Cours 6. Codes correcteurs

A. Désilles

10 mai 2010

Résumé

Résumé

Résumé

Notion de code de canal

Définition

Soit un canal $X, Y, P(Y|X)$. Un code (n, l) pour ce canal est un couple (W, g) où

1. $W = \{w_1, \ldots, w_n\}$ est un ensemble de mots de longueur l dans l'alphabet d'entrée du canal Ω_X, appelés mots du code.
2. g est une règle de décodage

$$g : (\Omega_Y)^l \to W$$

qui associe à toute séquence reçue de longueur l dans l'alphabet de sortie Ω_Y un des mots du code.

Les mots du code w_i peuvent être associés aux symboles de la source, x_i ou aux blocks de symboles de la source de longueur k (autrement dit aux symbole d'une extension de la source X).

Notion de code de canal

Définition

Soit un canal $X, Y, P(Y|X)$. Un code (n, l) pour ce canal est un couple (W, g) où

1. $W = \{w_1, \ldots, w_n\}$ est un ensemble de mots de longueur l dans l'alphabet d'entrée du canal Ω_X, appelés mots du code.
2. g est une règle de décodage

$$g : (\Omega_Y)^l \rightarrow W$$

qui associe à toute séquence reçue de longueur l dans l'alphabet de sortie Ω_Y un des mots du code.

Les mots du code w_i peuvent être associés aux symboles de la source, x_i ou aux blocks de symboles de la source de longueur k (autrement dit aux symbole d'une extension de la source X).

Notion de code de canal

Définition

Soit un canal $X, Y, P(Y|X)$. Un code (n, l) pour ce canal est un couple (W, g) où

1. $W = \{w_1, \ldots, w_n\}$ est un ensemble de mots de longueur l dans l'alphabet d'entrée du canal Ω_X, appelés mots du code.
2. g est une règle de décodage

$$g : (\Omega_Y)^l \rightarrow W$$

qui associe à toute séquence reçue de longueur l dans l'alphabet de sortie Ω_Y un des mots du code.

Les mots du code w_i peuvent être associés aux symboles de la source, x_i ou aux blocks de symboles de la source de longueur k (autrement dit aux symbole d'une extension de la source X).

Notion de code de canal

Définition

Soit un canal $X, Y, P(Y|X)$. Un code (n, l) pour ce canal est un couple (W, g) où

1. $W = \{w_1, \ldots, w_n\}$ est un ensemble de mots de longueur l dans l'alphabet d'entrée du canal Ω_X, appelés mots du code.
2. g est une règle de décodage

$$g : (\Omega_Y)^l \rightarrow W$$

qui associe à toute séquence reçue de longueur l dans l'alphabet de sortie Ω_Y un des mots du code.

Les mots du code w_i peuvent être associés aux symboles de la source, x_i ou aux blocks de symboles de la source de longueur k (autrement dit aux symbole d'une extension de la source X).

Probabilité d'erreur conditionnelle

Définition

Soient un canal $X, Y, P(Y|X)$ et un code (n, l) pour ce canal (W, g). Pour chaque mot du code w_i on définit la probabilité d'erreur conditionnelle

$$\lambda_i^{(l)} = P[E|w_i]$$

Probabilité d'erreur moyenne et maximale

Probabilité d'erreur moyenne

Soient un canal $X, Y, P(Y|X)$ et un code (n, l) pour ce canal (W, g). On définit la probabilité d'erreur moyenne **(algébrique)** du code par :

$$\bar{\lambda}^{(l)} = \frac{1}{n} \sum_{i=1}^{n} \lambda_i^{(l)}.$$

Cette moyenne est définie en supposant que le choix d'un mot de code à transmettre est fait selon une distribution uniforme (tous les mots du code sont équiprobables)

Probabilité d'erreur maximale

$$\lambda^{(l)} = \max_{i=1,\ldots,n} \lambda_i^{(l)}.$$

Probabilité d'erreur moyenne et maximale

Probabilité d'erreur moyenne

Soient un canal $X, Y, P(Y|X)$ et un code (n, l) pour ce canal (W, g). On définit la probabilité d'erreur moyenne **(algébrique)** du code par :

$$\bar{\lambda}^{(l)} = \frac{1}{n} \sum_{i=1}^{n} \lambda_i^{(l)}.$$

Cette moyenne est définie en supposant que le choix d'un mot de code à transmettre est fait selon une distribution uniforme (tous les mots du code sont équiprobables)

Probabilité d'erreur maximale

$$\lambda^{(l)} = \max_{i=1,\ldots,n} \lambda_i^{(l)}.$$

Probabilité d'erreur moyenne et maximale

Probabilité d'erreur moyenne

Soient un canal $X, Y, P(Y|X)$ et un code (n, l) pour ce canal (W, g). On définit la probabilité d'erreur moyenne **(algébrique)** du code par :

$$\bar{\lambda}^{(l)} = \frac{1}{n} \sum_{i=1}^{n} \lambda_i^{(l)}.$$

Cette moyenne est définie en supposant que le choix d'un mot de code à transmettre est fait selon une distribution uniforme (tous les mots du code sont équiprobables)

Probabilité d'erreur maximale

$$\lambda^{(l)} = \max_{i=1,\ldots,n} \lambda_i^{(l)}.$$

Débit de communication d'un code

Définition

Soient un canal $X, Y, P(Y|X)$ et un code (n, l) pour ce canal (W, g). On définit le débit de communication du code par :

$$R = \frac{log_2(n)}{l}.$$

l'unité de mesure est Shannon par symbole transmis.

Dans l'hypothèse d'équiprobabilité des mots de code $\log_2(n)$ représente l'entropie associée au choix d'un mot du code à transmettre.

Débit de communication d'un code

Définition

Soient un canal $X, Y, P(Y|X)$ et un code (n, l) pour ce canal (W, g). On définit le débit de communication du code par :

$$R = \frac{log_2(n)}{l}.$$

l'unité de mesure est Shannon par symbole transmis.

Dans l'hypothèse d'équiprobabilité des mots de code $\log_2(n)$ représente l'entropie associée au choix d'un mot du code à transmettre.

Second théorème de Shannon

Théorème

Soit un canal $X, Y, P(Y|X)$ de capacité $C > 0$. Pour tout $R < C$ il est possible de trouver un code de canal avec un débit R et une probabilité d'erreur aussi petite que possible. Plus précisément, il existe une suite de codes $(M(l), l)$ tels que $M(l) = 2^{lR}$ telle que

$$\lim_{l \to \infty} \lambda^{(l)} = 0$$

Hypothèses de travail

- On suppose qu'on dispose d'un message à transmettre sous forme d'une suite binaire de longueur M.
- Le message sera découpé en blocks de k bits
- On suppose que tous les blocks de k bits possibles sont équiprobables.
- Le codage consiste à remplacer chaque block de k bits par un mot binaire de longueur $n \geq k$.

Hypothèses de travail

- On suppose qu'on dispose d'un message à transmettre sous forme d'une suite binaire de longueur M.
- Le message sera découpé en blocks de k bits
- On suppose que tous les blocks de k bits possibles sont équiprobables.
- Le codage consiste à remplacer chaque block de k bits par un mot binaire de longueur $n \geq k$.

Hypothèses de travail

- On suppose qu'on dispose d'un message à transmettre sous forme d'une suite binaire de longueur M.
- Le message sera découpé en blocks de k bits
- On suppose que tous les blocks de k bits possibles sont équiprobables.
- Le codage consiste à remplacer chaque block de k bits par un mot binaire de longueur $n \geq k$.

Hypothèses de travail

- On suppose qu'on dispose d'un message à transmettre sous forme d'une suite binaire de longueur M.
- Le message sera découpé en blocks de k bits
- On suppose que tous les blocks de k bits possibles sont équiprobables.
- Le codage consiste à remplacer chaque block de k bits par un mot binaire de longueur $n \geq k$.

