Codage
Correction d'erreurs

E. Jeandel

Emmanuel.Jeandel at lif.univ-mrs.fr

Les erreurs

Lors de la transmission/lecture d'informations, plusieurs types d'erreurs peuvent se produire

- On perd un paquet (réseau) ;
- On n'arrive pas à lire un bit (CD rayé) ;
- L'endroit où on lit est bruité.

Pour résoudre ces problèmes, on ne transmet pas l'information directement mais codée.

Codes - Exemples triviaux

- On double chaque bit

$$0 \cdot 0 \cdot 1 \cdot 0 \cdot 0 \cdot 1 \cdot 0 \cdot 1 \cdot 0 \mapsto 00 \cdot 00 \cdot 11 \cdot 00 \cdot 00 \cdot 11 \cdot 00 \cdot 11 \cdot 00$$

- Si un bit est corrompu dans la transmission, on s'en rend compte

$$00 \cdot 01 \cdot 11 \cdot 00 \cdot 00 \cdot 11 \cdot 00 \cdot 11 \cdot 00$$

- mais on ne sait pas lequel, on ne peut pas corriger l'erreur
- Si deux bits sont corrompus, on peut s'en rendre compte

$$00 \cdot 01 \cdot 11 \cdot 00 \cdot 00 \cdot 01 \cdot 00 \cdot 11 \cdot 00$$

- et on peut ne pas s'en rendre compte

$$00 \cdot 11 \cdot 11 \cdot 00 \cdot 00 \cdot 11 \cdot 00 \cdot 11 \cdot 00$$

Codes - Exemples triviaux

- On double chaque bit

$$0 \cdot 0 \cdot 1 \cdot 0 \cdot 0 \cdot 1 \cdot 0 \cdot 1 \cdot 0 \mapsto 00 \cdot 00 \cdot 11 \cdot 00 \cdot 00 \cdot 11 \cdot 00 \cdot 11 \cdot 00$$

- Si un bit est corrompu dans la transmission, on s'en rend compte

$$00 \cdot 01 \cdot 11 \cdot 00 \cdot 00 \cdot 11 \cdot 00 \cdot 11 \cdot 00$$

- mais on ne sait pas lequel, on ne peut pas corriger l'erreur
- Si deux bits sont corrompus, on peut s'en rendre compte

$$00 \cdot 01 \cdot 11 \cdot 00 \cdot 00 \cdot 01 \cdot 00 \cdot 11 \cdot 00$$

- et on peut ne pas s'en rendre compte

$$00 \cdot 11 \cdot 11 \cdot 00 \cdot 00 \cdot 11 \cdot 00 \cdot 11 \cdot 00$$

Codes - Exemples triviaux

- On double chaque bit

$$0 \cdot 0 \cdot 1 \cdot 0 \cdot 0 \cdot 1 \cdot 0 \cdot 1 \cdot 0 \mapsto 00 \cdot 00 \cdot 11 \cdot 00 \cdot 00 \cdot 11 \cdot 00 \cdot 11 \cdot 00$$

- Si un bit est corrompu dans la transmission, on s'en rend compte

$$00 \cdot 01 \cdot 11 \cdot 00 \cdot 00 \cdot 11 \cdot 00 \cdot 11 \cdot 00$$

- mais on ne sait pas lequel, on ne peut pas corriger l'erreur
- Si deux bits sont corrompus, on peut s'en rendre compte

$$00 \cdot 01 \cdot 11 \cdot 00 \cdot 00 \cdot 01 \cdot 00 \cdot 11 \cdot 00$$

- et on peut ne pas s'en rendre compte

$$00 \cdot 11 \cdot 11 \cdot 00 \cdot 00 \cdot 11 \cdot 00 \cdot 11 \cdot 00$$

Codes - Exemples triviaux

- On double chaque bit

$$0 \cdot 0 \cdot 1 \cdot 0 \cdot 0 \cdot 1 \cdot 0 \cdot 1 \cdot 0 \mapsto 00 \cdot 00 \cdot 11 \cdot 00 \cdot 00 \cdot 11 \cdot 00 \cdot 11 \cdot 00$$

