SEPTIEME COLLOQUE SUR LE TRAITEMENT DU
SIGNAL ET SES APPLICATIONS

NICE du 28 MAI au 2 JUIN 1979

DECODAGE CORRECTEUR ET TRANSFORMATION DE FOURIER SUR UN ENSEMBLE FINI
ERROR DECODING AND DISCRETE FOURIER TRANSFORM

Ph. GODLEWSKI, NGUYEN Chi Thanh et TAN Siv Tong

DEPARTEMENT SYSTEMES ET COMMUNICATIONS
ENST. 46 rue Barrault, 75634, PARIS - Cedex 13

RESUME

Les liens entre le décodage algébrique d'un code cyclique correcteur d'erreurs et la transformation de Fourier discrète (t.F.d.) sont présentés. Nous avons simulé sur microprocesseur 8080 un algorithme de décodage utilisant une t.F.d. rapide pour un code de Reed Solomon (16,10) sur le corps à 17 éléments.

SUMMARY

Links between algebraic decoding and Fourier transform are presented. We have simulated with a 8080 microprocessor, a decoding algorithm using a FFT for a (16,10) Reed-Solomon code over GF(17).

DÉCODAGE CORRECTEUR ET TRANSFORMATION DE FOURIER SUR UN ENSEMBLE FINI

1. Introduction

Le décodage algébrique des codes en blocs correcteurs d'erreurs est actuellement le seul procédé général, de complexité acceptable permettant la correction d'erreurs multiples. C'est dans le cas des codes de Reed Solomon (R.S) que ce décodage est le plus simple puisqu'il ne met alors en jeu que des calculs sur le corps à partir duquel le code est construit. Ces codes de R.S. s'adaptent à de larges classes de canaux et leur utilisation est souvent envisagée.

On considère généralement des codes sur $\mathbb{F}_2 = \{0,1\}$ ou sur une extension binaire $\mathbb{F}_{2^m}$: les éléments binaires à transmettre sont rassemblés par groupe de m, chaque groupe formant ainsi un symbole d'un alphabet à 2^m éléments, $\mathbb{F}_{2^m}$. Récemment, on s'est intéressé à des codes sur $\mathbb{F}_{2^m+1}$ où certaines opérations sont bien adaptées à l'utilisation de microprocesseurs. Nous examinons le problème du transcodage des données binaires qui se pose alors.

Nous étudions la réalisation d'un algorithme de décodage d'un code de R.S. défini sur un tel ensemble. Cet algorithme utilise pour le calcul du syndrome une transformation de Fourier rapide sur un corps fini (cf. [1], [2]). Le polynôme localisateur d'erreurs est déterminé grâce à l'algorithme d'Euclide ([3]). Le polynôme d'erreurs est évalué à partir d'une récurrence suivant la méthode exposée par Reed ([2]).

Nous considérons plus particulièrement le code (16, 10) corrigeant trois erreurs sur le corps $\mathbb{F}_{17}$. L'algorithme de décodage a été simulé sur un microprocesseur du type 8080. Les temps d'éxécution des différentes parties du programme sont donnés.

2. Notations et définitions

Soit F un alphabet à q éléments que l'on peut munir d'une structure de corps. Si q est un nombre premier, F est isomorphe à $\mathbb{Z}/q\mathbb{Z}$, l'ensemble des entiers modulo q. Plus généralement q est de la forme p^i ou p est un nombre premier. Tous les polynômes que nous considérons sont à coefficients dans F.

On étudie des codes en blocs sur F dont la longueur n divise q-1. Pour fixer les idées nous supposons dans ce qui suit :

$$n = q-1 \qquad (1)$$

ce qui correspond aux codes de R.S. au sens strict.