Codes de répétition

- chaque bit du message à transmettre sera répété n fois.
- Les blocks ici sont de taille $k = 1$ et les mots de code seront de taille n.
- Prenons l'exemple de $n = 3$.
- La table de code est

s	w
0	000
1	111

Codes de répétition

- chaque bit du message à transmettre sera répété n fois.
- Les blocks ici sont de taille $k = 1$ et les mots de code seront de taille n.
- Prenons l'exemple de $n = 3$.
- La table de code est

s	w
0	000
1	111

Codes de répétition

- chaque bit du message à transmettre sera répété n fois.
- Les blocks ici sont de taille $k = 1$ et les mots de code seront de taille n.
- Prenons l'exemple de $n = 3$.
- La table de code est

s	w
0	000
1	111

Codes de répétition

- chaque bit du message à transmettre sera répété n fois.
- Les blocks ici sont de taille $k = 1$ et les mots de code seront de taille n.
- Prenons l'exemple de $n = 3$.
- La table de code est

s	w
0	000
1	111

Canal de transmission

- Supposons que nous utilisons pour la transmission un canal binaire symétrique dont la probabilité d'erreur est $0 < \alpha < 0.5$.
- La matrice de transmission est alors

$$P(Y|X) = \begin{pmatrix} 1-\alpha & \alpha \\ \alpha & 1-\alpha \end{pmatrix}$$

- Chaque bit transmis a la probabilité α d'être reçu comme son contraire.

Canal de transmission

- Supposons que nous utilisons pour la transmission un canal binaire symétrique dont la probabilité d'erreur est $0 < \alpha < 0.5$.
- La matrice de transmission est alors

$$P(Y|X) = \begin{pmatrix} 1-\alpha & \alpha \\ \alpha & 1-\alpha \end{pmatrix}$$

- Chaque bit transmis a la probabilité α d'être reçu comme son contraire.

Canal de transmission

- Supposons que nous utilisons pour la transmission un canal binaire symétrique dont la probabilité d'erreur est $0 < \alpha < 0.5$.
- La matrice de transmission est alors

$$P(Y|X) = \begin{pmatrix} 1-\alpha & \alpha \\ \alpha & 1-\alpha \end{pmatrix}$$

- Chaque bit transmis a la probabilité α d'être reçu comme son contraire.

Règle de décodage.

- A la réception, le message sera analysé par blocks de 3 bits.
- Compte tenu des erreurs de transmission l'ensemble de mots possibles à la réception contient tous les 8 mots de 3 bits.
- Si on reçoit l'un des mots du code (000 et 111) le décodeur le remplacera pas le symbole correspondant.
- Mais que doit on faire si on reçoit 010 ?
- La règle du vote majoritaire :
  - Si la majorité des 3 bits reçus sont 0 on considère qu'il s'agit du mot de code 000 et on décode donc par 0.
  - Si la majorité des bits sont 1 on considère que le mots de code transmis est 111 est on remplace par 1.

Règle de décodage.

- A la réception, le message sera analysé par blocks de 3 bits.
- Compte tenu des erreurs de transmission l'ensemble de mots possibles à la réception contient tous les 8 mots de 3 bits.
- Si on reçoit l'un des mots du code (000 et 111) le décodeur le remplacera pas le symbole correspondant.
- Mais que doit on faire si on reçoit 010 ?
- La règle du vote majoritaire :
  - Si la majorité des 3 bits reçus sont 0 on considère qu'il s'agit du mot de code 000 et on décode donc par 0.
  - Si la majorité des bits sont 1 on considère que le mots de code transmis est 111 est on remplace par 1.

Règle de décodage.

- A la réception, le message sera analysé par blocks de 3 bits.
- Compte tenu des erreurs de transmission l'ensemble de mots possibles à la réception contient tous les 8 mots de 3 bits.
- Si on reçoit l'un des mots du code (000 et 111) le décodeur le remplacera pas le symbole correspondant.
- Mais que doit on faire si on reçoit 010 ?
- La règle du vote majoritaire :
  - Si la majorité des 3 bits reçus sont 0 on considère qu'il s'agit du mot de code 000 et on décode donc par 0.
  - Si la majorité des bits sont 1 on considère que le mots de code transmis est 111 est on remplace par 1.

Règle de décodage.

- A la réception, le message sera analysé par blocks de 3 bits.
- Compte tenu des erreurs de transmission l'ensemble de mots possibles à la réception contient tous les 8 mots de 3 bits.
- Si on reçoit l'un des mots du code (000 et 111) le décodeur le remplacera pas le symbole correspondant.
- Mais que doit on faire si on reçoit 010 ?
- La règle du vote majoritaire :
  - Si la majorité des 3 bits reçus sont 0 on considère qu'il s'agit du mot de code 000 et on décode donc par 0.
  - Si la majorité des bits sont 1 on considère que le mots de code transmis est 111 est on remplace par 1.

Règle de décodage.

- A la réception, le message sera analysé par blocks de 3 bits.
- Compte tenu des erreurs de transmission l'ensemble de mots possibles à la réception contient tous les 8 mots de 3 bits.
- Si on reçoit l'un des mots du code (000 et 111) le décodeur le remplacera pas le symbole correspondant.
- Mais que doit on faire si on reçoit 010 ?
- La règle du vote majoritaire :
  - Si la majorité des 3 bits reçus sont 0 on considère qu'il s'agit du mot de code 000 et on décode donc par 0.
  - Si la majorité des bits sont 1 on considère que le mots de code transmis est 111 est on remplace par 1.

Règle de décodage.

- A la réception, le message sera analysé par blocks de 3 bits.
- Compte tenu des erreurs de transmission l'ensemble de mots possibles à la réception contient tous les 8 mots de 3 bits.
- Si on reçoit l'un des mots du code (000 et 111) le décodeur le remplacera pas le symbole correspondant.
- Mais que doit on faire si on reçoit 010 ?
- La règle du vote majoritaire :
  - Si la majorité des 3 bits reçus sont 0 on considère qu'il s'agit du mot de code 000 et on décode donc par 0.
  - Si la majorité des bits sont 1 on considère que le mots de code transmis est 111 est on remplace par 1.

Règle de décodage.

- A la réception, le message sera analysé par blocks de 3 bits.
- Compte tenu des erreurs de transmission l'ensemble de mots possibles à la réception contient tous les 8 mots de 3 bits.
- Si on reçoit l'un des mots du code (000 et 111) le décodeur le remplacera pas le symbole correspondant.
- Mais que doit on faire si on reçoit 010 ?
- La règle du vote majoritaire :
  - Si la majorité des 3 bits reçus sont 0 on considère qu'il s'agit du mot de code 000 et on décode donc par 0.
  - Si la majorité des bits sont 1 on considère que le mots de code transmis est 111 est on remplace par 1.

Exemple

Message original	1	0	1	0	1
Message codé	111	000	111	000	111
Message reçu	000	101	110	000	011
Message décodé	0	1	1	0	1

Vote majoritaire vs maximum de vraisemblance

- Considérons un mot reçu $t = t_1 t_2 t_3$ de trois bits.
- Evaluons les probabilités conditionnelles $P(t|s=0)$ et $P(t|s=1)$
-

$$P(t = t_1 t_3 t_3 | s = 0) = P(t = t_1 t_2 t_3 | w = 000) = P(t_1|0)P(t_2|0)P(t_3|0)$$

- D'après la matrice de transmission de notre canal

$$P(t_i|0) = \begin{cases} 1-\alpha, & \text{si } t_i = 0 \text{ transmission correcte} \\ \alpha, & \text{si } t_i = 1 \text{ transmission erronée} \end{cases}, \quad i = 1, 2, 3$$

Vote majoritaire vs maximum de vraisemblance

- Considérons un mot reçu $t = t_1 t_2 t_3$ de trois bits.
- Evaluons les probabilités conditionnelles $P(t|s=0)$ et $P(t|s=1)$
- $$P(t = t_1 t_3 t_3 | s = 0) = P(t = t_1 t_2 t_3 | w = 000) = P(t_1|0)P(t_2|0)P(t_3|0)$$
- D'après la matrice de transmission de notre canal

$$P(t_i|0) = \begin{cases} 1-\alpha, & \text{si } t_i = 0 \text{ transmission correcte} \\ \alpha, & \text{si } t_i = 1 \text{ transmission erronée} \end{cases}, \quad i = 1, 2, 3$$

