Master CSI 2 — 2010-2011

Cryptanalyse — MHT912

Responsable : G. Castagnos

Devoir surveillé — 2 novembre 2010

Exercice 1. Dans tout l'exercice on note $s = (s_n)_{n\geqslant 0}$ une suite binaire et $S(X)$ sa série génératrice définie par $S(X) = \sum_{n\geqslant 0} s_n X^n$.

(a) Soit $f(X) \in \mathbb{F}_2[X]$ un polynôme de degré L avec $f(X) = 1 + c_1 X + c_2 X^2 + \cdots + c_L X_L$. Rappeler sans démonstration la formule reliant $S(X)$ et $f(X)$ pour que s soit produite par un LFSR de polynôme de rétroaction $f(X)$.

La suite s est produite par un LFSR de polynôme de rétroaction $f(X)$ si et seulement si on a le développement en série formelle $s(X) = g(X)/f(X)$, où $g(X)$ est un polynôme de $\mathbb{F}_2[X]$ tel que $\deg(g) < \deg(f)$. En outre, le polynôme $g(X)$ est entièrement déterminé par l'état initial du registre :

$$g(X) = \sum_{i=0}^{L-1} X^i \sum_{j=0}^{i} c_j s_{i-j}.$$

(b) On suppose maintenant que s est une suite binaire périodique quelconque de période T. Montrer que $X^T S(X) = S(X) + \sum_{i=0}^{T-1} s_i X^i$. En déduire le polynôme de rétroaction d'un LFSR permettant d'engendrer S ainsi qu'une méthode pour déterminer le polynôme de rétroaction minimal d'une suite binaire périodique.

On calcule $X^T S(X) = \sum_{i\geqslant 0} s_i X^{i+T} = \sum_{i\geqslant 0} s_{i+T} X^{i+T}$ car s est périodique de période T. Par changement de variable, on obtient que $X^T S(X) = \sum_{i\geqslant T} s_i X^i = S(X) + \sum_{i=0}^{T-1} s_i X^i$. On en déduit que $(X^T + 1)S(X) = \sum_{i=0}^{T-1} s_i X^i$. Donc que $S(X) = g(X)/f(X)$ avec $f(X) = 1 + X^T$ et $g(X) = \sum_{i=0}^{T-1} s_i X^i$. On peut donc engendrer s avec un LFSR de polynôme de rétroaction $1 + X^T$. Pour trouver le polynôme de rétroaction minimal, on écrit $S(X) = g_0(X)/f_0(X)$ avec $\gcd(f_0, g_0) = 1$, en divisant $f(X)$ et $g(X)$ par leur pgcd.

(c) On suppose que s est une *m-sequence* de complexité linéaire L. Comparer l'efficacité de la méthode de la question précédente avec la méthode vue en cours pour trouver le polynôme de rétroaction minimal de s (on rappelle que le calcul du pgcd de deux polynômes de $\mathbb{F}_2$ de degrés inférieurs à e peut être effectué en $\mathcal{O}(e \log^2 e \log\log e)$ opérations dans $\mathbb{F}_2$).

L'algorithme de Berlekamp Massey permet de retrouver le polynôme minimal de s avec $2L$ bits en L^2 opérations. Ici la période est 2^L-1 et on a besoin de calculer le pgcd entre le polynôme $f(X)$ et $g(X)$ de degrés respectifs 2^L-1 et 2^L-2. On obtient donc une complexité exponentielle en L, nécessitant de plus 2^L bits de s, extrêmement moins efficace que la méthode de Berlekamp Massey.

(d) **Application avec Magma.** Soit une suite binaire périodique de période 7, répétant le motif 1000101. Calculer le polynôme de rétroaction minimal avec la méthode de la question (b). Retrouver ce résultat avec une méthode plus efficace. Indiquer et justifier les commandes Magma utilisées.

```
s :=[GF(2) ! 1,0,0,0,1,0,1] ;
PR<X> :=PolynomialRing(GF(2)) ;
g :=Pr !s[1..7] ; Attention au décalage d'indices!
f :=X^7+1 ;
f/Gcd(g,f) ;
```

On trouve le polynôme $X^4+X^3+X^2+1$. On obtient donc une complexité linéaire de 4, il faut donc au moins $2\times 4=8$ bits en entrée pour que l'algorithme de Berlekamp Massey trouve le polynôme de rétroaction (d'autre part, par la question (b) on sait que la complexité linéaire est plus petite que la période).