Soit $\mathcal{R} = F^n$ l'espace vectoriel de dimension n sur F. L'addition de deux éléments y et z de $\mathcal{R}$ s'écrit donc :

$y + z = (y_o + z_o , \ldots , y_{n-1} + z_{n-1})$

où $y = (y_o, \ldots , y_{n-1})$ et $z = (z_o , \ldots , z_{n-1})$; $y_i, z_i \in F$. On peut munir R d'une structure d'algèbre. Il faut, pour cela, définir une multiplication. Deux manières sont envisageables :

1°/ On identifie chaque élément y de $\mathcal{R}$ avec un polynôme y(U) de l'algèbre $\mathcal{A}_U$ des polynômes en U modulo U^n-1 : $\mathcal{A}_U \simeq F[U]/(U^n-1)$:

$$y = (y_o , \ldots y_{n-1}) \leftrightarrow y(U) = \sum_{i=o}^{n-1} y_i U^i \qquad (2)$$

Le produit de deux éléments y et z de $\mathcal{R}$ correspond ainsi au produit modulo U^n-1 de y(U) et z(U).

2°/ On définit le produit composante à composante

$y \times z = (y_o . z_o, \ldots , y_{n-1} . z_{n-1})$

appelé parfois produit d'Hadamard. On considère alors l'algèbre $(\mathcal{R}, +, \times)$.

Un code cyclique C sur F, de longueur n est un idéal de l'algèbre $\mathcal{A}_U$. Tout polynôme c(U) de C est alors multiple d'un polynôme générateur g(U) qui caractérise le code. Dans ce qui suit un mot du code C sera repéré soit par un n-uplet $c = (c_o, \ldots , c_{n-1})$ soit par un polynôme en U ou en X : c(U) ou c(X).

3. Codage et transformation de Fourier

On considère en codage algébrique une transformation du type Fourier discrète (t.F.d.), appelée transformation de Mattson Solomon. Elle n'est généralement pas utilisée dans des buts de calculs pratiques mais plutôt pour des considérations théoriques.

L'existence d'une telle transformation repose sur celle d'un élément s d'ordre n ($s = \exp(2\pi i/n)$ pour la t.F.d. habituelle). Dans le cas des codes de R.S. le corps F possède un élément d'ordre n et il n'est pas nécessaire de le plonger dans une extension. Le groupe multiplicatif de tout corps fini est en effet cyclique ainsi :

$$F^* \triangleq F-\{o\} = \{s, s^2 , \ldots , s^n = 1\}, \qquad (3)$$

les n = q-1 puissances successives de s génèrent tous les éléments non nuls de F.

Soit $(A_o , \ldots , A_{n-1})$ un élément de $\mathcal{R}$ que l'on identifie à un polynôme de $\mathcal{A}_X$: $A(X) = \Sigma_i A_i X^i$. La transformation discrète $\mathcal{Y}$ que nous considérons est

définie par :

$$A(X) \xrightarrow{\varphi} a(U) = \sum_{i=o}^{n-1} A(s^i)\, U^i, \qquad (4)$$

L'application φ est un isomorphisme d'algèbre et cela de deux manières :

$$\mathcal{A}_X \xrightarrow{\varphi} (\mathcal{R}, +, \times) \qquad (5)$$

$$(\mathcal{R}, +, \times) \xrightarrow{\varphi} \mathcal{A}_U \qquad (6)$$

on identifie en (5) (resp. en (6)) les polynômes en U (resp. en X) avec les éléments de l'algèbre $(\mathcal{R}, +, \times)$.

La transformation φ^{-1} est donnée par :

$$a(U) \xrightarrow{\varphi^{-1}} A(X) = \frac{1}{n} \sum_{i=o}^{n-1} a(s^{-i})\, X^i \quad . \qquad (7)$$

En considérant la dernière formule il devient clair que le code $C \subset \mathcal{A}_U$ dont tous les mots sont multiples du polynôme générateur :

$$g(U) = \prod_{i=o}^{d-1} (U - s^i) \qquad (8)$$

est l'image par φ de l'ensemble des polynômes de $\mathcal{A}_X$ de degré strictement inférieur à k = n-d+1. On montre facilement qu'un tel code est de distance minimale d.

