Un algorithme rapide pour l'acquisition du synchronisme d'une séquence PN dans un système de transmission par étalement de spectre à séquence directe

XiaoYong GUO, Gérard MARAL
Ecole Nationale Supérieure des Télécommunications
(TELECOM Paris), site de Toulouse
Groupe de Recherches et d'Etudes
en Télécommunications et Systèmes Aérospatiaux
BP 4004 - 31028 TOULOUSE Cedex, France

André MARGUINAUD,
ALCATEL ESPACE
11, avenue Dubonnet
92407 Courbevoie Cedex, France

Résumé :

Cet article présente un algorithme permettant de réaliser l'acquisition rapide d'une séquence pseudoaléatoire (PN) dans un système de transmission par étalement de spectre. Cet algorithme est basé sur le calcul rapide de produits de convolution et présente un gain d'un facteur égal à 6 par rapport à un calcul direct pour une longueur de séquence PN égale à 1023.

Introduction :

Les techniques de transmission par étalement de spectre ont un nombre croissant d'applications dans les domaines des télécommunications et de la radionavigation. L'un des problèmes critiques dans l'utilisation de ces techniques est la synchronisation rapide, ou acquisition, de la séquence pseudoaléatoire (PN) de désétalement au niveau du récepteur.

Les récepteurs actuels utilisent des fonctions de traitement analogique, ou bien leurs équivalents numériques. Ils présentent des inconvénients tels que : manque de souplesse, de fiabilité, etc... Le développement rapide des techniques micro-électroniques permet d'envisager la réalisation d'équipements numériques programmables. Il faut toutefois disposer d'algorithmes performants pour l'acquisition de la séquence de désétalement.

Cet article considère le problème de l'acquisition du synchronisme de la séquence pseudoaléatoire dans un récepteur numérique. Une approche séquentielle est utilisée, fondée sur le calcul rapide de corrélations successives.

Le traitement des échantillons se fait par blocs. Les méthodes classiques consistent à effectuer des transformées telle que la transformée de Fourier rapide (FFT). Ces méthodes sont générales, mais pas toujours efficaces en terme de complexité de calcul pour des longueurs de bloc inféricures à environ 1000. En effet, elles font appel à des opérations de multiplication dont la mise en oeuvre sur le silicium (circuits ASIC) est lourde comparée aux cas où les seules opérations à effectuer sont des opérations plus simples, comme l'addition ou la soustraction.

GUO et al. dans [1] ont proposé une nouvelle approche pour le calcul de produits de convolution, qui prend en compte la nature binaire des éléments constitutifs de l'une des séquences en jeu (celle de la séquence locale), et de ce fait ne mettent en oeuvre que des additions et des soustractions. La solution proposée dans [1] se résume comme suit :

- On découpe la séquence des échantillons en segments de faible taille,
- On calcule les produits de convolution entre les segments et la séquence locale,
- On combine les produits de convolution segment.

Cette méthode se montre efficace et permet de réduire le nombre d'opérations (additions ou soustractions) par un facteur de 4 à 7 pour des longueurs de séquence comprises entre 100 et 1000.

Cet article propose une variante à cette approche. Cette variante consiste à *découper non seulement la séquence des échantillons mais aussi la séquence locale binaire*. Le découpage de la séquence locale est nécessaire car le nombre de portes disponibles sur un seul composant (ex. Gate Array) est souvent insuffisant pour mettre en oeuvre la méthode proposée dans [1]. C'est notamment le cas pour des circuits embarqués à bord d'une charge utile de satellite.

Il faut noter que ce double découpage a été initialement proposée par AGARWAL dans le cadre plus général du calcul de convolution entre deux valeurs quelconques (non-binaires) [2].

L'algorithme proposé dans cet article consiste donc à étendre l'algorithme proposé dans [1] en utilisant la technique de double découpage proposée dans [2].

1 - Position du problème de l'acquisition du synchronisme :

On associe à la séquence des paires d'échantillons analytiques reçues $\{x_i , y_i\}$ (i = 0 à ∞) deux séries.

$$(1) \qquad X(t) = \sum_{i=0}^{\infty} x_i t^i \qquad Y(t) = \sum_{i=0}^{\infty} y_i t^i$$

On associe à la séquence locale $\{c_i\}$ (i = 0, N-1) un polynôme :

$$(2) \qquad C(t) = \sum_{i=0}^{N-1} c_{N-1-i} t^i$$

On considère le produit polynomial :

(3) $Z(t) = C(t)[X(t) + jY(t)] = \sum_{i=0}^{\infty} z_i t^i$

avec $z_i = \sum_{n=0}^{N-1} [x_{i+n-N+1} + jy_{i+n-N+1}]\, c_n$, $i = 0$ à ∞.

j est le nombre complexe représenté dans le plan complexe par (0,1).

