the MDS case.

□

Example 1. We ran our simulation programme this way for the product-code $RS(14,7) \otimes RS(14,7)$, where $RS(14,7)$ is the Reed-Solomon code of length 14 and dimension 7 over $GF(16)$, this product-code has parameters:

$$(q = 16; N = 196, K = 49, D = 64).$$

For each value of i between 64 and 147, 4 millions of random patterns were generated, and for $i < 120$ we didn't have any decoding failure. This was unexpected since the minimum distance of the code is only 64! (cf. Table 1)

Using the e_i we are able to compute the values of the probability of decoding failure for a given erasure probability p. These results are given in Table 2.

i	e_i	i	e_i	i	e_i
< 120	1.000000	129	0.989830	139	0.344820
120	0.999998	130	0.980354	140	0.240351
121	0.999995	131	0.964168	141	0.153589
122	0.999980	132	0.938254	142	0.088032
123	0.999929	133	0.899242	143	0.044455
124	0.999846	134	0.844023	144	0.019035
125	0.999582	135	0.770268	145	0.006589
126	0.998985	136	0.678731	146	0.001676
127	0.997665	137	0.572586	147	0.000249
128	0.994961	138	0.458300	> 147	0

Table 1: $RS(14,7) \otimes RS(14,7)$, **erasures correction.**

p	P_{fail}	p	P_{fail}	p	P_{fail}
0,45	$1{,}97.10^{-10}$	0,54	$0{,}24.10^{-4}$	0,63	0,0321
0,46	$1{,}01.10^{-9}$	0,55	$0{,}68.10^{-4}$	0,64	0,0549
0,47	$0{,}44.10^{-8}$	0,56	$1{,}77.10^{-4}$	0,65	0,0893
0,48	$1{,}83.10^{-8}$	0,57	$0{,}43.10^{-3}$	0,66	0,1381
0,49	$0{,}70.10^{-7}$	0,58	$1{,}01.10^{-3}$	0,67	0,2032
0,50	$0{,}25.10^{-6}$	0,59	$0{,}22.10^{-2}$	0,68	0,2847
0,51	$0{,}87.10^{-6}$	0,60	$0{,}47.10^{-2}$	0,69	0,3803
0,52	$0{,}28.10^{-6}$	0,61	$0{,}94.10^{-2}$	0,70	0,4853
0,53	$0{,}86.10^{-5}$	0,62	$1{,}78.10^{-2}$		

Table 2: $RS(14,7) \otimes RS(14,7)$, **probability of decoding failure**, p is the probability for each symbol to be erased.

2.2.2 How to evaluate decoding beyond the minimum distance?

We are given the code:

$$C(q; n, k, d),$$

with a decoding algorithm $\mathcal{A}$ able to correct erasures beyond the minimum distance, let e_i be the ratio of correctable erasure patterns of weight i. The probability of correction of this algorithm for an erasure probability p is:

$$P_{cor}(p) = \sum_{i=0}^{n} \binom{n}{i} p^i (1-p)^{n-i} e_i. \tag{9}$$

Let's consider now the following code:

$$C^*(q; n, k, d^*),$$

with a decoding algorithm able to correct $d^* - 1$ erasures or less in any case but no more in any case. The probability of correction of this algorithm for an erasure probability p is:

$$P_{cor}^{(d^*)}(p) = \sum_{i=0}^{d^*-1} \binom{n}{i} p^i (1-p)^{n-i}. \tag{10}$$

Definition 3 *I will say that the "***correcting capability** *of C for the algorithm $\mathcal{A}$ and for the erasure probability p" is the only integer d^* satisfying:*

$$P_{cor}^{(d^*)}(p) \leq P_{cor}(p) < P_{cor}^{(d^*+1)}(p). \tag{11}$$

This definition means that if d^* is the *correcting capability* of C for the erasure probability p, then the decoding algorithm $\mathcal{A}$ of the code C will behave at least as well as a minimum distance decoding algorithm of the code C^* for an erasure probability of p.

