

Codes correcteurs d'erreurs

par A. HOCQUENGHEM,

Professeur au Conservatoire des Arts et Métiers,
Ingénieur conseil à la S.E.A.

Généralisant un travail de Hamming, l'auteur construit des codes permettant de corriger k erreurs dans une transmission de digits binaires.

The paper is a generalization of Hamming's work. The author gives a coding system available to correct k errors in a transmission of binary digits.

Eine Arbeit von Hamming verallgemeinernd, entwickelt der Autor Codes die es ermöglichen bei Übertragung binärer Bits k Fehler zu korrigieren.

Обобщая работу Хамминга, автор предлагает коды, которые дают возможность исправлять k ошибок в передаче двоичных цифр.

1. Introduction.

Introduisons dans un système de transmission un mot, constitué par un nombre de n chiffres binaires :

$$a_1 a_2 \dots a_n$$

Le mot reçu peut différer du mot initial par un certain nombre d'erreurs (certains chiffres a_i étant altérés en $1 - a_i$). Pour essayer de détecter et de corriger ces erreurs, on n'utilise que m chiffres du mot comme support de l'information, les chiffres restant appelés chiffres de test devant servir à la vérification du mot après la transmission. Donner une loi de détermination de ces chiffres de test en fonction des m chiffres d'information de façon à pouvoir détecter — ou corriger — un nombre maximum k d'erreurs, c'est former un code détecteur — ou correcteur — de k erreurs.

L'exemple le plus simple est le code détecteur d'une erreur. Dans ce cas $m = n - 1$, et on choisit le chiffre de test de façon que le nombre total de chiffres 1 du mot soit pair. La vérification du mot consiste alors en un test de parité.

Hamming (*Bell System Technical Journal*, 1950) a donné la loi de formation d'un code correcteur d'une erreur. Le nombre de chiffres de test est l'entier N déterminé par les inégalités

$$(11) \quad \log_2(1 + n) \leq N < 1 + \log_2(1 + n)$$

Dans le cas général d'un code correcteur de k erreurs, le nombre de configurations d'erreurs possibles est :

$$H = 1 + C_n^1 + C_n^2 + \dots + C_n^k$$

Par suite le code le plus économique utiliserait un nombre de chiffres de test égal à l'entier immédiatement supérieur à $\log_2 H$. A part le code de Hamming, on n'a pu construire de tels codes. Ceux que nous proposons utilisent un nombre de chiffres de test égal à

$$n - m = kN$$

La différence

$$kN - \log_2 H$$

est de l'ordre de $\log_2(k!)$, donc assez faible pour que ces codes soient satisfaisants.

Après avoir défini un anneau dans lequel nous ferons nos calculs, nous exposerons le code de Hamming sous cette optique, puis les principes de formation des codes qui nous conduiront à une détermination quasi-expérimentale et à une détermination systématique de ces codes. Nous terminerons par un exemple de code correcteur de 2 erreurs.

2. Définition de l'anneau $\mathcal{C}$

Les éléments de l'anneau $\mathcal{C}$ sont les nombres entiers écrits en numération binaire.

A chaque élément de l'anneau $\mathcal{C}$ nous faisons correspondre un polynôme ayant comme coefficients les chiffres de l'élément. Le polynôme est alors défini sur le corps de caractéristique 2.

Toute opération sur les éléments de $\mathcal{C}$ sera faite sur les polynômes correspondants — au cours de ces opérations tout coefficient pair sera remplacé par 0, tout coefficient impair par 1. Le résultat sera un polynôme auquel correspondra un élément de l'anneau $\mathcal{C}$.

On a donc toutes les opérations habituelles sur les nombres entiers — afin d'éviter toute ambiguïté, toutes les expressions calculées selon ces règles seront suivies de l'indication ($\mathcal{C}$).

Exemples :

$$\text{Addition : } 101 + 111 = 10 \quad (\mathcal{C})$$

$$\text{Multiplication : } 101 \times 111 = 11.011 \quad (\mathcal{C})$$

$$\text{Puissance : } 101^2 = 10.001 \quad (\mathcal{C})$$

$$\text{Division : } 1.101 = 111 \times 10 + 11 \quad (\mathcal{C})$$

$$\text{En particulier : } p + p = 0, (p + q)^2 = p^2 + q^2 \quad (\mathcal{C})$$

Lorsque le polynôme sera irréductible sur le corps de caractéristique 2, nous dirons que le nombre correspondant est irréductible (il n'admet pas, dans l'anneau $\mathcal{C}$, d'autre diviseur que lui-même et l'unité).