L'opération d'acquisition du synchronisme consiste à rechercher les exposants i (i = 0, ∞) du polynôme Z(t) pour lesquels les coefficients complexes z_i présentent un module supérieur à un seuil préétabli. Il reste ensuite à rechercher par confirmation la bonne hypothèse, en prolongeant l'intervalle d'observation.

Pour réduire le volume de calculs, il est intéressant de découper la séquence des paires analytiques reçues $\{x_i , y_i\}$ (i = 0 à ∞) en une suite de blocs comportant chacun N paires. A chaque bloc de numéro d'ordre k est associé un polynôme de bloc de degré N - 1, $I_k(t) + jQ_k(t)$, construit à partir des N paires analytiques $\{x_{kN+i}, y_{kN+i}\}$ (i = 0 à N-1) :

(4) $X(t) = \sum_{k=0}^{\infty} I_k(t)\, t^{kN}$ $\quad Y(t) = \sum_{k=0}^{\infty} Q_k(t)\, t^{kN}$

(5) $Z(t) = \sum_{k=0}^{\infty} C(t)[I_k(t) + j\, Q_k(t)]t^{kN}$

On calcule ensuite séparément les 2 produits polynomiaux $C(t)I_k(t)$ et $C(t)Q_k(t)$ pour chaque valeur de k correspondant à l'intervalle temporel considéré. Chaque produit est un polynôme de degré 2(N-1) = 2N - 2.

Le calcul du polynôme Z(t) se ramène donc au calcul d'un ensemble de produits polynomiaux. Chaque produit est de la forme :

(6) $S(t) = A(t)B(t) = \sum_{i=0}^{2N-2} s_i\, t^i$

où A(t) est l'un des polynômes de bloc d'échantillons $I_k(t)$ ou $Q_k(t)$, B(t) est le polynôme associé à la séquence locale :

$A(t) = \sum_{i=0}^{N-1} a_i\, t^i$ $\quad B(t) = \sum_{i=0}^{N-1} b_i\, t^i$

Avec $\{a_i\}$ entiers relatifs, i = 0,..., N-1, $\{b_i\}$ valeurs binaires soit : $b_i = \{1, -1\}$, i = 0,..., N-1, , $\{s_i\}$ valeurs produits (i = 0,..., 2N-2). $\{s_i\}$ représente les 2N - 1 corrélations entre le bloc d'échantillons $\{a_i\}$(i = 0 à N - 1) et la séquence locale $\{b_i\}$(i = 0 à N - 1).

On mesure la complexité du calcul du produit polynomial S(t) = A(t)B(t) par le nombre d'opérations par coefficient du polynôme A(t), Σ_N. L'intégration du produit S(t) dans le produit Z(t) nécessite N additions. Le nombre total d'opérations requises pour le traitement de N échantillons reçus est donc égal à $N + N\Sigma_N$, soit $\Sigma_N + 1$ opérations par échantillon reçu. Dans la suite, on utilisera toujours le nombre d'opérations par coefficient pour mesurer la complexité des méthodes de calcul. Le nombre d'opérations par échantillon reçu s'obtient en y ajoutant une addition.

2 - Algorithme proposé:

On exprime les polynômes A(t) et B(t), associés respectivement à la séquence des échantillons $\{a_i\}$ (i = 0 à N - 1) et la séquence binaire $\{b_i\}$ (i = 0 à N - 1), comme suit :

(7) $A(t) = \sum_{i=0}^{r_1} A_i(t)t^{iv_1}$, $B(t) = \sum_{i=0}^{r_1} B_i(t)t^{iv_1}$

Le produit polynomial S(t) s'exprime comme :

(8) $S(t) = A(t)B(t) = \sum_{i=0}^{r_1} \sum_{j=0}^{r_1} A_i(t)B_j(t)t^{(i+j)v_1}$

L'identité algébrique suivante permet de réduire le nombre de calculs :

(9) $A_i(t)B_j(t)+A_j(t)B_i(t)$
$= [A_i(t)+A_j(t)][B_i(t)+B_j(t)] - A_i(t)B_i(t) - A_j(t)B_j(t)$

Noter que cette identité introduit deux types de produits polynomiaux élémentaires : les précédents, du type $A_i(t)B_i(t)$, et ceux du type $[A_i(t)+A_j(t)][B_i(t)+B_j(t)]$.