Example 1. We computed values of d^* for the code $RS(14,7) \otimes RS(14,7)$, for an erasure probability p between 45% and 70%. Results are given in Table 3.

If we consider a probability of decoding failure of 10^{-5} as small enough, then, from Table 3, we can say that this product code is as good in a way as a code of minimum distance $d^* = 134$ (for erasure correction).

p	r.u.w.	d^*	p	r.u.w.	d^*	p	r.u.w.	d^*
0,45	$1{,}97.10^{-10}$	132	0,54	$0{,}24.10^{-4}$	134	0,63	0,0321	136
0,46	$1{,}01.10^{-9}$	132	0,55	$0{,}68.10^{-4}$	134	0,64	0,0549	136
0,47	$0{,}44.10^{-8}$	132	0,56	$1{,}77.10^{-4}$	134	0,65	0,0893	136
0,48	$1{,}83.10^{-8}$	132	0,57	$0{,}43.10^{-3}$	135	0,66	0,1381	137
0,49	$0{,}70.10^{-7}$	133	0,58	$1{,}01.10^{-3}$	135	0,67	0,2032	137
0,50	$0{,}25.10^{-6}$	133	0,59	$0{,}22.10^{-2}$	135	0,68	0,2847	137
0,51	$0{,}87.10^{-6}$	133	0,60	$0{,}47.10^{-2}$	135	0,69	0,3803	137
0,52	$0{,}28.10^{-5}$	133	0,61	$0{,}94.10^{-2}$	135	0,70	0,4853	138
0,53	$0{,}86.10^{-5}$	134	0,62	$1{,}78.10^{-2}$	136			

Table 3: $RS(14,7) \otimes RS(14,7)$, **correcting capability** for erasures.

Of course it obviously appears that this way of measuring the performances is unfair, most code, and especialy codes of great length can be corrected beyond the minimum distance. We can then see the *correcting capability* only as a scale that could be use to mesure the performances of any code.

2.3 Error correction

2.3.1 The simulation

We will now consider an error channel, the model we will choose for simulation will be the q-ary symmetric channel with a total error probability for each symbol equal to p. As for erasures we will have the following definitions:

Definition 4 *We consider a code of length N,*

1. *let e_i, $i = 0 \ldots N$, be the ratio of error patterns of weight i that can be corrected:*

$$e_i = \frac{S_i}{\binom{N}{i}}, \tag{12}$$

where S_i is the number of correctable patterns of weight i.

2. *if the error probability of each symbol is p, then the probability for a codeword to be corrected is:*

$$P_{cor}(p) = \sum_{i=0}^{N} p^i (1-p)^{N-i} S_i, \tag{13}$$

$$P_{cor}(p) = \sum_{i=0}^{N} \binom{N}{i} p^i (1-p)^{N-i} e_i, \tag{14}$$

3. *and we define the probability of decoding failure as:*

$$P_{fail}(p) = 1 - P_{cor}(p) = \sum_{i=0}^{N} \binom{N}{i} p^i (1-p)^{N-i} (1-e_i). \tag{15}$$

We consider the product code:

$$C(q; N, K, D) = C_1(q; n_1, k_1, d_1) \otimes C_2(q; n_2, k_2, d_2),$$

we will denote $t_1 = \lfloor \frac{d_1-1}{2} \rfloor$ and $t_2 = \lfloor \frac{d_2-1}{2} \rfloor$ the number of errors that C_1 and C_2 can respectively correct.