Vote majoritaire vs maximum de vraisemblance

- Considérons un mot reçu $t = t_1 t_2 t_3$ de trois bits.
- Evaluons les probabilités conditionnelles $P(t|s=0)$ et $P(t|s=1)$
-

$$P(t = t_1 t_3 t_3 | s = 0) = P(t = t_1 t_2 t_3 | w = 000) = P(t_1|0)P(t_2|0)P(t_3|0)$$

- D'après la matrice de transmission de notre canal

$$P(t_i|0) = \begin{cases} 1-\alpha, & \text{si } t_i = 0 \text{ transmission correcte} \\ \alpha, & \text{si } t_i = 1 \text{ transmission erronée} \end{cases}, \quad i = 1, 2, 3$$

Vote majoritaire vs maximum de vraisemblance

- Considérons un mot reçu $t = t_1 t_2 t_3$ de trois bits.
- Evaluons les probabilités conditionnelles $P(t|s = 0)$ et $P(t|s = 1)$
-

$$P(t = t_1 t_3 t_3|s = 0) = P(t = t_1 t_2 t_3|w = 000) = P(t_1|0)P(t_2|0)P(t_3|0)$$

- D'après la matrice de transmission de notre canal

$$P(t_i|0) = \begin{cases} 1-\alpha, & \text{si } t_i = 0 \quad \text{transmission correcte} \\ \alpha, & \text{si } t_i = 1 \quad \text{transmission erronée} \end{cases}, \quad i = 1, 2, 3$$

Vote majoritaire vs maximum de vraisemblance

- Soit l le nombre de 0 dans t. Alors il y a l bits correctes et $3-l$ transmis avec erreur. Alors

$$P(t = t_1 t_3 t_3 | s = 0) = (1-\alpha)^l \alpha^{3-l}$$

- Pour la même séquence $t = t_1 t_2 t_3$ on a alors

$$P(t = t_1 t_3 t_3 | s = 1) = (\alpha)^l (1-\alpha)^{3-l}$$

- car si c'est 1 qui est transmis alors chaque 0 dans t correspond à une erreur, donc à la probabilité α.

Vote majoritaire vs maximum de vraisemblance

- Soit l le nombre de 0 dans t. Alors il y a l bits correctes et $3-l$ transmis avec erreur. Alors

$$P(t = t_1 t_3 t_3 | s = 0) = (1-\alpha)^l \alpha^{3-l}$$

- Pour la même séquence $t = t_1 t_2 t_3$ on a alors

$$P(t = t_1 t_3 t_3 | s = 1) = (\alpha)^l (1-\alpha)^{3-l}$$

- car si c'est 1 qui est transmis alors chaque 0 dans t correspond à une erreur, donc à la probabilité α.

Vote majoritaire vs maximum de vraisemblance

- Soit l le nombre de 0 dans t. Alors il y a l bits correctes et $3-l$ transmis avec erreur. Alors

$$P(t = t_1 t_3 t_3 | s = 0) = (1-\alpha)^l \alpha^{3-l}$$

- Pour la même séquence $t = t_1 t_2 t_3$ on a alors

$$P(t = t_1 t_3 t_3 | s = 1) = (\alpha)^l (1-\alpha)^{3-l}$$

- car si c'est 1 qui est transmis alors chaque 0 dans t correspond à une erreur, donc à la probabilité α.

Vote majoritaire vs maximum de vraisemblance

- Pour comparer les deux probabilités, considérons leur rapport

$$\frac{P(t = t_1 t_3 t_3 | s = 0)}{P(t = t_1 t_3 t_3 | s = 1)} = \left(\frac{1-\alpha}{\alpha}\right)^{l} \left(\frac{\alpha}{1-\alpha}\right)^{3-l} = \left(\frac{1-\alpha}{\alpha}\right)^{2l-3}$$

- Comme $\alpha < 0.5$ la quantité $\frac{1-\alpha}{\alpha} > 1$ est supérieure à 1.
- Si $2l > 3$ **(les 0 sont majoritaires)** , la probabilité de réaliser une telle séquence t, sachant que le mot envoyé est 000 est plus importante que la probabilité de réaliser la même séquence en envoyant le mot 111.
- Si ce sont les 1 qui sont majoritaires alors $2l - 3 < 0$ et on choisit encore des deux hypothèses celle qui rend la séquence de bits reçue la plus probable.
- Ainsi la règle de vote majoritaire coïncide avec celle du maximum de vraisemblance.

Vote majoritaire vs maximum de vraisemblance

- Pour comparer les deux probabilités, considérons leur rapport

$$\frac{P(t = t_1 t_3 t_3 | s = 0)}{P(t = t_1 t_3 t_3 | s = 1)} = \left(\frac{1-\alpha}{\alpha}\right)^{l} \left(\frac{\alpha}{1-\alpha}\right)^{3-l} = \left(\frac{1-\alpha}{\alpha}\right)^{2l-3}$$

- Comme $\alpha < 0.5$ la quantité $\frac{1-\alpha}{\alpha} > 1$ est supérieure à 1.
- Si $2l > 3$ **(les 0 sont majoritaires)** , la probabilité de réaliser une telle séquence t, sachant que le mot envoyé est 000 est plus importante que la probabilité de réaliser la même séquence en envoyant le mot 111.
- Si ce sont les 1 qui sont majoritaires alors $2l - 3 < 0$ et on choisit encore des deux hypothèses celle qui rend la séquence de bits reçue la plus probable.
- Ainsi la règle de vote majoritaire coïncide avec celle du maximum de vraisemblance.

Vote majoritaire vs maximum de vraisemblance

- Pour comparer les deux probabilités, considérons leur rapport

$$\frac{P(t = t_1 t_3 t_3 | s = 0)}{P(t = t_1 t_3 t_3 | s = 1)} = \left(\frac{1-\alpha}{\alpha}\right)^{l} \left(\frac{\alpha}{1-\alpha}\right)^{3-l} = \left(\frac{1-\alpha}{\alpha}\right)^{2l-3}$$

- Comme $\alpha < 0.5$ la quantité $\frac{1-\alpha}{\alpha} > 1$ est supérieure à 1.
- Si $2l > 3$ **(les 0 sont majoritaires)** , la probabilité de réaliser une telle séquence t, sachant que le mot envoyé est 000 est plus importante que la probabilité de réaliser la même séquence en envoyant le mot 111.
- Si ce sont les 1 qui sont majoritaires alors $2l - 3 < 0$ et on choisit encore des deux hypothèses celle qui rend la séquence de bits reçue la plus probable.
- Ainsi la règle de vote majoritaire coïncide avec celle du maximum de vraisemblance.

Vote majoritaire vs maximum de vraisemblance

- Pour comparer les deux probabilités, considérons leur rapport

$$\frac{P(t = t_1 t_3 t_3 | s = 0)}{P(t = t_1 t_3 t_3 | s = 1)} = \left(\frac{1-\alpha}{\alpha}\right)^{l} \left(\frac{\alpha}{1-\alpha}\right)^{3-l} = \left(\frac{1-\alpha}{\alpha}\right)^{2l-3}$$

- Comme $\alpha < 0.5$ la quantité $\frac{1-\alpha}{\alpha} > 1$ est supérieure à 1.
- Si $2l > 3$ **(les 0 sont majoritaires)** , la probabilité de réaliser une telle séquence t, sachant que le mot envoyé est 000 est plus importante que la probabilité de réaliser la même séquence en envoyant le mot 111.
- Si ce sont les 1 qui sont majoritaires alors $2l - 3 < 0$ et on choisit encore des deux hypothèses celle qui rend la séquence de bits reçue la plus probable.
- Ainsi la règle de vote majoritaire coïncide avec celle du maximum de vraisemblance.

Vote majoritaire vs maximum de vraisemblance

- Pour comparer les deux probabilités, considérons leur rapport

$$\frac{P(t = t_1 t_3 t_3 | s = 0)}{P(t = t_1 t_3 t_3 | s = 1)} = \left(\frac{1-\alpha}{\alpha}\right)^{l} \left(\frac{\alpha}{1-\alpha}\right)^{3-l} = \left(\frac{1-\alpha}{\alpha}\right)^{2l-3}$$

- Comme $\alpha < 0.5$ la quantité $\frac{1-\alpha}{\alpha} > 1$ est supérieure à 1.
- Si $2l > 3$ **(les 0 sont majoritaires)** , la probabilité de réaliser une telle séquence t, sachant que le mot envoyé est 000 est plus importante que la probabilité de réaliser la même séquence en envoyant le mot 111.
- Si ce sont les 1 qui sont majoritaires alors $2l - 3 < 0$ et on choisit encore des deux hypothèses celle qui rend la séquence de bits reçue la plus probable.
- Ainsi la règle de vote majoritaire coïncide avec celle du maximum de vraisemblance.