Exemple : q = 17, n = 16, les puissances successives de 6 modulo 17 génèrent les éléments non nuls de $F = \mathbf{Z}/17\mathbf{Z}$: $F^* = \{6,2,-5,4,7,8,-3,-1,-6,-2,5,-4,-7,-8,3,1\}$

Les multiples modulo $X^{16}-1$ du polynôme $g(X) = \prod_{i=1}^{6} (x-6^i)$ forment un code sur F de longueur n = 16 avec k = 10 symboles d'information et d = 7.

4. Codage et conversion q-aire /binaire

Soit $A_o, A_1, \ldots, A_{k-1}$, k symboles d'information q-aires que l'on désire coder à l'aide d'un (n,k) code sur F. On forme le polynôme $A(X) = \Sigma_i A_i X^i$. Deux procédés de codage sont envisageables :

1°/ Codage non systématique. On effectue une t.F.d sur A. D'après une remarque du paragraphe précédent, $\varphi(A)$ appartient à un (n,k) code cyclique sur F. Cette méthode permet d'économiser au décodage une t.F.d.

2°/ Codage systématique. On utilise le procédé classique qui consiste à calculer le reste r(U) de la division de $U^{n-k} A(U)$ par g(U). On transmet $c(U) = X^{n-k} A(U) - r(U)$.

Dans des problèmes de transmission de données binaires ce dernier procédé permet une meilleur conversion q-aire binaire. Le nombre d'éléments binaires (e.b.) transmis par mot de code est respectivement

1°/ $n \lfloor \log_2 (q) + 1 \rfloor$

2°/ $k \lfloor \log_2 (q) \rfloor + (n-k) \lfloor \log_2 (q) + 1 \rfloor$.

On suppose que les bits d'information sont groupés par paquets de $\lfloor \log_2 (q) \rfloor$. Chacun de ces paquets est assimilé à un élément de F. Dans la seconde estimation, seuls les (n-k) symboles de redondance nécessitent chacun $\lfloor \log_2 (q) + 1 \rfloor$ e.b. pour leur conversion binaire.

Dans le cas $q = 2^m+1$, on s'intéresse au sous ensemble C' des mots de C dont les n composantes appartiennent à un alphabet F' à q-1 éléments. La cardinalité $|C'|$ de C' est le nombre W_n de mot de C de poids n, qui est connu (cf. polynôme énumérateur de poids d'un code "maximum séparable length", [4]) :

$$W_n = \sum_{j=o}^{k-1} (-1)^j \binom{n}{j} (q^{k-j}-1) \quad . \qquad (10)$$

On en déduit l'approximation :

$$|C'| = W_n \simeq (q-1)^k \left(\frac{q-1}{q}\right)^{n-k} \text{ pour } q >> 2.$$

Ce résultat montre que si on choisit les k symboles d'information dans F', les n-k symboles de redondance obtenus par codage systématique semblent tirés au hasard parmi l'alphabet F, en ce sens que la probabilité qu'ils appartiennent tous à F' est environ $((q-1)/q)^{n-k}$.

Si on souhaite pour des raisons de synchronisation ou de format que chaque symbole soit représenté par m e.b., on peut soit accepter que le mot codé contienne a priori des erreurs, soit diminuer le nombre de symboles binaires d'information.

Exemple : q = 17, n = 16, k = 10, chaque mot de l'ensemble C' est formé de 16 composantes appartenant à un alphabet F' à 16 éléments, et ne nécessite donc que 16 x 4 e.b. pour sa transcription binaire :

$|C'| \simeq 16^{10} \cdot 0,7$

$\log_2 |C'| = 39,48$

La probabilité que les 6 symboles de contrôle appartiennent à F' est 0,7; la probabilité qu'il y ait au plus un symbole n'appartenant pas à F' est 0,956. Si l'on ne désire pas commettre "d'erreurs" au codage il faut alors restreindre le nombre de bits d'information. On obtient ainsi un code dont les caractéristiques binaires sont $n_b = 64$, $k_b = 39$, qui corrige 3 paquets de longueur 4.

5. Décodage

Le décodage algébrique d'un code en blocs comprend trois phases principales :

1/ le calcul du syndrome.

2/ le calcul du polynôme localisateur d'erreurs.

3/ le calcul de l'amplitude de chaque erreur.