Proposition 3 *We assume that we have for C_1 (resp. C_2), an algorithm able to correct t_1 (resp. t_2) errors but never more, then, using the algorithm we described for the product code $C_1 \otimes C_2$, we have the following properties for the ratio of correctable patterns:*

$$0 \leq i < L, \quad e_i = 1, \tag{16}$$

$$L \leq i \leq U, \quad 0 < e_i < 1, \tag{17}$$

$$U < i \leq N, \quad e_i = 0, \tag{18}$$

where

$$\begin{cases} U = N - (n_1 - t_1)(n_2 - t_2) \\ L = (t_1 + 1)(t_2 + 1) \end{cases}$$

proof: The proof is similar to the proof for erasures patterns, except for the case $i > U$, which is clear if we accept Proposition 10 of section 5. □

remark: L is the smallest weight of an uncorrectable pattern, and $L \approx \frac{D}{4}$ instead of $\frac{D}{2}$, this means that the decoding algorithm we use is not able to correct up to half the minimum distance in all the cases, though it is able to correct much farther in many case as we will show in our example.

i	e_i	i	e_i	i	e_i
<50	1.00000	59	0.98937	69	0.38256
50	0.99994	60	0.98252	70	0.25701
51	0.99983	61	0.97046	71	0.15092
52	0.99978	62	0.95159	72	0.07540
53	0.99943	63	0.92268	73	0.02980
54	0.99936	64	0.87870	74	0.00710
55	0.99894	65	0.81776	75	0.00140
56	0.99781	66	0.73343	>75	0
57	0.99629	67	0.63188		
58	0.99372	68	0.51199		

Table 4: $RS(14,7) \otimes RS(14,7)$, **errors correction.**

Example 1. We ran our simulation programme for the code $RS(14,7) \otimes RS(14,7)$, and for errors correction. For each weight 100000 error patterns were generated. The results are given in Table 4.

2.3.2 The correcting capability for errors

As for erasure correction we will define a *correcting capability* for errors, let's consider the code:

$$C^*(q; n, k, d^*),$$

and let's suppose that we are given for this code an error correcting algorithm able to correct $t^* = \lfloor \frac{d^*-1}{2} \rfloor$ errors but never more, the probability of correction for such a code is:

$$P_{cor}^{(t^*)}(p) = \sum_{i=0}^{t^*} \binom{n}{i} p^i (1-p)^{n-i}. \tag{19}$$

Definition 5 *Let $C(q; n, k, d)$ be any linear code, $\mathcal{A}$ an error correcting algorithm for C. I will say that the* "**correcting capability** *of C for the*

p	P_{fail}	t^*	d^*	p	P_{fail}	t^*	d^*
0,15	$1{,}85.10^{-8}$	59	119	0,23	$0{,}75.10^{-3}$	64	129
0,16	$1{,}08.10^{-7}$	59	119	0,24	$1{,}89.10^{-3}$	64	129
0,17	$0{,}53.10^{-6}$	60	121	0,25	$0{,}44.10^{-2}$	64	129
0,18	$0{,}23.10^{-5}$	61	123	0,26	$0{,}96.10^{-2}$	65	131
0,19	$0{,}87.10^{-5}$	61	123	0,27	0,0194	65	131
0,20	$0{,}30.10^{-4}$	62	125	0,28	0,0367	65	131
0,21	$0{,}95.10^{-4}$	63	127	0,29	0,0646	66	133
0,22	$0{,}28.10^{-3}$	63	127	0,30	0,1066	66	133

Table 5: $RS(14,7) \otimes RS(14,7)$, **probability of decoding failure**, p is the probability for each symbol to be erroneous.

algorithm $\mathcal{A}$ and for the error probability p" is $d^* = 2t^* + 1$ where t^* is the only integer satisfying:

$$P_{cor}^{(t^*)}(p) \leq P_{cor}(p) < P_{cor}^{(t^*+1)}(p). \tag{20}$$

Example 1. The values of the *correcting capability* for error were computed for the code $RS(14,7) \otimes RS(14,7)$, if we consider a probability of failure of 10^{-5} as small enough, then this code is at least as good as a code of minimum distance $d^* = 123$ with a minimum distance algorithm. (cf. Table 5)

For the same code, and for erasure correction, we found $d^* = 134$ for the same failure probability, it appears so, that the algorithm is relatively more efficient for erasure correction than for error correction. An explanation of this could be that the minimun distance of the code $RS(14,7)$ is even ($d = 8$), and we use a decoding algorithm that only corrects 3 errors (instead of $3\frac{1}{2}$!).