Groupe $G_k = \{0,1\}^k$

- Considérons l'ensemble de séquences binaires de taille k. On le note $G_k = \{0,1\}^k$.
- On a $Card(G_k) = 2^k$.
- On pourra appeler les éléments de cet ensemble mots ou vecteurs.
- Sur cet ensemble nous définissons l'opération d'addition modulo 2, notée $\oplus$, entre les éléments.
- Cela correspond à effectuer un XOR (un ou exclusif) bit par bit, sachant que

b_1	b_2	$b_1 \oplus b_2$
1	1	0
1	0	1
0	1	1
0	0	0

Groupe $G_k = \{0,1\}^k$

- Considérons l'ensemble de séquences binaires de taille k. On le note $G_k = \{0,1\}^k$.
- On a $Card(G_k) = 2^k$.
- On pourra appeler les éléments de cet ensemble mots ou vecteurs.
- Sur cet ensemble nous définissons l'opération d'addition modulo 2, notée $\oplus$, entre les éléments.
- Cela correspond à effectuer un XOR (un ou exclusif) bit par bit, sachant que

b_1	b_2	$b_1 \oplus b_2$
1	1	0
1	0	1
0	1	1
0	0	0

Groupe $G_k = \{0,1\}^k$

- Considérons l'ensemble de séquences binaires de taille k. On le note $G_k = \{0,1\}^k$.
- On a $Card(G_k) = 2^k$.
- On pourra appeler les éléments de cet ensemble mots ou vecteurs.
- Sur cet ensemble nous définissons l'opération d'addition modulo 2, notée $\oplus$, entre les éléments.
- Cela correspond à effectuer un XOR (un ou exclusif) bit par bit, sachant que

b_1	b_2	$b_1 \oplus b_2$
1	1	0
1	0	1
0	1	1
0	0	0

Groupe $G_k = \{0,1\}^k$

- Considérons l'ensemble de séquences binaires de taille k. On le note $G_k = \{0,1\}^k$.
- On a $Card(G_k) = 2^k$.
- On pourra appeler les éléments de cet ensemble mots ou vecteurs.
- Sur cet ensemble nous définissons l'opération d'addition modulo 2, notée $\oplus$, entre les éléments.
- Cela correspond à effectuer un XOR (un ou exclusif) bit par bit, sachant que

b_1	b_2	$b_1 \oplus b_2$
1	1	0
1	0	1
0	1	1
0	0	0

Groupe $G_k = \{0,1\}^k$

- Considérons l'ensemble de séquences binaires de taille k. On le note $G_k = \{0,1\}^k$.
- On a $Card(G_k) = 2^k$.
- On pourra appeler les éléments de cet ensemble mots ou vecteurs.
- Sur cet ensemble nous définissons l'opération d'addition modulo 2, notée $\oplus$, entre les éléments.
- Cela correspond à effectuer un XOR (un ou exclusif) bit par bit, sachant que

b_1	b_2	$b_1 \oplus b_2$
1	1	0
1	0	1
0	1	1
0	0	0

Groupe $G_k = \{0,1\}^k$

- Muni de cette loi additive, G_k est un groupe abélien d'élément neutre $(0,\ldots,0)$.
- On le munit également de loi externe multiplicative selon la table de ET logique :

b_1	b_2	$b_1 \otimes b_2$
1	1	1
1	0	0
0	1	0
0	0	0

- G_k est un espace vectoriel.

Groupe $G_k = \{0,1\}^k$

- Muni de cette loi additive, G_k est un groupe abélien d'élément neutre $(0,\ldots,0)$.
- On le munit également de loi externe multiplicative selon la table de ET logique :

b_1	b_2	$b_1 \otimes b_2$
1	1	1
1	0	0
0	1	0
0	0	0

- G_k est un espace vectoriel.

Groupe $G_k = \{0,1\}^k$

- Muni de cette loi additive, G_k est un groupe abélien d'élément neutre $(0,\ldots,0)$.
- On le munit également de loi externe multiplicative selon la table de ET logique :

b_1	b_2	$b_1 \otimes b_2$
1	1	1
1	0	0
0	1	0
0	0	0

- G_k est un espace vectoriel.

Distance de Hamming

Définition (Distance de Hamming)

Soient $v = (v_1, \ldots, v_k)$, $w = (w_1, \ldots, w_k) \in G_k$ deux vecteurs de G_k. On appelle distance de Hamming le nombre de bits différents entre v et w

$$d_h(v, w) = Card\{i \in [1, \ldots, n] \mid v_i \neq w_i\}$$

Topologie

- on peut définir la notion de boule ouverte de rayon r centrée en $w \in G_k$ de façon suivante :

$$B_h(w, r) = \{v \in G_k, \quad d_h(w, v) < r\} \tag{.1}$$

- Boule fermée :

$$\overline{B_h}(w, r) = \{v \in G_k, \quad d_h(w, v) \leq r\} \tag{.2}$$

Topologie

- on peut définir la notion de boule ouverte de rayon r centrée en $w \in G_k$ de façon suivante :

$$B_h(w, r) = \{v \in G_k, \quad d_h(w, v) < r\} \tag{.1}$$

- Boule fermée :

$$\overline{B_h}(w, r) = \{v \in G_k, \quad d_h(w, v) \leq r\} \tag{.2}$$

Topologie

- Une boule ouverte $B_h(w, r)$ définit l'ensemble de mots binaires $v \in G_k$ qui diffèrent de w en moins de r bits.
- Par exemple, si $k = 4$ et $w = 1010$ alors

$$B_h(w, 2) = \{1010, 0010, 1110, 1000, 1010\}$$

C'est l'ensemble de mots que l'on peut obtenir à partir de w en changeant 0 ou 1 bits à la fois.

Topologie

- Une boule ouverte $B_h(w, r)$ définit l'ensemble de mots binaires $v \in G_k$ qui diffèrent de w en moins de r bits.
- Par exemple, si $k = 4$ et $w = 1010$ alors

$$B_h(w, 2) = \{1010, 0010, 1110, 1000, 1010\}$$

C'est l'ensemble de mots que l'on peut obtenir à partir de w en changeant 0 ou 1 bits à la fois.

Poids

Définition (Poids d'un vecteur)

Soit $v = (v_1, \ldots, v_k) \in G_k$. On appelle poids de v le nombre de bits différents de 0 dans v :

$$\omega(v) = Card\{i \in [1, \ldots, n] \mid v_i = 1\}$$

Exemple

- Soit $k = 5$. G_5 contient 2^k mots binaires de longueur 5. Soient

$$v = 00110, \quad w = 10101$$

- On a alors

$$d_h(w, v) = 3, \quad w \oplus v = 10011, \quad \omega(w \oplus v) = 3$$

Exemple

- Soit $k = 5$. G_5 contient 2^k mots binaires de longueur 5. Soient

$$v = 00110, \quad w = 10101$$

- On a alors

$$d_h(w, v) = 3, \quad w \oplus v = 10011, \quad \omega(w \oplus v) = 3$$

Décodage selon le minimum de distance de Hamming

Définition

Soit une suite de mots $C = \{w_1, \ldots, w_M\} \subset G_n$ de longueur n qui forment n code. Soit $z \in G_n$ un mot de longueur n reçu après transmission à travers d'un canal bruité d'un des mots du code C. la règle de décodage selon le minimum de distance de Hamming associe à z le mot du code w le plus proche de z selon d_h :

$$g(z) = w_0 \in C \quad \text{t.q.} \quad d_h(z, w_0) = \min_{w \in C} d_h(z, w)$$

Code (n, k)

- On appelle **code de paramètres** (k, n) toute application injective $\phi : G_k \longrightarrow G_n$.
- Le paramètre k s'appelle **dimension du code** Le paramètre n s'appelle **longueur du code**
- L'ensemble image de ϕ,

$$C = \{\phi(m), \quad m \in G_k\}$$

s'appelle **image de code**. Ses éléments s'appellent **mots de code**. Les éléments de G_k s'appellent **mots de source**.

