M1 Informatique

Année 2010-2011

Examen de rattrapage de Protocoles Réseaux

éléments de corrections


corrigé partiel et non officiel,
à lire d'un œil critique


(Exercice 1 à 3 non traités)

Exercice 4 :

Avant tout, comprenons bien l'énoncé :

- "Z / $2Z$" = "les restes entiers des divisons par 2", c'est-à-dire 0 et 1, donc c'est juste une façon un peu sophistiquée et mathématicienne de désigner les bits.
- Les *codes cycliques* font partie de la famille des *codes correcteurs* : l'idée est de transformer un mot binaire en un autre mot binaire plus long avec une méthode précise. Les mots qu'on peut ainsi créer forment un *code*. Avant de transmettre un mot, l'expéditeur le code, et envoie le mot codé. À son arrivée, si le mot a été légèrement modifié (de 1 bit par exemple, cela arrive parfois lors d'une transmission), le destinataire pourra détecter l'incohérence du message et le corriger.
- Il s'agit ici de coder des mots de longueur 4 (d'après la question 1) pour en faire des mots de longueur 7 (d'après le début de l'énoncé, et les questions 4 et 5) à l'aide d'un polynôme de degré 3.

Question 1 :

Méthode

Comment transformer un message (comme 1010) en un mot du code d'après le polynôme générateur ?
Rappelons qu'on peut voir tout mot en binaire comme un polynôme, et vice versa.
Quelques exemples :

- $1 + X + X^2 + X^3$ → 1111
- $1 + X + X^3$ → 1101 (polynôme de l'énoncé)
- $1 + 2X + X^3$ → 1001 (on ne garde que le reste modulo 2 de chaque monôme)
- $1 + X^5 + X^7$ → 10000101

Remarque : Pour construire la matrice génératrice on met les bits de poids fort à droite.

Donc notre polynôme générateur g(X) correspond au mot binaire 1101, gardons cette info de coté.

Pour coder un mot peut passer par une matrice génératrice notée G.
On pourra alors multiplier un mot à coder par la matrice G et obtenir le mot codé.
Comme les mot à codé sont de taille 4, et les mots du code de taille 7, on sait que cette matrice aura 4 lignes et 7 colonnes.

$$\begin{vmatrix} - & - & - & - & - & - & - \\ - & - & - & - & - & - & - \\ - & - & - & - & - & - & - \\ - & - & - & - & - & - & - \end{vmatrix}$$

(*mot m*) (*code du mot m*)

Pour remplir la première ligne de la matrice, on utilise notre polynôme g(X) converti en mot binaire trouvé précédemment, et on complète la ligne avec des 0 :

$$\begin{vmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ - & - & - & - & - & - & - \\ - & - & - & - & - & - & - \\ - & - & - & - & - & - & - \end{vmatrix}$$

Pour remplir les lignes suivantes on recommence en décalant le polynôme à chaque fois :

$$\begin{vmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{vmatrix}$$

À présent qu'on a la matrice génératrice G, il suffit de multiplier 1010 par G pour trouver son code :

				1	1	0	1	0	0	0				
				0	1	1	0	1	0	0				
				0	0	1	1	0	1	0				
				0	0	0	1	1	0	1				
(1	0	1	0)	(1	1	1	0	0	1	0)				

On obtient ainsi le mot codé <u>1110010</u>.

Correction

Le code du mot 1010 est 1110010.

Autre méthode

Il est également possible de trouver le code sans passer par la matrice génératrice.
On converti le mot cherché en polynôme : 1010 → $1 + X^2$
Puis on le multiplie par le polynôme générateur :

$(1 + X^2) * (1 + X + X^3)$ $= 1 * (1 + X + X^3) + X^2 * (1 + X + X^3)$
$= 1 + X + X^3 + X^2 + X^3 + X^5$
$= 1 + X + X^2 + 2X^3 + X^5$

Et enfin on le reconvertit le polynôme obtenu en binaire :
$1 + X + X^2 + 2X^3 + X^5$ → <u>1110010</u>

On trouve bien le même résultat qu'avec la première méthode.

<u>Question 2</u> :

Matrice génératrice déjà calculée, cf question 1.

Correction

1	1	0	1	0	0	0
0	1	1	0	1	0	0
0	0	1	1	0	1	0
0	0	0	1	1	0	1

Question 3 :

La Matrice de contrôle H va servir au destinataire du message pour détecter une éventuelle erreur : il va multiplier le mot du code reçu par H^t (transposée de la matrice de contrôle), et s'il obtient le vecteur nul (un mot constitué uniquement de 0) cela voudra dire que le transfert s'est déroulé sans erreur. S'il trouve un vecteur non nul, cela voudra dire au contraire que le mot a été modifié, et le code d'erreur ainsi trouvé, appelé *syndrome*, lui permettra de corriger l'erreur.

Méthode

Comme la matrice G, la matrice H peut se construire à partir d'un polynôme h(X), et $h(X) = (X^n + 1) / g(X)$, où n est la longueur du code (n=7 ici).

Là, il faut faire un peu de division Euclidienne, je passe sur les calculs.
On obtient normalement $(X^7 + 1) / (X^3 + X + 1) = X^4 - X^2 - X + 1$
et $X^4 - X^2 - X + 1 \rightarrow 10111$ ("+X" équivalent à "−X" en modulo 2)

Remarque : Pour construire la matrice de contrôle on met les bits de poids fort à gauche.