Considérons le cas où le nombre de segment est $r_1 = 2$. On a :

(10)
$A(t) = A_0(t) + A_1(t)t^{Vn+1}$, $B(t) = B_0(t) + B_1(t)t^{Vn+1}$
$S(t) = A(t)B(t)$
$= A_0(t)B_0(t) + [A_0(t)B_1(t) + A_1(t)B_0(t)]t^{Vn+1} + A_1(t)B_1(t)t^{2Vn+1}$

(11) $A_0(t)B_1(t) + A_1(t)B_0(t)$
$= [A_0(t)+A_1(t)][B_0(t)+B_1(t)] - A_0(t)B_0(t) - A_1(t)B_1(t)$

En remplaçant (11) dans (10), il vient :

(12) $S(t) = A(t)B(t)$
$= A_0(t)B_0(t) + \{[A_0(t)+A_1(t)][B_0(t)+B_1(t)] - A_0(t)B_0(t) - A_1(t)B_1(t)\}t^{Vn+1} + A_1(t)B_1(t)t^{2Vn+1}$

Ainsi le produit S(t) = A(t)B(t) calculé selon (12) requiert 3 produits polynomiaux élémentaires au lieu des 4 requis selon (10).

D'une manière générale, pour obtenir le produit S(t) = A(t)B(t) de degré $2N - 2 = 2r_1 v_1 - 2$, il faut selon cette méthode : $r_1 (r_1+1)/2$ produits polynomiaux élémentaires entre deux polynômes de degré $v_1 - 1$ et $2(r_1 -1)(v_1 - 1) + [r_1(r_1 -1)/2]v_1 + [(3 r_1 -2)(r_1 -1)/2](2 v_1 -1)$ additions ou soustractions.

Soit $(\Sigma_{v1})_{r1}$ le nombre d'opérations (produit, addition ou soustraction) par coefficient du polynôme $A_i(t)$ pour obtenir un produit polynomial élémentaire entre deux polynômes de degré $v_1 - 1$ pour une décomposition en r_1 polynômes. On note que le nombre d'opérations requises pour calculer les produits polynomiaux élémentaires n'est pas toujours le même, car il dépend du type de produit polynomial élémentaire considéré. Il est plus commode d'introduire comme paramètre le nombre moyen d'opérations requises par coefficient pour calculer un produit élémentaire entre deux polynômes de degré $v_1 - 1$, pour une décomposition en r_1 polynômes, soit $<(\Sigma_{v1})_{r1}>$. Le nombre d'opérations par coefficient Σ_N pour obtenir le produit S(t) = A(t)B(t) de degré 2N - 2 est approximativement égal à :

(13) $\Sigma_{v_1} \approx 7/2\,(r_1 -1) + \frac{r_1 +1}{2} <(\Sigma_{v1})_{r1}>$

3 - Mise en oeuvre de la méthode :

Dans ce paragraphe, on applique la méthode développée ci-dessus pour obtenir le produit polynomial entre le polynôme

de bloc d'échantillons et le polynôme de la séquence locale. On peut l'appliquer directement en une seule étape ou en plusieurs étapes par itération.

Dans le cas où on applique de façon itérative l'algorithme, les facteurs de décompositions sont notés par r_i avec i = 1, 2 etc. En pratique il vaut mieux retenir pour r_i uniquement des valeurs faibles comme 2 ou 3.

Le nombre d'opérations (produit, addition ou soustraction) par coefficient pour obtenir un produit entre deux polynômes de degré N - 1 = r v - 1 ($r = r_1 r_2 r_3 \ldots$) à partir des produits polynomiaux de degré 2v - 2 est égal à :

$$(14) \quad \Sigma_N = \left(\frac{3}{2}\right)^{e_1} 2^{e_2} \left(< (\Sigma_v)_{2^{e_1} 3^{e_2}} > + 7 \right) - 7$$

avec N = rv, $r = 2^{e1} 3^{e2}$.

Il faut exprimer le produit S(t) = A(t)B(t) comme une combinaison de produits polynomiaux élémentaires afin d'identifier ceux qui sont nécessaires à la recombinaison. Cette procédure est particulièrement aisée à mettre en oeuvre si l'on utilise la représentation polonaise (post-fix) des produits polynomiaux. L'un des avantages de cette représentation est qu'elle évite l'emploi de parenthèses, ce qui simplifie l'écriture de la décomposition. Un programme a été ainsi développé pour réaliser la décomposition de façon automatique.