- Si un bit est corrompu dans la transmission, on s'en rend compte

$$00 \cdot 01 \cdot 11 \cdot 00 \cdot 00 \cdot 11 \cdot 00 \cdot 11 \cdot 00$$

- mais on ne sait pas lequel, on ne peut pas corriger l'erreur
- Si deux bits sont corrompus, on peut s'en rendre compte

$$00 \cdot 01 \cdot 11 \cdot 00 \cdot 00 \cdot 01 \cdot 00 \cdot 11 \cdot 00$$

- et on peut ne pas s'en rendre compte

$$00 \cdot 11 \cdot 11 \cdot 00 \cdot 00 \cdot 11 \cdot 00 \cdot 11 \cdot 00$$

Codes - Exemples triviaux

- On double chaque bit

$$0 \cdot 0 \cdot 1 \cdot 0 \cdot 0 \cdot 1 \cdot 0 \cdot 1 \cdot 0 \mapsto 00 \cdot 00 \cdot 11 \cdot 00 \cdot 00 \cdot 11 \cdot 00 \cdot 11 \cdot 00$$

- Si un bit est corrompu dans la transmission, on s'en rend compte

$$00 \cdot 01 \cdot 11 \cdot 00 \cdot 00 \cdot 11 \cdot 00 \cdot 11 \cdot 00$$

- mais on ne sait pas lequel, on ne peut pas corriger l'erreur
- Si deux bits sont corrompus, on peut s'en rendre compte

$$00 \cdot 01 \cdot 11 \cdot 00 \cdot 00 \cdot 01 \cdot 00 \cdot 11 \cdot 00$$

- et on peut ne pas s'en rendre compte

$$00 \cdot 11 \cdot 11 \cdot 00 \cdot 00 \cdot 11 \cdot 00 \cdot 11 \cdot 00$$

Codes - Exemples triviaux

- On triple chaque bit
- Si un bit est corrompu dans la transmission, on s'en rend compte

$$000 \cdot 001 \cdot 111 \cdot 000 \cdot 000 \cdot 111 \cdot 000 \cdot 111 \cdot 000$$

- et on sait comment le corriger
- Si deux bits sont corrompus, on peut s'en rendre compte

$$000 \cdot 011 \cdot 111 \cdot 000 \cdot 000 \cdot 111 \cdot 000 \cdot 111 \cdot 000$$

$$000 \cdot 001 \cdot 111 \cdot 000 \cdot 000 \cdot 110 \cdot 000 \cdot 111 \cdot 000$$

- Mais on ne peut pas corriger

Codes - Exemples triviaux

- On triple chaque bit
- Si un bit est corrompu dans la transmission, on s'en rend compte

$$000 \cdot 001 \cdot 111 \cdot 000 \cdot 000 \cdot 111 \cdot 000 \cdot 111 \cdot 000$$

- et on sait comment le corriger
- Si deux bits sont corrompus, on peut s'en rendre compte

$$000 \cdot 011 \cdot 111 \cdot 000 \cdot 000 \cdot 111 \cdot 000 \cdot 111 \cdot 000$$

$$000 \cdot 001 \cdot 111 \cdot 000 \cdot 000 \cdot 110 \cdot 000 \cdot 111 \cdot 000$$

- Mais on ne peut pas corriger

Codes - Exemples triviaux

- On triple chaque bit
- Si un bit est corrompu dans la transmission, on s'en rend compte

$$000 \cdot 001 \cdot 111 \cdot 000 \cdot 000 \cdot 111 \cdot 000 \cdot 111 \cdot 000$$

- et on sait comment le corriger
- Si deux bits sont corrompus, on peut s'en rendre compte

$$000 \cdot 011 \cdot 111 \cdot 000 \cdot 000 \cdot 111 \cdot 000 \cdot 111 \cdot 000$$

$$000 \cdot 001 \cdot 111 \cdot 000 \cdot 000 \cdot 110 \cdot 000 \cdot 111 \cdot 000$$

- Mais on ne peut pas corriger

Codes - Exemples triviaux

- On triple chaque bit
- Si un bit est corrompu dans la transmission, on s'en rend compte

$$000 \cdot 001 \cdot 111 \cdot 000 \cdot 000 \cdot 111 \cdot 000 \cdot 111 \cdot 000$$