Anyway in both cases the decoding performances of the code are much better than what they were expected to be.

Part II
Theoretical results

We will try to explain the previous results by some combinatoric means, there are two possible directions in which we can find a proof of the goodness of the product-codes:

- first by the study of the weight distribution, if we have very few codewords of small weight, then we could easily explain a *correcting capability* much higher than the minimum distance,
- we can also try to count how many error patterns of given weight can or cannot be corrected.

Those two fields can appear very close, but the former has to do with the code itself, as the latter has to do with the algorithm.

3 The weight distribution of a product code

$$C_1(q; n_1, k_1, d_1),$$

$$C_2(q; n_2, k_2, d_2),$$

$$C_1 \otimes C_2 = C(q; N, K, D).$$

3.1 The small weights

For a given error correcting code $\mathcal{C}$, we will denote by $A_i(\mathcal{C})$ the number of codewords of $\mathcal{C}$ of weight i.

Proposition 4 *The number of codewords of* $C = C_1 \otimes C_2$ *of weight* $d = d_1 d_2$ *is:*

$$A_D(C) = \frac{1}{q-1} A_{d_1}(C_1) A_{d_2}(C_2), \tag{21}$$

and there is no other weight before $D + min(d_1, d_2)$*:*

$$\forall w, D < w < D + min(d_1, d_2), A_w(C) = 0. \tag{22}$$

We will need for the proof the following easy Lemma:

Lemma 1 *Let C be a linear code over $GF(q)$ of minimum distance d, and let $\bar{x}, \bar{y}$ be two codewords of C of weight d, then:*

$$supp(\bar{x}) = supp(\bar{y}) \Leftrightarrow \bar{x} = \lambda \bar{y} \tag{23}$$

where $\lambda \in GF(q)\backslash\{0\}$, and supp(.) denotes the support.

proof: (of Proposition 4)

1. $w = D$

   (a) the only way to have a codeword of weight $D = d_1 d_2$, is to have a $d_2 \times d_1$ submatrix as support. (cf. proof of Proposition 1)

   (b) from the lemma, two words of minimum weight of C_1 (or C_2) which have the same support are proportional.

   from (a) and (b), we can claim that a codeword of $C_1 \otimes C_2$ of weight D has every row proportionnal (possibly zero), and every column proportionnal. So any minimum weight codeword of $C_1 \otimes C_2$ will look like:

$$M(\bar{x}_1, \bar{x}_2) = \begin{pmatrix} x_{11}{}^t\bar{x}_2 & x_{12}{}^t\bar{x}_2 & \dots & x_{1n_1}{}^t\bar{x}_2 \end{pmatrix} = \begin{pmatrix} x_{21}\bar{x}_1 \\ x_{22}\bar{x}_1 \\ \vdots \\ x_{2n_2}\bar{x}_1 \end{pmatrix} \tag{24}$$

   where,

$$\bar{x}_1 = (x_{11}, x_{12}, \dots, x_{1n_1}) \in C_1$$
$$\bar{x}_2 = (x_{21}, x_{22}, \dots, x_{2n_2}) \in C_2$$

   and furthermore $w(\bar{x}_1) = d_1$ and $w(\bar{x}_2) = d_2$. If we examine all the matrices $M(\bar{x}_1, \bar{x}_2)$ with $\bar{x}_1$ and $\bar{x}_2$ of minimum weight, then from the equivalence:

$$M(\bar{x}_1, \bar{x}_2) = M(\bar{x}'_1, \bar{x}'_2) \Leftrightarrow \begin{cases} \bar{x}_1 = \lambda \bar{x}'_1 \\ \lambda \bar{x}_2 = \bar{x}'_2 \\ \lambda \in GF(q)\backslash\{0\} \end{cases} \tag{25}$$

   we can say that each matrix of weight $d_1 d_2$ is obtained exactely $q - 1$ times. This proves equation 21.