Code (n, k)

- On appelle **code de paramètres** (k, n) toute application injective $\phi : G_k \longrightarrow G_n$.
- Le paramètre k s'appelle **dimension du code** Le paramètre n s'appelle **longueur du code**
- L'ensemble image de ϕ,

$$C = \{\phi(m), \quad m \in G_k\}$$

s'appelle **image de code**. Ses éléments s'appellent **mots de code**. Les éléments de G_k s'appellent **mots de source**.

Code (n, k)

- On appelle **code de paramètres** (k, n) toute application injective $\phi : G_k \longrightarrow G_n$.
- Le paramètre k s'appelle **dimension du code** Le paramètre n s'appelle **longueur du code**
- L'ensemble image de ϕ,

$$C = \{\phi(m), \quad m \in G_k\}$$

s'appelle **image de code**. Ses éléments s'appellent **mots de code**. Les éléments de G_k s'appellent **mots de source**.

Distance minimale d'un code

Définition

Soit un code de paramètres (k, n) dont l'image est l'ensemble de mots C. On appelle distance minimale d'un code la valeur

$$d = min_{w,v \in C} d_h(w, v)$$

c'est la plus petite distance de Hamming entre deux mots du code.

Détection et non détection des erreurs

- Soit un code (k, n) d'image C.
- Supposons qu'un mot du code $w \in C$ est transmis. On reçoit le mot $z \in G_n$.
- Si z coïncide avec un des mots de code, $v \in C$, il sera décodé comme v.
- Cela peut se produire si les erreurs se produisent exactement sur les bits qui sont différents entre le mot transmis, w et le mot du code reçu, v.
- Donc s'il y a $d_h(w, v)$ erreurs et si elles se produisent "aux bons endroits", elles sont **indétectables**.
- On détecte les erreurs si le mot reçu, z ne correspond à aucun des mots de code.

Détection et non détection des erreurs

- Soit un code (k, n) d'image C.
- Supposons qu'un mot du code $w \in C$ est transmis. On reçoit le mot $z \in G_n$.
- Si z coïncide avec un des mots de code, $v \in C$, il sera décodé comme v.
- Cela peut se produire si les erreurs se produisent exactement sur les bits qui sont différents entre le mot transmis, w et le mot du code reçu, v.
- Donc s'il y a $d_h(w, v)$ erreurs et si elles se produisent "aux bons endroits", elles sont **indétectables.**
- On détecte les erreurs si le mot reçu, z ne correspond à aucun des mots de code.

Détection et non détection des erreurs

- Soit un code (k, n) d'image C.
- Supposons qu'un mot du code $w \in C$ est transmis. On reçoit le mot $z \in G_n$.
- Si z coïncide avec un des mots de code, $v \in C$, il sera décodé comme v.
- Cela peut se produire si les erreurs se produisent exactement sur les bits qui sont différents entre le mot transmis, w et le mot du code reçu, v.
- Donc s'il y a $d_h(w, v)$ erreurs et si elles se produisent "aux bons endroits", elles sont **indétectables.**
- On détecte les erreurs si le mot reçu, z ne correspond à aucun des mots de code.

Détection et non détection des erreurs

- Soit un code (k, n) d'image C.
- Supposons qu'un mot du code $w \in C$ est transmis. On reçoit le mot $z \in G_n$.
- Si z coïncide avec un des mots de code, $v \in C$, il sera décodé comme v.
- Cela peut se produire si les erreurs se produisent exactement sur les bits qui sont différents entre le mot transmis, w et le mot du code reçu, v.
- Donc s'il y a $d_h(w, v)$ erreurs et si elles se produisent "aux bons endroits", elles sont **indétectables.**
- On détecte les erreurs si le mot reçu, z ne correspond à aucun des mots de code.

Détection et non détection des erreurs

- Soit un code (k, n) d'image C.
- Supposons qu'un mot du code $w \in C$ est transmis. On reçoit le mot $z \in G_n$.
- Si z coïncide avec un des mots de code, $v \in C$, il sera décodé comme v.
- Cela peut se produire si les erreurs se produisent exactement sur les bits qui sont différents entre le mot transmis, w et le mot du code reçu, v.
- Donc s'il y a $d_h(w, v)$ erreurs et si elles se produisent "aux bons endroits", elles sont **indétectables.**
- On détecte les erreurs si le mot reçu, z ne correspond à aucun des mots de code.

Détection et non détection des erreurs

- Soit un code (k, n) d'image C.
- Supposons qu'un mot du code $w \in C$ est transmis. On reçoit le mot $z \in G_n$.
- Si z coïncide avec un des mots de code, $v \in C$, il sera décodé comme v.
- Cela peut se produire si les erreurs se produisent exactement sur les bits qui sont différents entre le mot transmis, w et le mot du code reçu, v.
- Donc s'il y a $d_h(w, v)$ erreurs et si elles se produisent "aux bons endroits", elles sont **indétectables.**
- **On détecte les erreurs si le mot reçu, z ne correspond à aucun des mots de code.**

Capacité de détection

- Soit d est la distance minimale du code.
- Alors pour obtenir à partir d'un mot du code $w \in C$ un autre mot du code, $v \in C$ il faut changer au moins d bits.
- le nombre maximal d'erreurs qu'il est possible de détecter quels que soient les bits où elles se produisent est égal à $d-1$
- On appelle capacité de détection d'erreurs d'un code le nombre maximal e_d d'erreurs qu'il est **toujours** possible de détecter. On a

$$e_d = d - 1$$

Capacité de détection

- Soit d est la distance minimale du code.
- Alors pour obtenir à partir d'un mot du code $w \in C$ un autre mot du code, $v \in C$ il faut changer au moins d bits.
- le nombre maximal d'erreurs qu'il est possible de détecter quels que soient les bits où elles se produisent est égal à $d-1$
- On appelle capacité de détection d'erreurs d'un code le nombre maximal e_d d'erreurs qu'il est **toujours** possible de détecter. On a

$$e_d = d - 1$$

Capacité de détection

- Soit d est la distance minimale du code.
- Alors pour obtenir à partir d'un mot du code $w \in C$ un autre mot du code, $v \in C$ il faut changer au moins d bits.
- le nombre maximal d'erreurs qu'il est possible de détecter quels que soient les bits où elles se produisent est égal à $d - 1$
- On appelle capacité de détection d'erreurs d'un code le nombre maximal e_d d'erreurs qu'il est **toujours** possible de détecter. On a

$$e_d = d - 1$$

Capacité de détection

- Soit d est la distance minimale du code.
- Alors pour obtenir à partir d'un mot du code $w \in C$ un autre mot du code, $v \in C$ il faut changer au moins d bits.
- le nombre maximal d'erreurs qu'il est possible de détecter quels que soient les bits où elles se produisent est égal à $d - 1$
- On appelle capacité de détection d'erreurs d'un code le nombre maximal e_d d'erreurs qu'il est **toujours** possible de détecter. On a

$$e_d = d - 1$$

Correction

- Lorsque les erreurs sont détectées on ne sait pas quels sont les bits erronés et combien sont ils.
- La règle de décodage du minimum de distance de Hamming va alors remplacer le mot reçu par celui des mots du code qui est le plus proche
- Si cette action conduit au mot qui a réellement été transmis, elle équivaut à la correction des erreurs.

Correction

- Lorsque les erreurs sont détectées on ne sait pas quels sont les bits erronés et combien sont ils.
- La règle de décodage du minimum de distance de Hamming va alors remplacer le mot reçu par celui des mots du code qui est le plus proche
- Si cette action conduit au mot qui a réellement été transmis, elle équivaut à la correction des erreurs.

Correction

- Lorsque les erreurs sont détectées on ne sait pas quels sont les bits erronés et combien sont ils.
- La règle de décodage du minimum de distance de Hamming va alors remplacer le mot reçu par celui des mots du code qui est le plus proche
- Si cette action conduit au mot qui a réellement été transmis, elle équivaut à la correction des erreurs.