On peut classer évidemment les nombres dans l'anneau $\mathcal{C}$ par ordre de grandeur, mais beaucoup plus important est le nombre de chiffres. On démontre que parmi les nombres ayant un nombre de chiffres donné, il existe toujours un nombre irréductible.

Etant donné un mot écrit en binaire

$$a_1 a_2 \dots a_n$$

nous attacherons à chaque indice i un nombre p_i de l'anneau $\mathcal{C}$ et au mot lui-même nous attacherons le nombre

$$T = a_1 p_1 + a_2 p_2 + \dots + a_n p_n \quad (\mathcal{C})$$

C'est la considération du nombre T qui, grâce à un choix convenable des nombres p_i , nous permettra de corriger les erreurs éventuelles.

3. Code de Hamming.

Nous retrouvons le code de Hamming en faisant

$$p_i = i$$

Les chiffres de test sont les chiffres du mot d'indices

$$1, 2, 2^2, \dots, 2^{N-1} \quad (N \text{ défini par les inégalités 11}).$$

L'information sera portée par les chiffres

$$a_3 a_5 a_6 a_7 a_9 \dots a_n$$

On détermine les chiffres de test par la condition

$$T = \sum p_i a_i = 0$$

condition qui s'écrit ici

$$a_1 + 2a_2 + 4a_4 + \dots + 2^{N-1} a_{2^{N-1}} = 3a_3 + 5a_5 + 6a_6 + \dots + na_n \quad (\mathcal{C})$$

Le second membre est un nombre binaire connu d'avance par les chiffres de test. L'égalité détermine donc parfaitement les valeurs des chiffres de test.

Si, après transmission, il n'y a pas d'erreur, on retrouvera $T = 0$.

S'il y a une erreur, portant par exemple sur le chiffre a_x , remplacé par $(1 - a_x)$, le nombre T prendra la valeur :

$$T = a_1 + 2a_2 + \dots + a(1 - a_n) + \dots + na_n = a \quad (\alpha)$$

La valeur de T sera l'indice du chiffre erroné.

S'il y a deux erreurs, portant sur les chiffres d'indice α et β , T prendra la valeur :

$$T = \alpha + \beta \neq 0 \quad (\alpha)$$

S'il y a plus de deux erreurs, T pourrait être nul. Le code obtenu est donc correcteur d'une erreur, détecteur de deux erreurs.

Il est commode, pour automatiser le contrôle, de supposer les nombres p disposés en matrice. Par exemple pour $n = 7$, on aura la matrice

$$\begin{array}{c} \text{Aux nombres} \\ 0010110 \end{array} \quad \text{et} \quad \begin{array}{c} 0010010 \end{array}$$

correspondront les matrices

$$\begin{array}{c} 0000110 \\ 0010010 \\ 0010100 \end{array} \quad \text{et} \quad \begin{array}{c} 0000010 \\ 0010010 \\ 0010000 \end{array}$$

Le nombre T s'obtient en faisant suivre chaque ligne de la matrice de son chiffre de parité (§ 1).

On obtient ici :

$$\begin{array}{c} 0 \\ 0 \\ 0 \end{array} \quad \text{et} \quad \begin{array}{c} 1 \\ 0 \\ 1 \end{array} = 5$$

Le premier nombre est correct, le 5^e chiffre du second nombre est faux.

4. Principe d'un code correcteur de k erreurs.

Voyons maintenant à quelles conditions doivent satisfaire les nombres p pour que le calcul de T permette de corriger k erreurs.

Nous supposons que les chiffres de test sont en nombre suffisant pour que, connaissant les chiffres d'information, on puisse réaliser la condition

$$(41) \quad T = \sum a_i p_i = 0 \quad (\alpha)$$

Si après transmission, les chiffres de rang

$$a_1, a_2, \dots, a_j \quad (j \leq k)$$

sont erronés, le nombre T calculé sur le mot déformé prendra la valeur :

$$T = p_{a_1} + p_{a_2} + \dots + p_{a_j} \quad (\alpha)$$

Il faut que le nombre ainsi trouvé soit caractéristique des rangs $a_1, a_2, \dots, a_j$, c'est-à-dire que :

$$p_{a_1} + p_{a_2} + \dots + p_{a_j} \neq p_{a_1} + p_{a_2} + \dots + p_{a_j'} \quad (\alpha)$$

lorsque

$$j \leq k, \quad j' \leq k$$

et les deux ensembles

$$(a_1, a_2, \dots, a_j), \quad (a'_1, a'_2, \dots, a'_j),$$

non identiques.