4 - Calcul des produits polynomiaux élémentaires :

Le calcul du produit polynomial S(t) = A(t)B(t) par les méthodes de combinaison précédentes est basé sur le calcul de produits polynomiaux élémentaires notés A?(t)B?(t). Pour le calcul, on tient compte de la particularité des coefficients des polynômes élémentaires B?(t). Ils appartiennent :

- soit à l'ensemble {-1, +1} : on les appelle polynomes de niveau 0.
- soit à l'ensemble {-2, 0, 2} : on les appelle polynomes de niveau 1
- soit à l'ensemble {-4, -2, 0, 2, 4} : on les appelle polynomies de niveau 2.
- etc.

On appelle dorénavant "polynômes de niveau i" les polynômes dont les coefficients appartiennent à l'ensemble {entiers pairs compris entre -2^i et 2^i}, et "produits polynomiaux de niveau i" les produits entre polynômes de niveau i et polynômes quelconques.

4.1- Nombre de produits polynomiaux de niveau i

Le nombre de produits polynomiaux de niveau i est égal à :

- $C_{e1}^{i} \, 2^{e1-i}$ pour le facteur de décomposition $r = 2^{e1}$.

- $3^{e2} \, C_{e2}^{i}$ pour le facteur de décomposition $r = 3^{e2}$.

4.2 - Répartition des valeurs pour un niveau i donné :

Un produit polynomial du niveau i implique un produit polynomial entre un polynôme dont les coefficients sont quelconques et un polynôme dont chacun des coefficients h_j (i = 0 à v-1) est une somme de 2^i termes égaux à +1 ou -1. D'après [3], les coefficients h_j peuvent être modélisés comme une variable aléatoire gaussienne dont la moyenne et la variance sont respectivement égales à 0 et 2^i. L'écart type est $2^{(i/2)}$. Les valeurs que peuvent prendre en pratique les coefficients sont concentrées autour de 0 et elles sont faibles devant 2^i. Pour i = 4, l'écart type est 4, et les valeurs absolues des coefficients sont pratiquement limitées aux valeurs 0, 2, 4, 6, 8, 10,12.

Plusieurs méthodes sont possibles pour calculer les produits polynomiaux élémentaires de niveau i.[3] Elles sont rappelées ci-dessous

4.3 - Méthode exhaustive :

Cette méthode permet de calculer les produits polynomiaux de niveau 0. Elle consiste à subdiviser les polynomes de niveau 0 et de degré v-1 en une somme de polynomes de niveau 0 et de degré v'-1 avec v'<v. La complexité du calcul d'un produit polynomial de niveau 0 et de degré 2(v-1) est :

$\Sigma_v = \Sigma_{v'} + (1 - 1/v)(v/v' - 1)$ opérations additions ou soustractions par coefficient.

avec $\Sigma_{v'} = (2^{v'-1} + (v'-1)^2/2 + v'-1)/v'$.

4.4 - Méthode binaire :

Cette méthode permet de calculer les produits polynomiaux de niveau i > 0. Elle consiste à subdiviser les polynomes de niveau i > 0 et de degré v-1 en une somme de polynomes de niveau 0 et de même degré. On calcule ensuite les produits entre le polynôme échantillon et ces polynômes de niveau 0 par la méthode du paragraphe 4.3. Puis on combine ces pour former les produits polynomiaux de niveau i.

La complexité du calcul d'un produit polynomial de niveau 1 et de degré 2(v-1) est :

$\Sigma_v = \Sigma_{v'} + 2v/v'$ opérations élémentaires par coefficient.
avec $\Sigma_{v'} = (2^{v'-1} + (v'-1)^2/2 + v'-1)/v'$.

4.5 - Calcul direct :

Le calcul direct du produit polynomial consiste à calculer terme à terme le produit A?(t)B?(t) suivant la définition du produit polynomial. Etant donné que les coefficients du polynôme de la séquence locale de niveau i sont connus à l'avance et appartiennent à un ensemble fini d'entiers pairs, on réalise les multiplications par des décalages, des additions ou des soustractions.

Pour multiplier une valeur par 6, on additionne la valeur décalée de 2 positions à gauche (multiplication par 4) et la valeur décalée d'une position à gauche (multiplication par 2). Si le décalage est réalisé par des registres à décalage, il suffit

dans ce cas de 2 additions. Pour un niveau i donné, la probabilité pour chaque valeur d'apparaître et le nombre d'opérations nécessaires sont différents. Il est donc utile de connaître la moyenne et la variance du nombre d'additions ou de soustractions pour une multiplication terme à terme pour les niveaux 1 à 7. Celles-ci sont données dans le tableau 1.

niveau	moyenne	variance
1	0,50	0,25
2	0,63	0,23
3	0,79	0,29
4	0,96	0,35
5	1,13	0,39
6	1,29	0,42
7	1,46	0,45

Tableau 1 - Moyenne et variance du nombre d'additions ou de soustractions pour une multiplication terme à terme pour les niveaux 1 à 7.