Capacité de correction

Définition

On appelle capacité de correction d'erreurs d'un code le nombre maximal e_c d'erreurs qu'il est **toujours** possible de corriger. On a

$$e_c = E\left[\frac{d-1}{2}\right]$$

où d est la distance minimale du code, $E[]$ est la partie entière d'un réel.

Code correcteur linéaire (n, k)

Définition

Un code linéaire (n, k) est un code de classe (n, k) tel que l'application $\phi : G_k \longrightarrow G_n$ qui le génère est une application linéaire, c'est-à-dire, tel qu'il existe une matrice $G \in \mathcal{M}^{n,k}(0, 1)$ telle que

$$\phi(m) = Gm, \quad \forall m = \begin{pmatrix} m_1 \\ \vdots \\ m_m \end{pmatrix}$$

On appelle G la matrice génératrice du code.

Exemple

- Les coefficients de la matrice G prennent les valeurs dans $\{0,1\}$.
- La multiplication matricielle se fait selon les opérations $\oplus$, $\otimes$ sur $\{0,1\}$, c'est à dire dans l'arithmétique modulo 2.
- $$\begin{pmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} (1 \otimes 1) \oplus (1 \otimes 1) \\ (0 \otimes 1) \oplus (1 \otimes 1) \\ (1 \otimes 1) \oplus (0 \otimes 1) \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}$$

Exemple

- Les coefficients de la matrice G prennent les valeurs dans $\{0,1\}$.
- La multiplication matricielle se fait selon les opérations $\oplus$, $\otimes$ sur $\{0,1\}$, c'est à dire dans l'arithmétique modulo 2.
- $$\begin{pmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} (1 \otimes 1) \oplus (1 \otimes 1) \\ (0 \otimes 1) \oplus (1 \otimes 1) \\ (1 \otimes 1) \oplus (0 \otimes 1) \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}$$

Exemple

- Les coefficients de la matrice G prennent les valeurs dans $\{0,1\}$.
- La multiplication matricielle se fait selon les opérations $\oplus$, $\otimes$ sur $\{0,1\}$, c'est à dire dans l'arithmétique modulo 2.
- $$\begin{pmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} (1 \otimes 1) \oplus (1 \otimes 1) \\ (0 \otimes 1) \oplus (1 \otimes 1) \\ (1 \otimes 1) \oplus (0 \otimes 1) \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}$$

Code systématique

Définition

On dit qu'un code linéaire (k, n) est systématique ssi sa matrice génératrice G est de la forme (en blocks)

$$G = \begin{pmatrix} I_k \\ G' \end{pmatrix}$$

où I_k est la matrice identité de dimension k et G' est une matrice de $n - k$ lignes et k colonnes.

Distance minimale d'un code linéaire

- La distance minimale d'un code linéaire est égale au plus petit poids non nul des mots de code.
- **Borne de Singleton :** La distance minimale d'un code linéaire de dimension k et de longueur n vérifie

$$d \leq n + 1 - k$$

Distance minimale d'un code linéaire

- La distance minimale d'un code linéaire est égale au plus petit poids non nul des mots de code.
- **Borne de Singleton :** La distance minimale d'un code linéaire de dimension k et de longueur n vérifie

$$d \leq n + 1 - k$$

Matrice de contrôle d'un code linéaire

Définition

Soit un code linéaire (k, n) de matrice génératrice G et d'image C. On appelle matrice de contrôle d ce code toute matrice $H \in \mathcal{M}_{n-k,n}$ telle que pour tout mot de code $w = \begin{pmatrix} w_1 \\ \vdots \\ w_n \end{pmatrix}$ on ait

$$Hw = 0$$

Autrement dit, H est une matrice dont le noyau coïncide avec l'ensemble de mots de code C.

Matrice de contrôle d'un code systématique

Proposition

Soit un code linéaire (k, n) systématique de matrice génératrice $G = \begin{pmatrix} I_k \\ G' \end{pmatrix}$

Alors sa matrice de contrôle est

$$H = (G' I_{n-k})$$

Syndrome et erreur

Définition

Soit un code linéaire (k, n) de matrice génératrice G et de matrice de contrôle H. Soit $Y \in C$ un mot de code et $Z \in \{0,1\}^n$ un mot reçu après la transmission de Y par un canal bruité.
On appelle **mot erreur** le mot $E \in \{0,1\}^n$ tel que

$$Z = Y + E$$

On appelle **le mot syndrome** de Z

$$S = HZ$$

Utilisation du syndrome

- Chaque bit de mot erreur E est nul si les bits correspondants de Z et Y coïncident, et est égal à 1 s'ils sont différents. Ainsi

$$\omega(E) = d_h(Y, Z)$$

- Si on a trouvé le mot erreur E on peut retrouver Y en posant :

$$Y = Z + E$$

- Donc, pour corriger les erreurs de transmission, il suffit de trouver le vecteur E.

Utilisation du syndrome

- Chaque bit de mot erreur E est nul si les bits correspondants de Z et Y coïncident, et est égal à 1 s'ils sont différents. Ainsi

$$\omega(E) = d_h(Y, Z)$$

- Si on a trouvé le mot erreur E on peut retrouver Y en posant :

$$Y = Z + E$$

- Donc, pour corriger les erreurs de transmission, il suffit de trouver le vecteur E.

Utilisation du syndrome

- Chaque bit de mot erreur E est nul si les bits correspondants de Z et Y coïncident, et est égal à 1 s'ils sont différents. Ainsi

$$\omega(E) = d_h(Y, Z)$$

- Si on a trouvé le mot erreur E on peut retrouver Y en posant :

$$Y = Z + E$$

- Donc, pour corriger les erreurs de transmission, il suffit de trouver le vecteur E.

Utilisation du syndrome

- Si le syndrome de Z $S = HZ$ est nul c'est que Z est un mot de code (par définition de la matrice de contrôle).
- Il sera alors décodé par le block correspondant dans la table du code. Ainsi, **le syndrome nul correspond à l'absence ou à la non détection des erreurs de transmission.**
- Le syndrome non nul équivaut la détection d'erreurs, car cela signifie que le mot reçu Z n'est pas un mot de code.

Utilisation du syndrome

- Si le syndrome de Z $S = HZ$ est nul c'est que Z est un mot de code (par définition de la matrice de contrôle).
- Il sera alors décodé par le block correspondant dans la table du code. Ainsi, **le syndrome nul correspond à l'absence ou à la non détection des erreurs de transmission.**
- Le syndrome non nul équivaut la détection d'erreurs, car cela signifie que le mot reçu Z n'est pas un mot de code.

Utilisation du syndrome

- Si le syndrome de Z $S = HZ$ est nul c'est que Z est un mot de code (par définition de la matrice de contrôle).
- Il sera alors décodé par le block correspondant dans la table du code. Ainsi, **le syndrome nul correspond à l'absence ou à la non détection des erreurs de transmission.**
- Le syndrome non nul équivaut la détection d'erreurs, car cela signifie que le mot reçu Z n'est pas un mot de code.

Proposition

Proposition

Soient Y le mot transmis et Z le mot reçu. Soient $E = Z + Y$ le mot erreur et $S = HZ$ le syndrome de Z. Alors le mot erreur a le même syndrome que Z : $HE = HZ$

Preuve

- Par définition de la matrice de contrôle on a

$$HY = 0$$

- car Y est un mot de code.
- Alors

$$HE = H(Y + Z) = HZ = S$$

Preuve

- Par définition de la matrice de contrôle on a

$$HY = 0$$

- car Y est un mot de code.
- Alors

$$HE = H(Y + Z) = HZ = S$$

Preuve

- Par définition de la matrice de contrôle on a

$$HY = 0$$

- car Y est un mot de code.
- Alors

$$HE = H(Y + Z) = HZ = S$$

Classe latérale

Définition

Soit un code linéaire (k, n) de matrice de contrôle H. Soit Z un mot de $\{0,1\}^n$. On appelle classe latérale de Z l'ensemble de mots ayant le même syndrome que Z :

$$L(Z) = \{W \in \{0,1\}^n, \quad HW = WZ = S\}$$

Vers l'algorithme de décodage

- le vecteur erreur E pour la correction du mot reçu, Z se trouve dans la classe latérale de Z.
- Rappelons que le décodage se fera selon le principe de minimum de distance de Hamming.
- Le poids du vecteur erreur $E = Z + W$ est égal à la distance de Hamming entre Z et W.
- Le mot reçu Z sera remplacé par le mot de code W le plus proche. Or,

$$\forall W \in C, \quad E_W = Z + W \in L(Z)$$

- Et réciproquement,

$$\forall E \in L(Z), \quad W = Z + E \in C$$

- Donc trouver le mot de code le plus proche du mot reçu Z équivaut à trouver l'élément $E \in L(Z)$ de plus petit poids.