Un mot $A = (A_o , \dots , A_{n-1})$ d'un code C de distance minimale d est transmis sur un canal bruité. Chaque symbole A_i est affecté de l'erreur additive E_i. A la réception on dispose du mot B de $\mathcal{R}$. Avec une notation polynomiale on peut écrire :

$B(X) = A(X) + E(X)$

D'après les propriétés métriques du code, il est possible de déterminer d'une manière unique A(X) à partir de B(X) si au plus $t = \lfloor (d-1)/2 \rfloor$ coefficients de E sont non nuls.

Le code C est spécifié par le polynôme g(X) (cf. (8)). Tout mot de code est un multiple de g(X) et admet donc les même racines que lui :

$$(A(X) \in C) \iff (\forall_i, 1 \leqslant i < d \ , \ A(s^i) = o) \qquad (11)$$

On appelle syndrome l'ensemble des d-1 valeurs du polynôme reçu, B(X), aux points $X = s^i$, $1 \leqslant i < d$; d'après (11) :

$$\left.\begin{aligned} S_1 &\triangleq B(s) = E(s) \\ S_i &\triangleq B(s^i) = E(s^i) \\ S_{d-1} &\triangleq B(s^{d-1}) = E(s^{d-1}) \end{aligned}\right\} \qquad (12)$$

Il est clair que la transformée de B(X) par φ (ou par φ^{-1}) fournit les d-1 coefficients du syndrome (cf. (4)).

La seconde phase du décodage consiste à résoudre la congruence ("key equation") :

$$e(Z) \equiv \frac{Q(Z)}{\sigma(Z)} \quad \text{modulo } Z^d \qquad (13)$$

où les deux polynômes inconnus Q(Z) et $\sigma(Z)$ vérifient $\deg(\sigma) < \deg(Q)$, et d'autre part :

$$e(Z) = \sum_{i=1}^{n} Z^i \ e_i \quad , \text{ où } \quad e_i = E(s^i) \ .$$

Seuls les (d-1) premiers coefficients de e(Z) sont connus (syndrome) mais cela est suffisant pour caractériser ce polynôme modulo Z^d. L'algorithme de Berlekamp ([5]) ou l'algorithme d'Euclide ([3]) que nous avons choisi, permettent de déterminer le polynôme $\sigma(Z)$ de plus bas degré vérifiant (13).

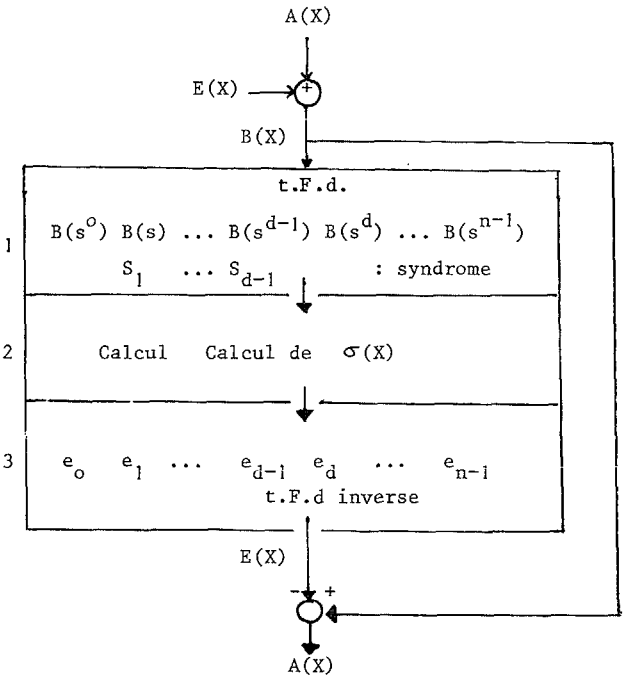
Enfin le polynôme erreur E(X) est évalué en deux temps. On détermine d'abord $e = (e_o , \dots , e_{n-1})$ à partir de $e_1 , \dots , e_{d-1}$ et d'une relation de récurrence spécifiée par $\sigma(Z) = \sum_{i} \sigma_i Z_i$

$$\sum_{j=1}^{t} \sigma_i \ e_{i+j} = o$$

Puis E(X) est calculé par t.F.d. inverse φ^{-1} et retranché de B.