Cette condition peut encore s'écrire :

$$(42) \quad p_{\lambda_1} + p_{\lambda_2} + \dots + p_{\lambda_l} \neq 0 \quad (\alpha)$$

lorsque $l \leq 2k$

et les $\lambda_1, \lambda_2, \dots, \lambda_l$ étant tous différents.

On devra donc choisir les nombres p tels que l'addition, dans l'anneau $\mathcal{A}$, d'au plus $2k$ de ces nombres donne un résultat non nul.

Une fois déterminé un ensemble de n nombres p , il faudra choisir les chiffres de test. Il est commode pour cela de remplacer l'ensemble obtenu par un autre ensemble de n nombres mais contenant les puissances successives de 2 :

$$1, 2, 2^2, \dots, 2^{K-1}$$

K désignant le nombre de chiffres du plus grand nombre p obtenu.

Disposons pour cela les nombres p en une matrice M de n colonnes et K lignes ($K < n$), chaque nombre p étant donc représenté par une colonne

$$p_i = \begin{pmatrix} \omega_i^{K-1} \\ \omega_i^{K-2} \\ \vdots \\ \omega_i^1 \\ \omega_i^0 \end{pmatrix}$$

Si le rang de cette matrice (dans l'anneau $\mathcal{A}$) est $K' < K$, c'est que $K - K'$ lignes de cette matrice sont des combinaisons linéaires des K' lignes restantes. Si l'on supprime ces $K - K'$ lignes, on obtiendra une matrice M' de nombres p' qui vérifieront encore la condition (42).

Ceci étant, nous pourrions extraire de la matrice M' une matrice carrée Δ de K' lignes dont le déterminant calculé dans l'anneau $\mathcal{A}$ ne sera pas nul. On aura donc

$$\det \Delta = 1$$

puisque les seules valeurs possibles sont 0 ou 1. En multipliant la matrice M' par Δ^{-1} , on obtiendra la matrice M'' formée de nombres p'' tels que

$$p_i'' = \begin{pmatrix} \omega_i^{K'} \\ \omega_i^{K'-1} \\ \vdots \\ \omega_i^1 \\ \omega_i^0 \end{pmatrix} = \Delta^{-1} \begin{pmatrix} \omega_i^{K'} \\ \omega_i^{K'-1} \\ \vdots \\ \omega_i^1 \\ \omega_i^0 \end{pmatrix}$$

et par suite les nombres p'' vérifieront encore la condition (42). De plus, la matrice M'' contiendra à ce moment la matrice $\Delta^{-1} \times \Delta$, c'est-à-dire la matrice unité, donc l'ensemble des p'' contiendra les puissances successives de 2 :

$$1, 2, 2^2, \dots, 2^{K'-1}$$

Les indices correspondants seront pris comme chiffres de test et la condition (41) déterminera ces chiffres en fonction des chiffres d'information par égalité de deux nombres binaires de K' chiffres.

Tout le problème se ramène donc à construire des ensembles de nombres p satisfaisant à la condition (42).

5. Formation de proche en proche d'une suite de nombres p .

Prenons d'abord :

$$p_1 = 1, p_2 = 2, p_3 = 2^2, \dots, p_{2k} = 2^{2k-1}$$

puis :

$$p_{2k+1} = 2^{2k} - 1$$

$$p_{2k+2} = 2^{2k}$$

Ces nombres satisfont déjà aux conditions (42). Pour prolonger cette suite dans l'ordre des p croissants, supposons être arrivé au nombre p_i de l chiffres. Considérons l'ensemble des nombres p_i à p_l et de leurs sommes dans l'anneau $\mathcal{A}$ par groupes de 2, 3, ... ($2k - 1$). Tous les nombres obtenus ont au plus l chiffres.

S'il existe un nombre non contenu dans l'ensemble ainsi formé et compris entre p_i et 2^l , ce nombre sera pris pour valeur de p_{i+1} (s'il y a plusieurs nombres on choisira évidemment le plus petit). Sinon on prendra $p_{i+1} = 2^l$.

On peut ainsi continuer pas à pas jusqu'à l'obtention des n nombres p . Si p_n a K chiffres, les nombres

$$1, 2, 2^2, \dots, 2^{K-1}$$

seront inclus dans la suite des p . La suite sera donc directement utilisable pour former un code. Il restera à établir le tableau de correspondance entre les H valeurs de la somme

$$p_{a_1} + p_{a_2} + \dots + p_{a_j} \quad (\alpha) \quad (j \leq k)$$

et la valeur des indices $a_1, a_2, \dots, a_j$.