Vers l'algorithme de décodage

- le vecteur erreur E pour la correction du mot reçu, Z se trouve dans la classe latérale de Z.
- Rappelons que le décodage se fera selon le principe de minimum de distance de Hamming.
- Le poids du vecteur erreur $E = Z + W$ est égal à la distance de Hamming entre Z et W.
- Le mot reçu Z sera remplacé par le mot de code W le plus proche. Or,

$$\forall W \in C, \quad E_W = Z + W \in L(Z)$$

- Et réciproquement,

$$\forall E \in L(Z), \quad W = Z + E \in C$$

- Donc trouver le mot de code le plus proche du mot reçu Z équivaut à trouver l'élément $E \in L(Z)$ de plus petit poids.

Vers l'algorithme de décodage

- le vecteur erreur E pour la correction du mot reçu, Z se trouve dans la classe latérale de Z.
- Rappelons que le décodage se fera selon le principe de minimum de distance de Hamming.
- Le poids du vecteur erreur $E = Z + W$ est égal à la distance de Hamming entre Z et W.
- Le mot reçu Z sera remplacé par le mot de code W le plus proche. Or,

$$\forall W \in C, \quad E_W = Z + W \in L(Z)$$

- Et réciproquement,

$$\forall E \in L(Z), \quad W = Z + E \in C$$

- Donc trouver le mot de code le plus proche du mot reçu Z équivaut à trouver l'élément $E \in L(Z)$ de plus petit poids.

Vers l'algorithme de décodage

- le vecteur erreur E pour la correction du mot reçu, Z se trouve dans la classe latérale de Z.
- Rappelons que le décodage se fera selon le principe de minimum de distance de Hamming.
- Le poids du vecteur erreur $E = Z + W$ est égal à la distance de Hamming entre Z et W.
- Le mot reçu Z sera remplacé par le mot de code W le plus proche. Or,

$$\forall W \in C, \quad E_W = Z + W \in L(Z)$$

- Et réciproquement,

$$\forall E \in L(Z), \quad W = Z + E \in C$$

- Donc trouver le mot de code le plus proche du mot reçu Z équivaut à trouver l'élément $E \in L(Z)$ de plus petit poids.

Vers l'algorithme de décodage

- le vecteur erreur E pour la correction du mot reçu, Z se trouve dans la classe latérale de Z.
- Rappelons que le décodage se fera selon le principe de minimum de distance de Hamming.
- Le poids du vecteur erreur $E = Z + W$ est égal à la distance de Hamming entre Z et W.
- Le mot reçu Z sera remplacé par le mot de code W le plus proche. Or,

$$\forall W \in C, \quad E_W = Z + W \in L(Z)$$

- Et réciproquement,

$$\forall E \in L(Z), \quad W = Z + E \in C$$

- Donc trouver le mot de code le plus proche du mot reçu Z équivaut à trouver l'élément $E \in L(Z)$ de plus petit poids.

Vers l'algorithme de décodage

- le vecteur erreur E pour la correction du mot reçu, Z se trouve dans la classe latérale de Z.
- Rappelons que le décodage se fera selon le principe de minimum de distance de Hamming.
- Le poids du vecteur erreur $E = Z + W$ est égal à la distance de Hamming entre Z et W.
- Le mot reçu Z sera remplacé par le mot de code W le plus proche. Or,

$$\forall W \in C, \quad E_W = Z + W \in L(Z)$$

- Et réciproquement,

$$\forall E \in L(Z), \quad W = Z + E \in C$$

- Donc trouver le mot de code le plus proche du mot reçu Z équivaut à trouver l'élément $E \in L(Z)$ de plus petit poids.

Vers l'algorithme de décodage

Proposition

Soit un code linéaire de matrice de contrôle H. Le décodage au sens de minimum de distance de hamming d'un mot Z revient à trouver dans la classe latérale de Z le mot de plus petit poids, E_0 et à remplacer Z par le mot de code $W = Z + E \in C$.

En pratique

- On explore $L(Z)$ dans l'ordre croissant des poids de mots.
- Pour faciliter les explications nous allons introduire une famille de mots particuliers :

$$E_i = (00\ldots010\ldots00)$$

- un mot E de poids p est égal à la somme de p mots parmi les $E_i,\ i = 1, \ldots, n$

$$\omega(E) = p \quad \Leftrightarrow \quad \exists i_1, \ldots, i_p, \quad t.q. \quad E = \sum_{j=1}^{p} E_{i_j}$$

- Alors le résultat de multiplication HE est une combinaison linéaire des colonnes de la matrice H, $C_1, \ldots, C_n$:

$$HE = \sum_{j=1}^{p} C_{i_j}$$

car les autres bits de E sont nuls.

En pratique

- On explore $L(Z)$ dans l'ordre croissant des poids de mots.
- Pour faciliter les explications nous allons introduire une famille de mots particuliers :

$$E_i = (00\ldots010\ldots00)$$

- un mot E de poids p est égal à la somme de p mots parmi les $E_i,\ i = 1, \ldots, n$

$$\omega(E) = p \quad \Leftrightarrow \quad \exists i_1, \ldots, i_p, \quad t.q. \quad E = \sum_{j=1}^{p} E_{i_j}$$

- Alors le résultat de multiplication HE est une combinaison linéaire des colonnes de la matrice H, $C_1, \ldots, C_n$:

$$HE = \sum_{j=1}^{p} C_{i_j}$$

car les autres bits de E sont nuls.

En pratique

- On explore $L(Z)$ dans l'ordre croissant des poids de mots.
- Pour faciliter les explications nous allons introduire une famille de mots particuliers :

$$E_i = (00 \ldots 010 \ldots 00)$$

- un mot E de poids p est égal à la somme de p mots parmi les $E_i,\ i = 1, \ldots, n$

$$\omega(E) = p \quad \Leftrightarrow \quad \exists i_1, \ldots, i_p, \quad t.q. \quad E = \sum_{j=1}^{p} E_{i_j}$$

- Alors le résultat de multiplication HE est une combinaison linéaire des colonnes de la matrice H, $C_1, \ldots, C_n$:

$$HE = \sum_{j=1}^{p} C_{i_j}$$

car les autres bits de E sont nuls.

En pratique

- On explore $L(Z)$ dans l'ordre croissant des poids de mots.
- Pour faciliter les explications nous allons introduire une famille de mots particuliers :

$$E_i = (00 \ldots 010 \ldots 00)$$

- un mot E de poids p est égal à la somme de p mots parmi les $E_i,\ i = 1, \ldots, n$

$$\omega(E) = p \quad \Leftrightarrow \quad \exists i_1, \ldots, i_p, \quad t.q. \quad E = \sum_{j=1}^{p} E_{i_j}$$

- Alors le résultat de multiplication HE est une combinaison linéaire des colonnes de la matrice H, $C_1, \ldots, C_n$:

$$HE = \sum_{j=1}^{p} C_{i_j}$$

car les autres bits de E sont nuls.

Vers l'algorithme de décodage

- Soit $S = HZ$ le syndrome de Z. Soit $0 \leq p \leq e_c$. (e_c est la capacité de correction du code).
- Dire qu'il existe dans $L(Z)$ un mot E de poids p équivaut à

$$\exists i_1, \ldots, i_p, \quad t.q. \quad E = \sum_{j=1}^{p} E_{i_j} \quad \text{et} \quad HE = \sum_{j=1}^{p} C_{i_j} = S$$

- il existe un choix de p colonnes de la matrice de contrôle H dont la somme est égale à S.
- on commence teste les poids p allant de 0 à e_c jusqu'à ce qu'on trouve une somme de p colonnes de H égale à S.
- Le vecteur erreur est alors égal à la somme correspondante de E_{i_j}.