- et on sait comment le corriger
- Si deux bits sont corrompus, on peut s'en rendre compte

$$000 \cdot 011 \cdot 111 \cdot 000 \cdot 000 \cdot 111 \cdot 000 \cdot 111 \cdot 000$$

$$000 \cdot 001 \cdot 111 \cdot 000 \cdot 000 \cdot 110 \cdot 000 \cdot 111 \cdot 000$$

- Mais on ne peut pas corriger

Codes - Exemples triviaux

- On triple chaque bit
- Si un bit est corrompu dans la transmission, on s'en rend compte

$$000 \cdot 001 \cdot 111 \cdot 000 \cdot 000 \cdot 111 \cdot 000 \cdot 111 \cdot 000$$

- et on sait comment le corriger
- Si deux bits sont corrompus, on peut s'en rendre compte

$$000 \cdot 011 \cdot 111 \cdot 000 \cdot 000 \cdot 111 \cdot 000 \cdot 111 \cdot 000$$

$$000 \cdot 001 \cdot 111 \cdot 000 \cdot 000 \cdot 110 \cdot 000 \cdot 111 \cdot 000$$

- Mais on ne peut pas corriger

Codes

- Un code de paramètre (n, m) est un procédé qui transforme n bits en $m > n$ bits. On appelle $f(x)$ la fonction
- On appelle mot du code un mot de la forme $f(x)$.
- Un mot x est transmis avec k erreurs si le résultat de la transmission contient k bits différents de leur valeur initiale.
- Un code détecte k erreurs si, lorsqu'un mot $f(x)$ est transmis avec moins de k erreurs, on peut se rendre compte que le message est erroné.
- Un code corrige k erreurs si, lorsqu'un mot $f(x)$ est transmis avec moins de k erreurs, on est capable de retrouver $f(x)$.

Retour à l'exemple

- Le premier code est un code de paramètre $(n, 2n)$ qui détecte une erreur.
- Le deuxième code est un code de paramètre $(n, 3n)$ qui corrige une erreur et en détecte deux.
- Si on corrige une erreur, on en détecte forcément une deuxième

Correction et détection

- Un code qui corrige une erreur peut en détecter deux.
- Preuve par contradiction : Soit $f(x)$ transmis avec deux erreurs, ce qui donne y.

$$\begin{array}{rl} f(x) & 01010100101100 \\ y & 01011100100100 \end{array}$$

- Le code ne se rend pas compte que y est erroné, donc c'est qu'il correspond à un mot $f(w)$.
- On considère un mot z entre $f(x)$ et $f(w)$, qui a un bit de différent avec $f(x)$ et un bit de différent avec $f(w)$

$$\begin{array}{rl} f(x) & 01010100101100 \\ z & 01011100101100 \\ f(w) & 01011100100100 \end{array}$$

- Le code est capable de corriger z..
- ..mais il ne peut pas savoir s'il faut corriger z en $f(x)$ ou en $f(w)$.

Correction et détection

- Un code qui corrige une erreur peut en détecter deux.
- Preuve par contradiction : Soit $f(x)$ transmis avec deux erreurs, ce qui donne y.

$$\begin{array}{rl} f(x) & 01010100101100 \\ y & 01011100100100 \end{array}$$

- Le code ne se rend pas compte que y est erroné, donc c'est qu'il correspond à un mot $f(w)$.
- On considère un mot z entre $f(x)$ et $f(w)$, qui a un bit de différent avec $f(x)$ et un bit de différent avec $f(w)$

$$\begin{array}{rl} f(x) & 01010100101100 \\ z & 01011100101100 \\ f(w) & 01011100100100 \end{array}$$

- Le code est capable de corriger z..
- ..mais il ne peut pas savoir s'il faut corriger z en $f(x)$ ou en $f(w)$.

Correction et détection

- Un code qui corrige une erreur peut en détecter deux.
- Preuve par contradiction : Soit $f(x)$ transmis avec deux erreurs, ce qui donne y.

$$\begin{array}{rl} f(x) & 01010100101100 \\ y & 01011100100100 \end{array}$$

- Le code ne se rend pas compte que y est erroné, donc c'est qu'il correspond à un mot $f(w)$.
- On considère un mot z entre $f(x)$ et $f(w)$, qui a un bit de différent avec $f(x)$ et un bit de différent avec $f(w)$

$$\begin{array}{rl} f(x) & 01010100101100 \\ z & 01011100101100 \\ f(w) & 01011100100100 \end{array}$$

- Le code est capable de corriger z..
- ..mais il ne peut pas savoir s'il faut corriger z en $f(x)$ ou en $f(w)$.