2. $D < w < D + \min(d_1, d_2)$, a codeword of weight $> D$ must have either more than d_2 non-zero rows, either more than d_1 non-zero columns, thus the weight of such a codeword is greater or equal to $D + \min(d_1, d_2)$.

□

Corollary 1 *If C_1 and C_2 are MDS, then:*

$$A_d(C) = (q-1)\begin{pmatrix} n_1 \\ d_1 \end{pmatrix}\begin{pmatrix} n_2 \\ d_2 \end{pmatrix} \tag{26}$$

proof: The number of codewords of weight d of a MDS code $(q; n, k, d)$ is:

$$(q-1)\begin{pmatrix} n \\ d \end{pmatrix}.$$

□

Example 1. In our example $RS(14,7) \otimes RS(14,7)$, the number of codewords of weight 64 is: $15\begin{pmatrix} 14 \\ 8 \end{pmatrix}^2$, the cardinality of the code is 16^{49}, the proportion of word of weight 64 is then $\approx 10^{-51}$, and the next weight will be 72.

Proposition 5 *we have:*

1. *if $d_1 < d_2$*

$$A_{D+d_1}(C) = \frac{A_{d_1}(C_1)A_{d_2+1}(C_2)}{q-1} \tag{27}$$

2. *if $d_1 = d_2 = d$*

$$\begin{aligned} A_{D+d}(C) \;\leq\; & (q-1)(d+1)!\begin{pmatrix} n_1 \\ d+1 \end{pmatrix}\begin{pmatrix} n_2 \\ d+1 \end{pmatrix} \\ & + \frac{A_{d+1}(C_1)A_d(C_2) + A_d(C_1)A_{d+1}(C_2)}{q-1} \end{aligned} \tag{28}$$

proof:

1. if $d_1 < d_2$, the only way to produce a codeword of weight $D + d_1 = d_1(d_2 + 1)$, is to have exactly $d_2 + 1$ non-zero rows of weight d_1, we can easily prove that in this case all the rows and all the columns are proportionnal, then the same argument as in Proposition 4 applies.

2. $d_1 = d_2 = d$, we want to count the number of codeword of weight $D + d = d(d + 1)$. Let $l_2 \times l_1$ be the smallest submatrix containing the support of such a codeword, we have $d\,l_2 \leq d(d+1)$ and $d\,l_1 \leq d(d+1)$, so $l_2 \leq d + 1$ and $l_1 \leq d + 1$. Thus any codeword of weight $D + d$ is one of the following:

   (a) the support of the codeword is a $(d + 1) \times d$ submatrix

   (b) the support of the codeword is a $d \times (d + 1)$ submatrix

   (c) the support of the codeword is included in a $(d + 1) \times (d + 1)$ submatrix with each row and each column of weight d.

   the case (c) will give us at most:

$$(d+1)! \begin{pmatrix} n_1 \\ d+1 \end{pmatrix} \begin{pmatrix} n_2 \\ d+1 \end{pmatrix}$$

   different support, so (from Lemma 1) at most:

$$(q-1)(d+1)! \begin{pmatrix} n_1 \\ d+1 \end{pmatrix} \begin{pmatrix} n_2 \\ d+1 \end{pmatrix}$$

   different codewords. Cases (a) and (b) will give us exactly:

$$\frac{A_{d+1}(C_1)A_d(C_2) + A_d(C_1)A_{d+1}(C_2)}{q-1}$$

   codewords.

□

Example 1. For the code $RS(14,7) \otimes RS(14,7)$, we have $A_{72} \leq 10^{-44}|C|$.