Vers l'algorithme de décodage

- Soit $S = HZ$ le syndrome de Z. Soit $0 \leq p \leq e_c$. (e_c est la capacité de correction du code).
- Dire qu'il existe dans $L(Z)$ un mot E de poids p équivaut à

$$\exists i_1, \ldots, i_p, \quad t.q. \quad E = \sum_{j=1}^{p} E_{i_j} \quad \text{et} \quad HE = \sum_{j=1}^{p} C_{i_j} = S$$

- il existe un choix de p colonnes de la matrice de contrôle H dont la somme est égale à S.
- on commence teste les poids p allant de 0 à e_c jusqu'à ce qu'on trouve une somme de p colonnes de H égale à S.
- Le vecteur erreur est alors égal à la somme correspondante de E_{i_j}.

Vers l'algorithme de décodage

- Soit $S = HZ$ le syndrome de Z. Soit $0 \leq p \leq e_c$. (e_c est la capacité de correction du code).
- Dire qu'il existe dans $L(Z)$ un mot E de poids p équivaut à

$$\exists i_1, \ldots, i_p, \quad t.q. \quad E = \sum_{j=1}^{p} E_{i_j} \quad \text{et} \quad HE = \sum_{j=1}^{p} C_{i_j} = S$$

- il existe un choix de p colonnes de la matrice de contrôle H dont la somme est égale à S.
- on commence teste les poids p allant de 0 à e_c jusqu'à ce qu'on trouve une somme de p colonnes de H égale à S.
- Le vecteur erreur est alors égal à la somme correspondante de E_{i_j}.

Vers l'algorithme de décodage

- Soit $S = HZ$ le syndrome de Z. Soit $0 \leq p \leq e_c$. (e_c est la capacité de correction du code).
- Dire qu'il existe dans $L(Z)$ un mot E de poids p équivaut à

$$\exists i_1, \ldots, i_p, \quad t.q. \quad E = \sum_{j=1}^{p} E_{i_j} \quad \text{et} \quad HE = \sum_{j=1}^{p} C_{i_j} = S$$

- il existe un choix de p colonnes de la matrice de contrôle H dont la somme est égale à S.
- on commence teste les poids p allant de 0 à e_c jusqu'à ce qu'on trouve une somme de p colonnes de H égale à S.
- Le vecteur erreur est alors égal à la somme correspondante de E_{i_j}.

Vers l'algorithme de décodage

- Soit $S = HZ$ le syndrome de Z. Soit $0 \leq p \leq e_c$. (e_c est la capacité de correction du code).
- Dire qu'il existe dans $L(Z)$ un mot E de poids p équivaut à

$$\exists i_1, \ldots, i_p, \quad t.q. \quad E = \sum_{j=1}^{p} E_{i_j} \quad \text{et} \quad HE = \sum_{j=1}^{p} C_{i_j} = S$$

- il existe un choix de p colonnes de la matrice de contrôle H dont la somme est égale à S.
- on commence teste les poids p allant de 0 à e_c jusqu'à ce qu'on trouve une somme de p colonnes de H égale à S.
- Le vecteur erreur est alors égal à la somme correspondante de E_{i_j}.

Vers l'algorithme de décodage

- Soit, par exemple $p = 0$.
- Si on trouve que $HZ = 0$ alors Z est un mot de code.
- Sinon, on passe à $p = 1$.
- si on trouve une colonne C_i de H telle que $C_i = S$ alors le mot de code corrigé sera $W = Z + E_i$.
- Sinon on passe à $p = 2$.
- S'il existe deux colonnes parmi n de la matrice H telles que $C_i + C_j = S$ alors le mot corrigé sera $W = Z + E_i + E_j$.

Vers l'algorithme de décodage

- Soit, par exemple $p = 0$.
- Si on trouve que $HZ = 0$ alors Z est un mot de code.
- Sinon, on passe à $p = 1$.
- si on trouve une colonne C_i de H telle que $C_i = S$ alors le mot de code corrigé sera $W = Z + E_i$.
- Sinon on passe à $p = 2$.
- S'il existe deux colonnes parmi n de la matrice H telles que $C_i + C_j = S$ alors le mot corrigé sera $W = Z + E_i + E_j$.

Vers l'algorithme de décodage

- Soit, par exemple $p = 0$.
- Si on trouve que $HZ = 0$ alors Z est un mot de code.
- Sinon, on passe à $p = 1$.
- si on trouve une colonne C_i de H telle que $C_i = S$ alors le mot de code corrigé sera $W = Z + E_i$.
- Sinon on passe à $p = 2$.
- S'il existe deux colonnes parmi n de la matrice H telles que $C_i + C_j = S$ alors le mot corrigé sera $W = Z + E_i + E_j$.

Vers l'algorithme de décodage

- Soit, par exemple $p = 0$.
- Si on trouve que $HZ = 0$ alors Z est un mot de code.
- Sinon, on passe à $p = 1$.
- si on trouve une colonne C_i de H telle que $C_i = S$ alors le mot de code corrigé sera $W = Z + E_i$.
- Sinon on passe à $p = 2$.
- S'il existe deux colonnes parmi n de la matrice H telles que $C_i + C_j = S$ alors le mot corrigé sera $W = Z + E_i + E_j$.

Vers l'algorithme de décodage

- Soit, par exemple $p = 0$.
- Si on trouve que $HZ = 0$ alors Z est un mot de code.
- Sinon, on passe à $p = 1$.
- si on trouve une colonne C_i de H telle que $C_i = S$ alors le mot de code corrigé sera $W = Z + E_i$.
- Sinon on passe à $p = 2$.
- S'il existe deux colonnes parmi n de la matrice H telles que $C_i + C_j = S$ alors le mot corrigé sera $W = Z + E_i + E_j$.

Vers l'algorithme de décodage

- Soit, par exemple $p = 0$.
- Si on trouve que $HZ = 0$ alors Z est un mot de code.
- Sinon, on passe à $p = 1$.
- si on trouve une colonne C_i de H telle que $C_i = S$ alors le mot de code corrigé sera $W = Z + E_i$.
- Sinon on passe à $p = 2$.
- S'il existe deux colonnes parmi n de la matrice H telles que $C_i + C_j = S$ alors le mot corrigé sera $W = Z + E_i + E_j$.

Décodage d'un code linéaire : algorithme

Etape 1. $p = 0$. On vérifie si $S = HZ = 0$. Si oui, Z est un mot de code. Sinon, on pose $p = 1$ et on passe à l'étape 2.

Etape 2 Tant que $p \leq e_c$ faire
tester toutes les sommes de p colonnes de H jusqu'à ce que

$$\sum_{j=1}^{p} C_{i_j} = S$$

Si trouvé, renvoyer $W = Z + \sum_{j=1}^{p} E_{i_j}$. Sinon, poser p=p+1.
Fin de tantque.

Etape 3 Si $p = e_c + 1$ et aucun solution n'est trouvée, renvoyer "erreur de décodage".

Décodage d'un code linéaire : algorithme

Etape 1. $p = 0$. On vérifie si $S = HZ = 0$. Si oui, Z est un mot de code. Sinon, on pose $p = 1$ et on passe à l'étape 2.

Etape 2 Tant que $p \leq e_c$ faire
tester toutes les sommes de p colonnes de H jusqu'à ce que

$$\sum_{j=1}^{p} C_{i_j} = S$$

Si trouvé, renvoyer $W = Z + \sum_{j=1}^{p} E_{i_j}$. Sinon, poser p=p+1.
Fin de tantque.

Etape 3 Si $p = e_c + 1$ et aucun solution n'est trouvée, renvoyer "erreur de décodage".

Décodage d'un code linéaire : algorithme

Etape 1. $p = 0$. On vérifie si $S = HZ = 0$. Si oui, Z est un mot de code. Sinon, on pose $p = 1$ et on passe à l'étape 2.

Etape 2 Tant que $p \leq e_c$ faire
tester toutes les sommes de p colonnes de H jusqu'à ce que

$$\sum_{j=1}^{p} C_{i_j} = S$$

Si trouvé, renvoyer $W = Z + \sum_{j=1}^{p} E_{i_j}$. Sinon, poser p=p+1. Fin de tantque.

Etape 3 Si $p = e_c + 1$ et aucun solution n'est trouvée, renvoyer "erreur de décodage".


"That's all Folks!"