Le procédé ainsi défini est assez long à exploiter. Cependant, pour des valeurs raisonnables de n et k , il ne dépasse pas les possibilités d'une calculatrice de moyenne puissance.

La détermination a priori du nombre K de chiffres de test paraît assez difficile. Aussi allons-nous exposer un procédé plus systématique de recherche des nombres p .

6. Formation systématique des nombres p .

La théorie des congruences, si utilisée dans les preuves des opérations arithmétiques, va nous fournir un mode de calcul des nombres p . Désignons par q un nombre irréductible de $N + 1$ chiffres et par q' le reste de la division dans l'anneau $\mathcal{A}$ d'un nombre q par q . Le nombre q' aura au maximum N chiffres.

Nous poserons alors :

$$p_i = i + 2^{N(i^N)'} + 2^{2N(i^{2N})'} + \dots + 2^{(K-1)N(i^{2^{K-1}})'} \quad (\alpha) \\ (i = 1, 2, \dots, n)$$

c'est-à-dire que le nombre p_i est formé de la juxtaposition des restes successifs de la division par q des puissances impaires dans l'anneau $\mathcal{C}$ du nombre i . Nous allons montrer que ces nombres p_i satisfont à la condition (42).

En effet, supposons :

$$(61) \quad p_{\lambda_1} + p_{\lambda_2} + \dots + p_{\lambda_l} = 0 \quad (\mathcal{C}) \quad (l \leq 2k)$$

Cela entraînerait :

$$(62) \quad S_1 = S'_3 = S'_5 = \dots = S'_{2k-1} = 0$$

en posant :

$$S_i = (\lambda_1)^i + (\lambda_2)^i + \dots + (\lambda_l)^i \quad (\mathcal{C})$$

$$\text{et} \quad S'_i = S_i \pmod{q} \quad (\mathcal{C})$$

Or, si nous considérons le produit

$$\prod (\lambda_i + \lambda_j) \quad \begin{matrix} i = 2, 3, \dots, l \\ j = 1, 2, \dots, l-1 \end{matrix} \quad i > j \quad (\mathcal{C})$$

ce produit peut s'écrire sous forme d'un déterminant de Van der Monde dont le carré contiendra la ligne

$$S_1 \ S_2 \ S_3 \ \dots \ S_l$$

Comme $S_{2k} = S_l^2$, les conditions (62) entraînent

$$\prod (\lambda_i + \lambda_j) = 0 \pmod{q} \quad (\mathcal{C})$$

Donc un des facteurs, par exemple $\lambda_i + \lambda_j$, serait divisible par q . Comme la somme dans $\mathcal{C}$ des nombres $\lambda_i + \lambda_j$ a moins de chiffres que le nombre q , il en résulterait

$$\lambda_i + \lambda_j = 0 \quad \lambda_i = \lambda_j$$

Par suite l'hypothèse (61) ne peut être réalisée que si au moins deux indices étaient égaux.

Donc les nombres p que nous avons formés remplissent la condition (42) et peuvent servir à former un code correcteur de k erreurs. Naturellement on les transformera comme il est indiqué au § 4 pour former une suite contenant des puissances de 2. Dans le cas général, p_i comprenant kN chiffres, il y aura lieu d'utiliser kN chiffres de test.

7. Exemple.

Nous avons formé un code de 15 chiffres correcteur pour 2 erreurs. Ici $N = \log_2 16 = 4$, il y aura 8 chiffres de test.

En prenant $q = 19 = 10,011$, on calcule aisément les nombres p et la matrice M :

0	1	1	1	1	0	0	1	1	1	1	1	1	1	1
0	0	1	1	0	0	0	0	1	1	1	0	0	0	1
0	0	1	0	1	0	0	1	1	1	0	0	1	0	0
1	0	1	0	0	1	1	0	1	1	0	0	0	0	0
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Cette matrice M est de rang 8, car le déterminant formé avec les colonnes 1, 2, 4, 8, 6, 12, 7, 14 (choisies parce qu'elles présentent le plus de zéros) vaut 1.