La matrice H a la même longueur que code.
On la construit comme G avec 10111.

1	0	1	1	1	0	0
0	1	0	1	1	1	0
0	0	1	0	1	1	1

Pour écrire ensuite la transposée H^t (pas demandée dans la question 3 mais qui sera utile pour les questions suivantes) on transforme les lignes en colonnes et les colonnes en lignes.
Par exemple, la première colonne de H est 100 (en lisant de haut en bas)
Donc la première ligne de H^t sera 100, et ainsi de suite.

1	0	0
0	1	0
1	0	1
1	1	0
1	1	1
0	1	1
0	0	1

Correction

1	0	1	1	1	0	0
0	1	0	1	1	1	0
0	0	1	0	1	1	1

Question 4 :

Méthode

Une fois le mot codé transmis, le destinataire le multiplie par H^t, et obtient ce qu'on appelle le *syndrome* du mot.
Si celui ci est égale au vecteur nul (000 dans notre cas) alors il n'y a pas d'erreur.
Si au contraire il y a eu une erreur (et une seule) alors le résultat sera différent de 0, et la ligne de H^t à laquelle il correspondra indiquera le n° du bit où se situe l'erreur.

							1	0	0
							0	1	0
							1	0	1
							1	1	0
							1	1	1
							0	1	1
							0	0	1
(1	1	1	0	1	0	0)	(1	0	0)

On trouve 100 comme résultat, donc non, 1110100 ne fait pas partie du code.
Le syndrome 100 correspond à la première ligne de H^t, donc si on suppose que l'expéditeur a envoyé un mot du code (c'est-à-dire un mot que la matrice G a pu lui donner), et qu'une seule erreur s'est produite durant le transfert (car l'erreur reste un phénomène relativement rare), alors l'erreur se situerait sur le premier bit du mot, et le mot expédié ne serait pas 1110100 mais 0110100.

Remarque : On peut en déduire le mot d'origine avant le codage qui serait alors 0100, mais il n'est pas demandé de décoder le mot dans cette exercice, on s'intéresse seulement à la détection d'erreurs.

Correction

En multipliant 1110100 par la transposée de la matrice de contrôle H^t on obtient pas 000 mais 100, donc 1110100 n'appartient pas au code.

Question 5 :

Méthode

Même méthode que pour la question précédente :

							1	0	0
							0	1	0
							1	0	1
							1	1	0
							1	1	1
							0	1	1
							0	0	1
(1	1	0	0	0	1	1)	(1	0	0)

Il se trouve qu'on tombe encore sur 100, donc s'il n'y a qu'une seule erreur le bit erroné est encore le premier, et le mot correct est 0100011.

Autres exemples (pour ne pas faire toujours le même)

1100000 est-il un mot du code ? Si non où est l'erreur ?
Mêmes questions pour 1100101.

Réponses :

							1	0	0
							0	1	0
							1	0	1
							1	1	0
							1	1	1
							0	1	1
							0	0	1
(1	1	0	0	0	0	0)	(1	1	0)

1100000 n'est pas un mot du code, le syndrome trouvé correspond à la 4ème ligne de la matrice, l'erreur est donc sur le 4ème bit, le mot d'origine était 1101000.

							1	0	0
							0	1	0
							1	0	1
							1	1	0
							1	1	1
							0	1	1
							0	0	1
(1	1	0	0	1	0	1)	(0	0	0)

Le syndrome est 000, donc 1100101 appartient bien au code, pas d'erreur détectée.

Correction

1100011 n'est pas un mot du code, son syndrome n'est pas 000 mais 100.
S'il n'y a eu qu'une erreur alors elle s'est produite sur le premier bit, car 100 correspond à la première ligne de la matrice H^t, 0100011 était donc le mot d'origine .

Question 6 :

(...)

Correction

Oui, le code peut toujours corriger une erreur unique, car la distance minimale du code est de 3.

Question 7 :

(...)

Correction

Non, le code ne peut pas corriger deux erreurs, car la distance minimale du code est de 3 (il faudrait qu'elle soit au moins de 5 pour pouvoir corriger 2 erreurs).

Autre méthode

On peut aussi exhiber un exemple pour justifier que le code ne permette pas de corriger 2 erreurs.

Par exemple, si le destinataire reçois le mot 1000001, il saura qu'il y a au moins une erreur, mais il sera incapable de distinguer dans lequel des cas suivants il se trouve :

- Le mot d'origine était 1010001 (code pour 1101) et il y a eu 1 seule erreur : sur le 3ème bit.
- Le mot d'origine était 0000000 (code pour 0000) et il y a eu 2 erreurs : sur le 1er et le 7ème bit.
- Le mot d'origine était 1001011 (code pour 1111) et il y a eu 2 erreurs : sur le 4ème et le 6ème bit.

Remarque : Avec 2 erreurs sur un mots, ce code ne peut pas les corriger, mais il est encore capable de détecter la présence d'erreurs. Mais à partir de 3 erreurs, le code ne peut plus toujours les détecter. Par exemple 0000000, qui est dans le code, peut être un mot arrivé sans erreurs, mais peut aussi être l'un des mots du code suivants avec 3 erreurs : 1101000, 0110100, 0011010, 0001101.

(Exercice 5 non traité pour l'instant)