Correction et détection

- Un code qui corrige une erreur peut en détecter deux.
- Preuve par contradiction : Soit $f(x)$ transmis avec deux erreurs, ce qui donne y.

$$\begin{array}{rl} f(x) & 01010100101100 \\ y & 01011100100100 \end{array}$$

- Le code ne se rend pas compte que y est erroné, donc c'est qu'il correspond à un mot $f(w)$.
- On considère un mot z entre $f(x)$ et $f(w)$, qui a un bit de différent avec $f(x)$ et un bit de différent avec $f(w)$

$$\begin{array}{rl} f(x) & 01010100101100 \\ z & 01011100101100 \\ f(w) & 01011100100100 \end{array}$$

- Le code est capable de corriger z..
- ..mais il ne peut pas savoir s'il faut corriger z en $f(x)$ ou en $f(w)$.

Correction et détection

- Un code qui corrige une erreur peut en détecter deux.
- Preuve par contradiction : Soit $f(x)$ transmis avec deux erreurs, ce qui donne y.

$$\begin{array}{rl} f(x) & 01010100101100 \\ y & 01011100100100 \end{array}$$

- Le code ne se rend pas compte que y est erroné, donc c'est qu'il correspond à un mot $f(w)$.
- On considère un mot z entre $f(x)$ et $f(w)$, qui a un bit de différent avec $f(x)$ et un bit de différent avec $f(w)$

$$\begin{array}{rl} f(x) & 01010100101100 \\ z & 01011100101100 \\ f(w) & 01011100100100 \end{array}$$

- Le code est capable de corriger z..
- ..mais il ne peut pas savoir s'il faut corriger z en $f(x)$ ou en $f(w)$.

Outline

Détection

- Un code qui détecte une erreur est capable, si on lui donne tous les bits sauf un, de le retrouver
- Pourquoi ?
- Un code qui est capable, lorsqu'on lui donne tous les bits sauf un, de retrouver le dernier, détecte une erreur.
- Pourquoi ?

Bits de parité

- On considère le $(n, n+1)$ code qui ajoute au mot x la somme des bits de x modulo 2 (donc la parité du nombre de 1)

$$000101010 \mapsto 0001010101$$

- Permet de détecter une erreur. Pourquoi ?
- Marche aussi en base 10 (faire la somme modulo 10), et en base quelconque.

Utilisation pratique

- ISBN : Le 10ème chiffre est calculé de la façon suivante : On multiplie le $i$ème bit par i, et on somme tout modulo 11, on obtient x
- Le résultat est x (si x = 10, le résultat est alors X)
- Exemple : 2-070-36621-y

  $$2+2*0+3*7+4*0+5*3+6*6+7*6+8*2+9 = 141 = 9 \mod 11$$

  Donc $y = 9$.
- Exemple : 2-226-10936-y
- Exemple : 2-754-00257-y
- Exemple : 2-203-1y149-6. Trouver y.
- C'est un code qui détecte une erreur (Preuve ?)

CRC (Cyclic Redundancy Check)

- Principe similaire
- On représente le mot a_i par le polynôme $\sum a_i X^i$.
- On calcule le résultat modulo un polynôme P (et modulo 2).
- Exemple : Calcul du CRC de 111011110 pour le polynôme $x^5 + x^2 + 1$ (CRC-5-USB)

$$x^8 \quad +x^7 \quad +0 \quad +0 \quad +x^4 \quad +x^3 \quad +x^2 \quad +x \quad +0 \;\Big|\; \underline{x^5 + x^2 + 1}$$

- Résultat : $x^2 + x + 1$

CRC (Cyclic Redundancy Check)

