TP TICOD N°03 Le Codage de Hamming

HALLA Senia - IGE43, Groupe 2

Le 11 Janvier 2022

1 Génération d'un code

1.1 L'expression théorique de la matrice G:

La matrice génératrice G d'un code linéaire est la matrice de l'application linéaire d'encodage.

Sous forme réduite elle est égale à :

$$G = [I_k, P]$$

Ou I est la matrice est une matrice d'identité de taille $(k \times k)$, et P de dimension $(k \times m)$ nous permets de calculer les bits de parité

1.2 Exécution du code et extraction des Matrice :

1.2.1 Les matrice H, G, G^T

Voir les Figure 1 - 2 - 3

```
>> size(G)
ans =
    4    7
>> G
G =
    1    1    0    1    0    0    0
    0    1    1    0    1    0    0
    1    1    0    1    0    0    1
    1    1    1    0    0    1    0
    1    1    0    0    0    1
    1    0    1    0    0    0    1
    1    0    1    0    0    0    0
    1    1    0    0    0    0    1
}
```

Figure 1: La Matrice de parité : G

Figure 2: La Matrice Génératrice : H

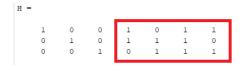
```
>> size(G')
ans =
     7
            4
>> G'
ans =
     1
            0
                   1
                          1
     1
            1
                   1
                          0
     0
            1
                   1
                          1
            0
     1
     0
            1
                   0
                          0
                          0
                   1
            0
                   0
                          1
     0
```

Figure 3: La Transposée de $\mathcal{G}:\mathcal{G}^T$

1.2.2 La valeur du vecteur C:

 $[0\ 1\ 1\ 0\ 1\ 0\ 0]$

1.2.3 Les bits de parité P



Les m (4) premiers bits, du C, représente le message et les k (3) derniers bits, représente la redondance



1.2.4 Comparaison entre C1 et C2:

 ${
m C1}$ et ${
m C0}$ sont identiques



1.2.5 Comparaison entre M et R0:

 ${\bf M}$ et R0 sont identiques

```
>> M'
ans =

0 1 1 1
>> R0'
ans =

0 1 1 1
```

2 Syndrome et correction

2.1 La valeur de $Z \rightarrow [101]$

2.2 Simulation de l'occurrence d'une erreur

2.2.1 L'opération réalisé à la ligne 20

$$C1(i) = mod(C1(i) + 1, 2);$$

On introduit des erreurs dans le code C1 : Les 0 vont être remplacés pas des 1 et vice-versa.

2.2.2 Le vecteur Z

En Multipliant Le message transmis (Vecteur C1) et la matrice de parité H en obtient le vecteur Z est le syndrome, nous indiquant l'erreur additive

2.2.3 Le matrice \mathbf{Z}_n

```
>> Zn
Zn =

1  0  0  1  0  1  1
0  1  0  1  1  1
0  0  0  1  0  1  1
```

Cette matrice rassemble les vecteur de syndrome pour détection d'erreur de chaque ligne

2.2.4 Le matrice H et \mathbf{Z}_n

Les deux matrice sont identique

Chaque colonne hi représente la position d'erreur (Conversion en base de 10)

2.3 Exemple d'application $C = [1 \ 1 \ 1 \ 1 \ 0 \ 1 \ 1]^t$

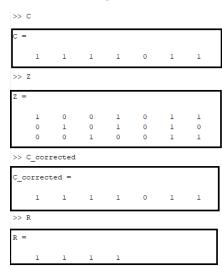
2.3.1 La valeur enregistré par la variable Test

Elle indique le nombre d'erreur dans le message codé.

2.3.2 Compléter le code

```
\begin{array}{ll} for & i = 1 : length\left(C\right) \\ & test = symerr\left(Z, \ H(1 : k \,, i \,)\right); \\ & i f\left(test = \! = \! 0\right) \\ & i \\ & C\_corrected\left(i\right) = mod(C1(i)*H(1 : k \,, i) \,+\, 1 \,, 2) \\ & end \\ end \end{array}
```

2.3.3 Affichage des valeur C, Z, C-corrected, et R



On a pu, à la réception, décoder le message reçu sans erreur et récupérer les bits [1111] contenant l'information, en enlevant la redondance introduite par le codage de Hamming.

Et ça grâce au syndrome qui nous a donné la position de l'erreur dans le mot reçu entaché d'une erreur simple.