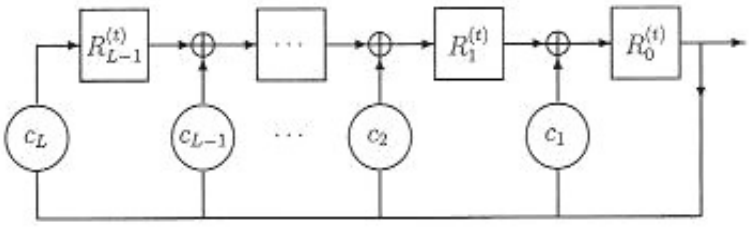
```
t := s cat s ;
BerlekampMassey(t) ;
```

On retrouve bien le polynôme $X^4+X^3+X^2+1$.

Exercice 2. Soit $f(X)\in\mathbb{F}_2[X]$ un polynôme de degré L avec $f(X)=1+c_1X+\cdots+c_LX_L$. On considère un automate constitué d'un registre à décalage de L bits. On note $R^{(t)}=(R_0^{(t)},R_1^{(t)},\ldots,R_{L-1}^{(t)})$ l'état du registre à l'instant $t\geqslant 0$. À l'instant t, on sort le bit de poids faible du registre, $R_0^{(t)}$, et on met à jour l'état du registre de la façon suivante (calculs dans $\mathbb{F}_2$) :

$$R_i^{(t+1)}=R_{i+1}^{(t)}+c_{i+1}R_0^{(t)},\ \text{pour } 0\leqslant i\leqslant L-2 \text{ et } R_{L-1}^{(t+1)}=c_LR_0^{(t)}$$

Un tel automate est appelé **LFSR en représentation Galois**. Le polynôme $f(X)$ est son polynôme de rétroaction. On le représente par le schéma suivant :



(a) **Avec Magma.** Donner le code d'une fonction prenant en entrée $f(X)$ de degré L, R de L bits et N et retournant les N premiers bits générés par un LFSR en représentation Galois de polynôme de rétroaction $f(X)$, dont le registre est initialisé par R (c'est à dire tel que $R^{(0)}=R$). Donner les 5 premiers bits produits par le LFSR Galois de polynôme de rétroaction $1+X+X^3$ et initialisé par $[1,1,0]$.

```
LFSRGalois := function(f,R,N);
    s := [];
    L := Degree(f);
    for i in [1..N] do
      Append(~s,R[1]);
      R := [Eltseq(f)[i+1]*R[1]+R[i+1] : i in [1..L-1]] cat
[R[1]*Eltseq(f)[L+1]];
    end for;
    return s;
end function;
LFSRGalois(1+X+X^3,[GF(2)! 1,1,0],5);
```

On obtient $[1, 0, 0, 1, 1]$.

(b) Pour tout entier t, on désigne par $R^{(t)}(X)$ le polynôme de $\mathbb{F}_2[X]$ de degré $L-1$ correspondant au registre au temps t : c'est à dire $R^{(t)}(X) = R_0^{(t)} + R_1^{(t)}X + \cdots + R_{L-1}^{(t)}X^{L-1}$. On note s_t le bit sorti au temps t (c'est à dire $R_0^{(t)}$).
Montrer que pour tout entier $t \geqslant 0$, $X \times R^{(t+1)}(X) = R^{(t)}(X) + s_t \times f(X)$.

C'est la traduction des formules de mise à jour du registre. Au temps t le contenu du registre est $R^{(t)}(X)$, et $s_t = R_0^{(t)}$. On a donc $R^{(t)}(X) + s_t f(X) = s_t + (s_t \times 1) + (R_1^{(t)} + s_t c_1)X + \cdots + (R_{L-1}^{(t)} + s_t c_{L-1})X^{L-1} + s_t c_L X^L$, ce qui est bien égal à $XR^{(t+1)}(X)$ d'après les formules de rétroaction.

(c) On note $S^{(0)}(X) = 0$ et pour tout $t \geqslant 1$, $S^{(t)}(X) := s_0 + s_1 X + \cdots + s_{t-1}X^{t-1}$. Montrer par récurrence sur t que pour tout $t \geqslant 0$, $R^{(0)}(X) = f(X) \times S^{(t)}(X) + X^t \times R^{(t)}(X)$.