3.2 Power moments of the weight distribution

Proposition 6 *Let C be a code of length n over $GF(q)$, and let $d^\perp$ be its dual minimum distance.*

The first $d^\perp - 1$ power moments of the weight distribution of C are equal to the the power moments of the weight distribution of $GF(q)^n$.

proof: The Pless identities give us the binomial moments of the weight distribution of C:

$$\forall s, \sum_{i=s}^{n} \begin{pmatrix} i \\ s \end{pmatrix} \frac{A_i(C)}{q^k} = \frac{1}{q^s} \sum_{i=0}^{s} (-1)^i \begin{pmatrix} n-i \\ s-i \end{pmatrix} A_i(C^\perp) \tag{29}$$

so for $s < d^\perp$:

$$\sum_{i=s}^{n} \begin{pmatrix} i \\ s \end{pmatrix} \frac{A_i(C)}{q^k} = \frac{1}{q^s} \begin{pmatrix} n \\ s \end{pmatrix} \tag{30}$$

if $C = GF(q)^n$, then $C^\perp = \{0\}$, so (29) gives us:

$$\forall s, \sum_{i=s}^{n} \begin{pmatrix} i \\ s \end{pmatrix} \frac{A_i(GF(q)^n)}{q^n} = \frac{1}{q^s} \begin{pmatrix} n \\ s \end{pmatrix} \tag{31}$$

from equations 30 and 31 we have the result. □

Lemma 2 *Let C be a code of dual minimum distance $d^\perp$.*

Any set of $d^\perp - 1$ positions of C is significant, that is all $q^{d^\perp - 1}$ vector of $GF(q)^{d^\perp - 1}$ will appear in the restriction to these positions the same number of time when we run over C.

proof: This is a property of orthogonal arrays. □

Proposition 7 *The dual miminun distance of the code $C = C_1 \otimes C_2$ is $d^\perp = min(d_1^\perp, d_2^\perp)$.*

proof: let's suppose that $d_1^\perp \leq d_2^\perp$

1. $d^\perp \leq d_1^\perp$

$$\forall \bar{y_1} \in C_1^\perp, \begin{pmatrix} \bar{y_1} \\ 0 \\ \vdots \\ 0 \end{pmatrix} \in C^\perp \Rightarrow d^\perp \leq d_1^\perp$$

2. $d^{\perp} \geq d_1^{\perp}$, I will not show an explicit proof, but it can be built from Lemma 2.

□

Corollary 2 *The mean weight of* $C_1 \otimes C_2$ *is* $\frac{q-1}{q} n_1 n_2$.

Example 1. Let's apply this corollary to the code $RS(14,7) \otimes RS(14,7)$: the mean weight of its codewords is $183\frac{3}{4}$.

4 Erasure patterns

This section is related to the algorithm, we will try to count the erasure patterns of given weight which are correctable. p_i is the proportion of erasure patterns of weight i which are uncorrectable.

4.1 Erasure patterns of weight $d_1 d_2$

Proposition 8

$$p_{d_1 d_2} = \frac{\binom{n_1}{d_1}\binom{n_2}{d_2}}{\binom{n_1 n_2}{d_1 d_2}} \tag{32}$$

proof: If the erasure pattern is not a $d_2 \times d_1$ submatrix, then one row or one column at least can be corrected, and any pattern of weight $< d_1 d_2$ can be corrected (cf. Proposition 2). Thus the number of non-correctable patterns of weight $d_1 d_2$ is $\binom{n_1}{d_1}\binom{n_2}{d_2}$, the number of $d_2 \times d_1$ submatrices, among a total of $\binom{n_1 n_2}{d_1 d_2}$.

□

4.2 Erasure patterns of weight $< d_1 d_2 + \min(d_1, d_2)$

Proposition 9

$$\forall w, d_1 d_2 \leq w < d_1 d_2 + min(d_1, d_2)$$

$$p_w = \frac{\binom{n_1}{d_1}\binom{n_2}{d_2}\binom{n_1 n_2 - d_1 d_2}{w - d_1 d_2}}{\binom{n_1 n_2}{w}} \tag{33}$$

proof: If an erasure pattern of weight $< d_1 d_2 + \min(d_1, d_2)$ does not contain a $d_2 \times d_1$ submatrix, then at least one row or one column can be corrected, and the same argument apply on the resulting pattern, so this pattern is correctable. So the support contains a $d_2 \times d_1$ submatrix, and the $w - d_1 d_2$ remaining erasures can be anywhere. □

