



Réseaux Locaux

Codes correcteurs d'erreurs

2022/2023

Rapport de projet

Hugo Himber 22000333

Vincent Cardile 21301144

Dans ce rapport, nous aborderons en premier lieu les réponses aux questions posées dans le sujet, car les questions du sujet couvrent déjà l'explication de l'implémentation des éléments complexes. Nous n'aurons pas de section dédiée aux choix d'implémentations.

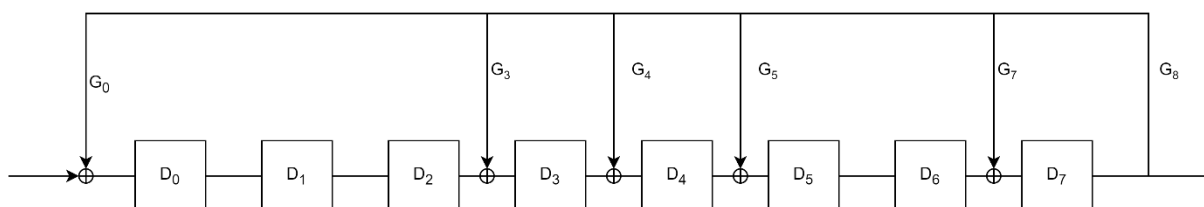
Question 1 : Quel est le rendement du code décrit par la matrice G ? Justifiez.

Question 4.2 : Expliquez votre algorithme.

Question 4.3 : Sachant la distance, déduisez combien d'erreur(s) ce code peut détecter/corriger. Est-il pertinent d'avoir un code avec une distance paire ?

Étant donné que la distance de Hamming est de 4, on peut détecter jusqu'à 3 erreurs avec ce code. Cependant, on ne peut corriger qu'une seule erreur car pour 2 erreurs, on ne sait pas (sans probabilités) quel était le mot d'origine. Ce qui montre qu'une distance de Hamming paire n'est pas très pertinente.

Question 5 : Dessinez le registre à décalage implémentant le code généré par P(X). Expliquez votre construction.



Les registres D_0 à D_7 représentent les registres à vérifier et les liens alimentant le circuit représente le masque du polynôme qui y est appliqué, ici de manière gros-boutiste. Si l'on considère la présence des liens comme un 1 et leurs absences un 0 nous obtenons (en petit-boutiste) 110111001. En remplaçant les 1 par des X aux degrés équivalents de leurs puissances de deux, on obtient $x^7 + x^8 + x^5 + x^4 + x^3 + 1$, ce qui est notre polynôme.

Question 9 : Au vu de vos observations, peut-on (a priori) faire du décodage par syndrome avec un code polynomial ? Justifiez votre réponse.

Oui, on peut faire un décodage par syndrome avec un code polynomial, car le passage du mot dans *mod_poly* renvoie le syndrome et donc la position de l'erreur. Cela fonctionne car le premier passage dans *mod_poly* encode le mot et le deuxième envoie le syndrome. (Voir jeux de tests du programme)

Question 10 : Expliquez votre implémentation. (correct)

Pour la fonction *correct*, tout d'abord j'utilise *mod_poly* pour obtenir le syndrome. Ensuite, je compare le syndrome avec les syndromes possibles via une boucle for parcourant tous les syndromes possibles dans l'ordre pour déterminer la position de l'erreur, et enfin je corrige grâce à la position déterminée. Pour la fonction *correct_matrix*, j'utilise la multiplication de matrices pour déterminer le syndrome, puis j'utilise la même technique que précédemment pour déterminer la position de l'erreur.

Il aurait aussi été possible de déterminer la position de l'erreur en établissant une liste de syndromes établie depuis les colonnes de H, mais je trouvais cette méthode moins élégante. J'ai choisi de prendre la méthode par polynôme car elle est plus simple à implémenter et engendre moins de calculs.

Question 11.1 : Y-a-t-il une correspondance unique entre les syndromes et les vecteurs d'erreurs de 2 bits ? Pourquoi ?

Non, car avec une distance de Hamming de 4 (déterminée dans la question 4) on peut détecter 3 erreurs mais on peut seulement corriger jusqu'à 1 erreur, car la 2ème erreur est ambiguë.

Question 11.2 : Peut-on faire du décodage par syndrome correct dans tous les cas si deux erreurs surviennent sur le medium ? Dans certains cas ?

Non, on ne peut pas faire de décodage par syndrome dans tous les cas, car il faut que la distance de Hamming soit impaire. Oui, il existe peut-être des situations où la deuxième erreur est corrigeable correctement si on a de la « chance ».

Question 11.3 : Le code est-il parfait ?

Oui, le code est parfait car le mieux qu'on puisse corriger efficacement sur une détection de 3 erreurs est 1 erreur.

Question 13 : Expliquez votre implémentation. (correct2errors)

J'utilise la même base que la fonction *correct*, les premières modifications commencent à la partie qui recherche la position de l'erreur. Ici, si on ne trouve pas qu'une seule erreur, on entre dans une boucle qui va chercher à calculer tous les syndromes pouvant être créés à partir de 2 erreurs. Dès lors qu'une correspondance est établie, je tire au sort une des corrections possibles.

Pour cette fonction, j'aurais aussi pu utiliser une liste de syndromes pour localiser l'erreur, car en faisant un ou exclusif entre le syndrome obtenu et l'un des éléments de la liste, on obtient le syndrome de la deuxième erreur si l'élément était une des erreurs.