Pour $t = 0$, on a $R^{(0)}(X) = 0 + X^0 \times R^{(0)}(X)$. On suppose la propriété vraie au rang t. On a alors $R^{(0)}(X) = f(X) \times S^{(t)}(X) + X^t \times R^{(t)}(X) = f(X) \times S^{(t)}(X) + X^t \times (X \times R^{(t+1)}(X) + s_t \times f(X)) = f(X) \times (S^{(t)}(X) + s_t X^t) + X^{t+1} \times R^{(t+1)}(X) = f(X) \times S^{(t+1)}(X) + X^{t+1} \times R^{(t+1)}(X)$.

(d) On note $S(X)$ la série génératrice de la suite produite par le LFSR en représentation Galois, c'est à dire que $S(X) = \sum_{t \geqslant 0} s_t X^t$. Déduire de la question précédente que $S(X) = R^{(0)}(X)/f(X)$. Montrer que toute suite récurrente linéaire produite par un LFSR classique peut l'être par un LFSR en représentation Galois et réciproquement.

D'après la question précédente, pour tout entier k, on a $R^{(0)}(X) + f(X)S(X) = f(X) \times \sum_{k \geqslant t} s_k X^k + X^t \times R^{(t)}(X)$. Donc $R^{(0)}(X) + f(X)S(X)$ est divisible par X^t pour tout entier t. On en déduit que $R^{(0)}(X) = f(X)S(X)$. Ainsi si s est produite par un LFSR classique, on a $S(X) = g(X)/f(X)$, où g est donné par l'état initial du registre comme vu en 1.(a). On peut donc produire s par un LFSR en représentation Galois de polynôme de rétroaction f et initialisé par les coefficients de g. Réciproquement, si s est produite par un LFSR en représentation Galois on a $S(X) = g(X)/f(X)$, où g est l'initialisation du registre. On peut produire s par un LFSR classique de polynôme de rétroaction f et initialisé par les premiers bits de s.

(e) **Application avec Magma.** Donner les commandes pour construire les 100 premiers bits de la suite produite par le LFSR en mode Galois de polynôme de rétroaction $1 + X^2 + X^4$ et initialisé par $[0, 1, 0, 1]$. Donner les commandes pour construire ces 100 bits par un LFSR en mode classique en précisant le polynôme de rétroaction et l'initialisation utilisés.

```
s1 :=LFSRGalois(1+X^2+X^4,[GF(2) | 0,1,0,1],100);
```
On utilise le même polynôme de rétroaction pour le LFSR classique et les 4 premiers bits de la suite s_1 pour l'initialisation.
```
s2 :=LFSRSequence(1+X^2+X^4,s1[1..4],100);
```
On a bien $s1 = s2$.

(f) **Autre application avec Magma.** Réciproquement, donner les commandes pour construire les 100 premiers bits de la suite produite par le LFSR classique de polynôme de rétroaction $1 + X^3 + X^5$ et initialisé par $[1, 0, 1, 0, 0]$. Donner les commandes pour construire ces 100 bits par un LFSR en mode Galois en précisant le polynôme de rétroaction et l'initialisation.

On utilise la fonction vue en TP, donnant le polynôme g correspondant à l'initialisation, state, d'un LFSR classique de polynôme de rétroaction f.
```
g :=function(f,state);
    L :=#state;
    return &+[ (&+[state[i-j+1]*Eltseq(f)[j+1] : j in [0..i]]) *
X^i : i in [0..L-1]];
end function;
s3 :=LFSRSequence(1+X^3+X^5, [GF(2) | 1,0,1,0,0], 100);
gg :=g(1+X^3+X^5, [GF(2) | 1,0,1,0,0]);
```
On trouve $0 \times X^4 + X^3 + X^2 + 1$. On génère le LFSR en représentation Galois initialisé par ce polynôme (on rajoute un 0 pour faire un état de 5 bits), avec le même polynôme de rétroaction.
```
s4 :=LFSRGalois(1+X^3+X^5,[GF(2)| 1, 0, 1, 1 ,0],100);
```
On a bien $s_3 = s_4$.

(g) Pour des implantations matérielles on préfère parfois représenter les LFSR en mode Galois plutôt qu'en mode classique (on dit Fibonacci). Pourquoi ?

Dans les LFSR en représentation Galois, les calculs de mises à jour de l'état peuvent être fait de manière simultanée (« en parallèle ») par des circuits d'additions différents. Dans les LFSR en représentation Fibonacci, le calcul de la rétroaction fait intervenir tout l'état. Il faut donc « attendre » qu'elle ait été calculée avec de décaler le registre. Ainsi la représentation Galois permet d'utiliser des cycles d'horloges plus rapides, et offre un meilleur débit.