La matrice Δ (§ 4) sera formée avec ces colonnes. En l'inversant on trouve la matrice :

$$\Delta^{-1} = \begin{vmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \end{vmatrix}$$

Le produit $\Delta^{-1}M$ (dans l'anneau $\mathcal{C}$) donne la matrice définitive :

1	0	1	0	0	0	0	0	0	0	0	0	1	0	1
0	1	0	0	1	0	0	0	1	1	1	0	0	0	1
0	0	1	1	0	0	0	0	1	1	1	0	0	0	1
0	0	1	0	1	0	0	1	1	1	0	0	1	0	0
0	0	0	0	1	1	0	0	0	1	1	0	1	0	1
0	0	0	0	0	0	1	0	0	1	1	0	1	0	0
0	0	0	0	1	0	0	0	1	0	1	1	0	0	0
0	0	1	0	1	0	0	0	0	1	0	0	1	1	1
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Les chiffres de rang

$$1, 2, 4, 6, 7, 8, 12, 14$$

serviront de chiffres de test, les 7 autres chiffres seront les supports de l'information.

Pour vérifier et corriger un mot on fera la somme

$$T = \sum p_i a_i \quad (\mathcal{C})$$

Si elle est nulle, il n'y aura pas eu d'altération du mot (ou plus de 4 erreurs). Si T n'est pas nulle, on pourra retrouver les chiffres faux (en admettant qu'il n'y en ait pas plus de 2) en utilisant la table suivante qui donne les valeurs possibles de T suivies entre parenthèses des rangs des chiffres faux.

1 (14)	—	2 (12)	—	3 (12,14)	—	4 (7)	—	5 (7,14)
6 (7,12)	—	8 (6)	—	9 (6,14)	—	10 (6,12)	—	12 (6,7)
15 (9,10)	—	16 (8)	—	17 (8,14)	—	18 (8,12)	—	19 (10,11)
20 (7,8)	—	24 (6,8)	—	27 (2,5)	—	28 (9,11)	—	29 (1,13)
32 (4)	—	33 (4,14)	—	34 (4,12)	—	36 (4,7)	—	40 (4,6)
41 (5,9)	—	42 (5,10)	—	44 (3,13)	—	46 (2,11)	—	48 (4,8)
49 (1,3)	—	50 (2,9)	—	53 (5,11)	—	61 (2,10)	—	64 (2)
65 (2,14)	—	66 (2,12)	—	68 (2,7)	—	72 (2,6)	—	75 (5,8)
77 (8,13)	—	78 (4,11)	—	80 (2,8)	—	82 (4,9)	—	83 (5,6)
85 (6,13)	—	88 (3,15)	—	89 (5,12)	—	90 (5,14)	—	91 (5)
93 (4,10)	—	96 (2,4)	—	98 (8,9)	—	102 (6,11)	—	103 (5,7)
105 (1,15)	—	106 (7,11)	—	108 (11,12)	—	110 (11)	—	111 (11,14)
112 (9,12)	—	113 (8,10)	—	114 (9)	—	115 (9,14)	—	116 (13,15)
117 (6,10)	—	118 (7,9)	—	121 (7,10)	—	122 (6,9)	—	123 (4,5)
124 (10,14)	—	125 (10)	—	126 (8,11)	—	127 (10,12)	—	128 (1)
129 (1,14)	—	130 (1,12)	—	132 (1,7)	—	135 (11,15)	—	136 (1,6)
144 (1,8)	—	145 (3,4)	—	152 (10,15)	—	153 (7,13)	—	155 (9,15)
156 (13,14)	—	157 (13)	—	159 (12,13)	—	160 (1,4)	—	161 (3,8)
169 (2,15)	—	176 (3,14)	—	177 (3)	—	178 (5,15)	—	179 (3,12)
181 (3,7)	—	185 (3,6)	—	189 (4,13)	—	192 (1,2)	—	195 (3,9)
198 (5,13)	—	201 (4,15)	—	204 (3,10)	—	219 (1,5)	—	221 (2,13)
223 (3,11)	—	224 (10,13)	—	225 (6,15)	—	232 (14,15)	—	233 (15)
234 (3,5)	—	235 (12,15)	—	237 (7,15)	—	238 (1,11)	—	239 (9,13)
241 (2,3)	—	242 (1,9)	—	243 (11,13)	—	249 (8,15)	—	253 (1,10)

On remarquera que le nombre T prend 121 valeurs possibles ($1 + C_{15}^1 + C_{15}^2$) et qu'on utilise un nombre de 8 chiffres pour l'écrire. Le code utilise un chiffre de test de plus qu'il n'est théoriquement indispensable, mais il n'est pas sûr qu'on puisse construire des codes n'ayant qu'un nombre de chiffres de test strictement égal à l'entier par excès de $\log_2 H$.