

# Codage de l'information

## M1 Informatique 2015/16

### TP Générateur pseudo-aléatoire

Letay Benoit

15 décembre 2015

## 1 Introduction

L'objectif de ce TP est d'écrire l'ensemble des fonctions composant la mise en oeuvre du codage de longueur maximale, du code de Gold et du code JPL.

## 2 Fonctionnalités

Notre programme génère des codes de longueur maximale, des codes Gold et des codes JPL en fonction des différents paramètres entrés par l'utilisateur.

Exemple d'utilisation : figure 1 page 3

## 3 Fonctionnement

La fonction `next_code_longueur_maximale` renvoie la prochaine séquence d'un code à longueur maximale à partir d'un vecteur d'initialisation (ou de la dernière séquence), la longueur maximale voulue et le polynôme tous entrés en paramètre. A chaque itération nous calculons le résultat d'un XOR entre les différents bits indiqués par le polynôme, puis on insère ce résultat au début de la séquence tout en décalant le reste des bits.

```
1
1 def next_code_longueur_maximale(init,n,polynome):
2     sequence = copy.copy(init)
3     somme = 0
4     for y in range(len(polynome)):
5         somme += sequence[polynome[y]-1]
6     somme = somme % 2
7     for y in reversed(range(1,n)):
8         sequence[y] = sequence[y-1]
9     sequence[0] = somme
10    return sequence
```

La fonction next code gold renvoie la prochaine séquence d'un code Gold à partir de deux séquences de code à longueur maximale. Le code Gold est extrêmement simple et consiste uniquement en un XOR entre les deux séquences.

```
1
1 def next_code_gold(m1,m2):
2     r = []
3     for x in range(0,len(m1)):
4         r.append(int((m1[x] + m2[x])%2))
5     return r
```

La fonction code JPL renvoi la prochaine séquence d'un code JPL à partir de m séquences de code à longueur maximale. Encore une fois c'est un XOR que nous effectuons ici sous la forme d'un modulo 2.

```
1
1 def code_JPL(tab_m):
2     r = copy.copy(tab_m[0])
3     for x in range(1,len(tab_m)): #Pour chaque code a longueur maximal
4         for i in range(0,len(tab_m)):
5             r[x] += tab_m[x][i%len(tab_m[x])]
6     for x in range(0,len(r)):
7         r[x] %= 2
8     return r
```

FIGURE 1 – affichage lors de l'exécution du programme

```
[paan] C:\Dossier\Programmation\M1 ISI\codage>python tp2.py
Entrez le type de code à generer:
  1.Code a longueur maximale
  2.Code Gold
  3.Code JPL
  4.Quitter
1
Entrez le nombre de bits :36
Entrez la longueur de la sequence maximale de la generer :5
Entrez le polynome de generation (forme : 5 2 pour un polynome[5,2]) :5 2
Entrez le vecteur d'initialisation (forme 1 1 1 1 pour [1,1,1,1] :1 1 1 1 1
011110011110011110010110010110010110
Entrez le type de code à generer:
  1.Code a longueur maximale
  2.Code Gold
  3.Code JPL
  4.Quitter
2
Entrez le nombre de bits :36
Entrez la longueur de la sequence a generer :5
Entrez le premier polynome de generation (forme : 5 2 pour un polynome[5,2]) :5
2
Entrez le second polynome de generation (forme : 5 2 pour un polynome[5,2]) :5 4
2 1
Entrez le vecteur d'initialisation (forme 1 1 1 1 pour [1,1,1,1] :1 1 1 1 1
100000100010100010100010110010110011
Entrez le type de code à generer:
  1.Code a longueur maximale
  2.Code Gold
  3.Code JPL
  4.Quitter
```