5 Error patterns

Let's consider:

$$C = C_1(q; n_1, k_1, d_1) \otimes C_2(q; n_2, k_2, d_2),$$

and,

$$C' = C'_1(q; n_1, k'_1, t_1 + 1) \otimes C'_2(q; n_2, k'_2, t_2 + 1),$$

$$\text{where } \begin{cases} t_1 = \lfloor \frac{d_1 - 1}{2} \rfloor \\ t_2 = \lfloor \frac{d_2 - 1}{2} \rfloor \end{cases}.$$

- We have a decoding algorithm $\mathcal{A}$ for C which is able to correct t_1 **errors** in every row and t_2 **errors** in every column.
- We have a decoding algorithm $\mathcal{A}'$ for C' which is able to correct t_1 **erasures** in every row and t_2 **erasures** in every column.

Proposition 10 *Let P be an* **error** *pattern for C, let P' be the* **erasure** *pattern for C' which has the same support (same positions in the codeword). If P can be corrected by the algorithm $\mathcal{A}$, then P' can be corrected by the algorithm $\mathcal{A}'$.* The reverse is not always true.

This means that the *correcting capability* of C' for **erasures** gives an upper bound to the *correcting capability* of C for **errors**. The question is: *how tigh is this bound?*

Example 1. We compared the error correcting algorithm for the code

$$RS(14,7,8) \otimes RS(14,7,8)$$

with the erasure correcting algorithm for the code

$$RS(14,11,4) \otimes RS(14,11,4).$$

The first one can correct 3 errors in every row and column, and the second one 3 erasures. We obtained the following performances:

- for an error probability of $p = 0,19$, the first code has a rate of uncorrected words for errors of (see Table 5):

$$0,87.10^{-5}$$

- for an erasure probability of $p = 0,19$, the second code has a rate of uncorrected words for erasures of:

$$0,83.10^{-5}$$

and the difference is not bigger for other values of p.

The main difference between error and erasure correction stands in the possibility of miscorrection. A pattern that would be correctable can become uncorrectable if an error is added. So it is likely that the product of codes with very low probability of miscorrection will have relatively better performances than, for instance, a product of perfect codes.

6 The complexity of the decoding algorithm

$$C_1 = C_2(n,k,d), \text{ and } C_1 \otimes C_2(N,K,D).$$

6.1 Erasures

Every row or column will be decoded at most once, so if $f(n,k,d)$ is the complexity for one row, then for the matrix it is smaller than:

$$2nf(n,k,d) = 2\sqrt{N}f(\sqrt{N},\sqrt{K},\sqrt{D})$$

and if $f(n,k,d) = O(n^a)$, then the complexity for the product code will be $O(N^{\frac{a+1}{2}})$.

The following result will give a little improvement of this figure:

Proposition 11 *Any correctable erasure pattern can be corrected in less than* $2n - d + 1$ *row or column decoding.*

proof: After decoding the $(n-d+1)^{th}$ row (resp. column if it occurs before), we decode only columns (resp. rows), since only at most $d-1$ uncorrected rows (resp. columns) remain, all the colums (resp. rows) can be corrected. So at most, $n + (n - d + 1) = 2n - d + 1$ decoding. □

The MDS case: $(k = n - d + 1)$ We take here for the component codes, Reed-Solomon codes, the best known algorithm has a complexity $f(n,k,d) \approx \lambda(n-k)^2$. Let $C'(n, \frac{k^2}{n})$ be a Reed-Solomon code of same rate over the same field. We have to take n as length of this code if we want to have the same field. The complexity for k^2 information symbols is for the product code:

$$\lambda(n+k)(n-k)^2 = \lambda n^3(1+R)(1-R)^2$$

and for C':

$$n\lambda(n - \frac{k^2}{n})^2 = \lambda n^3(1+R)^2(1-R)^2$$

So for the same number of symbol the product code is easier to decode, and has a much larger *correcting capability* than C'.