Figure 1.

Schéma de principe du décodage :



6. Mise en oeuvre de l'algorithme

t.F.d. rapide.

Lorsque la longueur du code est une puissance de 2 un algorithme de t.F.d. rapide s'impose pour le calcul du syndrome. Cet algorithme est d'autant plus efficace que la multiplication par l'élément s d'ordre n (cf. § 3) est simple. Il est par exemple intéressant qu'une puissance "faible" de s soit 2.

Exemple : Dans le cas q = 17, n = 16, s = 6, on peut écrire, puisque $6^2 = 2$:

$$A(6^i) = \sum_{j \text{ pair}} A_j \ 6^j + \sum_{j \text{ impair}} \dots = A'(2^i) + 6^i A''(2^i)$$

A l'exception, peut-être, de la multiplication par 6^i, le calcul de $A(6^i)$ ne met donc en jeu que des additions et des multiplications par 2. Le treillis associé à la t.F.d. rapide est représenté en figure 2. Un exemple de transformée est donné.

Représentation des éléments du corps fini.

Il existe plusieurs manières de représenter les

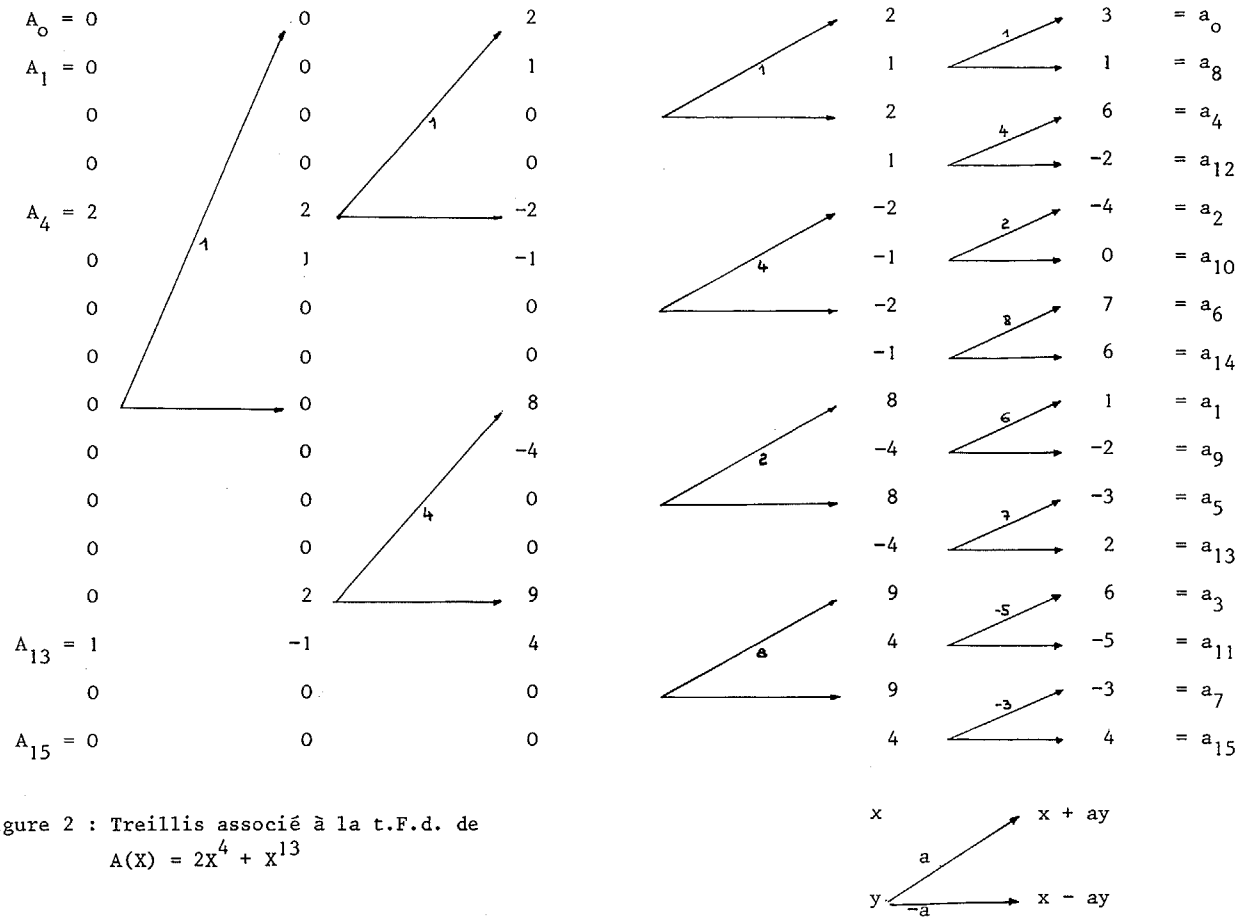


Figure 2 : Treillis associé à la t.F.d. de
$A(X) = 2X^4 + X^{13}$

nombres modulo un premier de Fermat. Certaines nécessitent pour le décodeur une architecture et des circuits particuliers ([6]). Elles n'ont pu être retenues car nous nous sommes restreints à l'utilisation d'un microprocesseur 8 bits de type courant.
La multiplication se fait grâce à une table de "logarithme" :
$a = 6^i \rightarrow i$
Cependant nous avons parfois utilisé la représentation "surabondante" suivante sur 8 e.b. :
$a = \sum_{i=o}^{7} a_i \, 2^i$;
un nombre modulo 17 possède ainsi plusieurs représentations. On vérifie que la multiplication par 2 est réalisée par un décalage circulaire simple. L'addition modulo 17 correspond à l'addition ordinaire si on prend le soin d'ajouter l'éventuel bit de carry au résultat. Cette représentation permet d'éviter les tests par rapport à 17.