Ce tableau permet de determiner le nombre d'additions ou de soustractions requises pour calculer un produit polynomial du niveau i (i = 1 à 7). Dans la suite, on utilise la moyenne pour chiffrer la complexité du calcul. Par exemple, pour calculer un produit polynomial de niveau 4 entre deux polynômes de degré v - 1, on considère qu'il faut $0{,}96v^2$ additions ou soustractions dans le cas où le décalage est réalisé par des registres à décalage, soit 0,96v additions ou soustractions par coefficient du polynôme A?(t).

5 - Application numérique:

Le complexité d'une méthode est comparée au calcul direct du produit polynomial S(t) = A(t)B(t) qui nécessite N - 1 opérations élémentaires par coefficient du polynôme du bloc d'échantillons A(t).

Le calcul du produit polynomial se fait en deux étapes :
- On découpe le bloc de N échantillons en segment de v échantillons et la séquence locale de longueur N en segments de longueur v. On calcule les produits polynomiaux élémentaires de degré 2(v - 1) avec les méthodes développées dans le paragraphe 4.
- on combine ensuite les produits polynomiaux de degré 2(v - 1) pour obtenir le produit polynomial de degré 2(N - 1) avec la méthode du paragraphe 3.

Supposons que N = r v, avec r entier.

Pour calculer les produits polynomiaux élémentaires de niveau 0, on utilise la méthode exhaustive du paragraphe 4.3.

Pour calculer les produits polynomiaux élémentaires de niveau 1, on utilise la méthode binaire du paragraphe 4.4.

Pour calculer les produits polynomiaux élémentaires de niveau 2, 3 et 4, on utilise la méthode du paragraphe 4.5.

On combine enfin les produits polynomiaux pour former le produit polynomial par la méthode de combinaison du paragraphe 3.

Le tableau ci-dessous donne le nombre d'opérations par coefficient du polynôme de bloc pour différentes valeurs de N. La taille v du bloc élémentaire est prise égale à 256.

N	v	ΣN
512	256	96
1024	256	169
2048	256	312

6 - Conclusion :

Cet article a présenté une méthode de calcul de produits de convolution en vue de l'acquisition rapide par un récepteur numérique d'une séquence pseudoaléatoire dans un système de transmissions par étalement de spectre. Il s'agit d'une extension de l'algorithme proposé dans [1] fondée sur une technique de double découpage. Le principe de la méthode est essentiellement combinatoire et utilise des identités algébriques, visant à réduire le nombre d'opérations (additions, soustractions ou décalages) nécessaires à leur mise en oeuvre.

La méthode s'avère particulièrement efficace. Par exemple, pour N = 1024, il suffit de 169 additions ou soustractions par élément. La méthode utilisant la transformée de Fourier rapide (FFT), avec une longueur de transformée égale à 2048, nécessite approximativement 19 multiplications réelles et 59 additions par élément [4]. Supposons que la mise en oeuvre par des réseaux de portes logiques d'une multiplication soit B fois plus complexe qu'une addition, la complexité équivalente de la solution FFT est égale à 19B+59. La solution proposée se montre donc avantageuse lorsque B est supérieur à 6. Non démontré dans cet article est le fait que l'avantage de cet algorithme croît lorsque le signal reçu est affecté par un effet Doppler important [3].

Références :

[1] X.Y. GUO, G. MARAL, A. MARGUINAUD et R. SAUVAGNAC, "Méthode de Calcul rapide de convolution entre deux séquences dont l'une est binaire", Annales des Télécommunications, Tome 46, N° 3-4, pp. 181-190, 1991.
[2]R. C. AGARWAL, C. BARRUS, Fast One-Dimensional Digital Convolution by Multidimensional Techniques, IEEE Trans.on Acoustics Speech, and Signal Processing, Vol. ASSP-22, pp. 1-10, Feb. 1974.
[3] X.Y. GUO, Méthodes numériques pour l'acquisition du synchronisme d'une séquence pseudoaléatoire dans un système de télécommunications par étalement de spectre à séquence direct, mémoire de thèse, Ecole Nationale Supérieure de l'Aéronautique et de l'Espace, Toulouse France, 1990.
[4] P. DUHAMEL, M. VETTERLI, Improved Fourier and Hartly Transform Algorithms : Application to Cyclic Convolution of Real Data, IEEE Trans.on Acoustics Speech, and Signal Processing, Vol. ASSP-35, No 6, June 1987.
[5] M. BELLANGER, Traitement Numérique du Signal, CNET-ENST, Masson, Paris, 1987.