6.2 Errors

The complexity is going to be higher in this case since we are are going to make a lot of unusuccesful decoding.

But in practice, if we use an algorithm which tries to correct a row or a column only the first time or if it has been modified, then the number of decoding in practice is not very high, not much more than $2n$.

For $RS(14,7) \otimes RS(14,7)$, we have an average number of decoding of 26 ($2n = 28$) for an error probability of $p = 0,20$, and less for smaller values of p.

Part III
Annex

7 A possible improvement

After the original algorithm has failed, we can try the following one:

1. detect all the rows with problems, that is:
   - either error detection
   - either conflict with a column for one of its symbol
2. If the number of such rows doesn't exeed $d-1$, then replace them by erasures, if we obtain a codeword $\rightarrow$ SUCCES else $\rightarrow$ 3.
3. do 1. and 2. for columns, if we have a codeword $\rightarrow$ SUCCES else $\rightarrow$ FAIL.

   Example 1. for the product code $RS(14,7) \otimes RS(14,7)$ the improvement is the following: for $p = 0,19$, the probability of decoding failure is $P_{fail} = 0,26.10^{-5}$ instead of $0,87.10^{-5}$, and the *correcting capability* is $d^* = 63$ instead of 61.

8 Some other simulation results

We can make a few remarks on the results shown on Table 6.

1. The error *correcting capability* seems to be relatively lower when the component codes have a low redondancy.
2. In all three cases the *correcting capability* is much higher than the actual minimum distance.
3. The last example $RS(15,7,9) \otimes RS(15,7,9)$ is very surprising since for $P_{fail} = 10^{-4}$, the *correcting capability* is 179 and $N - K + 1 = 225 - 49 + 1 = 177$, so d^* is beyond the Singleton bound.

$RS(14,10,5) \otimes RS(14,10,5)$		
P_{fail}	d^*	
	errors	erasures
10^{-4}	57	84
10^{-5}	53	84
10^{-6}	49	83
$RS(14,8,7) \otimes RS(14,8,7)$		
P_{fail}	d^*	
	errors	erasures
10^{-4}	119	119
10^{-5}	115	118
10^{-6}	113	118
$RS(15,7,9) \otimes RS(15,7,9)$		
P_{fail}	d^*	
	errors	erasures
10^{-4}	179	161
10^{-5}	177	160
10^{-6}	175	160

Table 6: **correcting capability** of some other product of Reed-Solomon codes.

it is neccessary to say however, that the *correcting capability* is not a minimum distance, it is only a integer related to the correcting performances of a given decoding algorithm.

9 Conclusions

- A good way to construct long codes over relatively small alphabets
- An efficient decoding algorithm with low complexity
- The density of the half minimun distance radius spheres in the vector space must be low (in order to avoid miscorrections)
- Some very good practical examples (product of Reed-Solomon codes)

It is important also to precise that there exist an algorithm due to Reddy and Robinson which is able to correct any error pattern of weight smaller than $D/2$. Our algorithm only corrects patterns of weight up to $\approx D/4$ with probability 1.

However our goal here was not to show the optimal decoding algorithm, but to prove that the product-codes were better than what they appear to be. Our goal was also to point out that the minimum distance was not always the proper criterion.

At last, we must say that the concept of *correcting capability* must be handled carefully, because it is not a minimum distance, and so it is inadequat and unfair to compare the *correcting capability* of the decoding algorithm of a code to the minimum distance of another code. The *correcting capability* can be a fair mesure, but only if we use it as a scale to compare the performances of two codes, or more precisely, the performances of two algorithms.

Contents

References

[1] F.J. Macwilliams and N.J.A. Sloane, *The Theory of Error-Correcting Codes.* North-Holland 1986.

[2] S. M. Reddy and J. P. Robinson, *Random error and burst correction by iterated codes* IEEE Trans. Info. Theory, **18** (1972) 182-185.

Imprimé en France
par
. l'Institut National de Recherche en Informatique et en Automatique .

ISSN 0249-6399