7. Résultats

Une première version de l'algorithme a été simulée grâce au programme INTERP associé au MAC 80. Les temps de décodage pour un mot du code (16, 10) corrigeant 3 erreurs sur le corps à 17 éléments sont donnés dans le tableau 1. Rappelons qu'un tel mot peut être représenté par 64 e.b. dont 40 (ou 39) sont des bits d'information, le décodeur corrige au plus 3 paquets d'erreurs de longueur 4.

Nombre d'erreurs	1	2	3
Phase 1 : t.F.d.	2,7	2,7	2,7
Phase 2 : calcul de	5	12	18
Phase 3 : récurrence	8	10	12
+ t.F.d.	2,7	2,7	2,7
Temps total en ms	19	28	35

Tableau 1.

Ces résultats peuvent être extrapolés pour un code (32, 26) corrigeant trois erreurs sur le corps à 257 éléments. Il faut alors utiliser un microprocesseur 16 bits. Le temps de décodage pour un mot (256 e.b.) est dans ce cas inférieur à 50 ms.

Références

[1] J. Justesen, On the complexity of decoding of Reed-Solomon codes.
IEEE Trans. Inform. Theory, Vol IT-22, Mars 1976, pp 237-238.

[2] I.R. Reed, R.A. Scholtz, ... , The fast decoding of Reed-Solomon Codes using Fermat theoretic transform and continued fractions.
IEEE Trans. on Inform. Theory, Vol IT-24, Janv. 1978, pp 100-108.

[3] Y. Sugiyama, M. Kashara, ... , A method for solving key equation for decoding Goppa codes.
Inform. Control 27 (1975) pp 87-99.

[4] W.W. Peterson et E.J. Weldon, Error Correcting codes. MIT Press 1972 ; Cambridge, Mass., p 72.

[5] E.R. Berlekamp, Algebraic Coding Theory. Mc Graw Hill 1968, New York.

[6] J.H. Mac Clellan, Hardware realisation of a Fermat number transform.
IEEE Trans. on Acoustics Speech an Signal Processing, Col ASSP-24, June 1976, pp 216-225.