- Principe similaire
- On représente le mot a_i par le polynôme $\sum a_i X^i$.
- On calcule le résultat modulo un polynôme P (et modulo 2).
- Exemple : Calcul du CRC de 111011110 pour le polynôme $x^5 + x^2 + 1$ (CRC-5-USB)

$$x^8 \quad +x^7 \quad +0 \quad +0 \quad +x^4 \quad +x^3 \quad +x^2 \quad +x \quad +0 \;\big|\; \underline{x^5 + x^2 + 1}$$

- Résultat : $x^2 + x + 1$

CRC (Cyclic Redundancy Check)

- Principe similaire
- On représente le mot a_i par le polynôme $\sum a_i X^i$.
- On calcule le résultat modulo un polynôme P (et modulo 2).
- Exemple : Calcul du CRC de 111011110 pour le polynôme $x^5 + x^2 + 1$ (CRC-5-USB)

$$\begin{array}{rrrrrrrrr|l} x^8 & +x^7 & +0 & +0 & +x^4 & +x^3 & +x^2 & +x & +0 & x^5+x^2+1 \\ \cline{10-10} x^8 & & & x^5 & & x^3 & & & & x^3 \\ \cline{1-9} & x^7 & 0 & x^5 & x^4 & 0 & x^2 & x & 0 & \end{array}$$

- Résultat : $x^2 + x + 1$

CRC (Cyclic Redundancy Check)

- Principe similaire
- On représente le mot a_i par le polynôme $\sum a_i X^i$.
- On calcule le résultat modulo un polynôme P (et modulo 2).
- Exemple : Calcul du CRC de 111011110 pour le polynôme $x^5 + x^2 + 1$ (CRC-5-USB)

$$
\begin{array}{cccccccccc|l}
x^8 & +x^7 & +0 & +0 & +x^4 & +x^3 & +x^2 & +x & +0 & x^5 + x^2 + 1 \\
\cline{10-10}
x^8 & & & x^5 & & x^3 & & & & x^3 + x^2 \\
\cline{1-9}
 & x^7 & 0 & x^5 & x^4 & 0 & x^2 & x & 0 & \\
 & x^7 & & & x^4 & & x^2 & & & \\
\cline{1-9}
 & & & x^5 & 0 & 0 & 0 & x & 0 &
\end{array}
$$

- Résultat : $x^2 + x + 1$

CRC (Cyclic Redundancy Check)

- Principe similaire
- On représente le mot a_i par le polynôme $\sum a_i X^i$.
- On calcule le résultat modulo un polynôme P (et modulo 2).
- Exemple : Calcul du CRC de 111011110 pour le polynôme $x^5 + x^2 + 1$ (CRC-5-USB)

$$
\begin{array}{rrrrrrrrr|l}
x^8 & +x^7 & +0 & +0 & +x^4 & +x^3 & +x^2 & +x & +0 & x^5+x^2+1 \\
\cline{10-10}
x^8 & & & x^5 & & x^3 & & & & x^3+x^2+1 \\
\cline{1-9}
 & x^7 & 0 & x^5 & x^4 & 0 & x^2 & x & 0 & \\
 & x^7 & & & x^4 & & x^2 & & & \\
\cline{1-8}
 & & & x^5 & 0 & 0 & 0 & x & 0 & \\
 & & & x^5 & & & x^2 & & 1 & \\
\cline{1-8}
 & & & & & & x^2 & x & 1 & \\
\cline{7-9}
\end{array}
$$

- Résultat : $x^2 + x + 1$

Outline

Parité

On peut utiliser le test de parité pour faire un code $(n^2, n^2 + 2n)$:

- On organise les n^2 bits en carré

1	1	1
0	0	1
1	0	1

- On calcule la parité de chaque ligne et de chaque colonne

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

- On peut alors détecter une erreur et la corriger

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

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

Distance de Hamming

- La distance de Hamming entre deux mots x et y est le nombre de bits sur lesquels ils diffèrent.

$$\begin{aligned} d(00010, 01010) &= \\ d(00011010, 00001011) &= \end{aligned}$$

- On note $B(x, k)$ la boule de centre x et de rayon k : c'est l'ensemble des mots qui diffèrent de x par moins de k bits.

Distance de Hamming

- La distance de Hamming entre deux mots x et y est le nombre de bits sur lesquels ils diffèrent.

$$\begin{aligned} d(00010, 01010) &= 1 \\ d(000110101, 000010111) &= 2 \end{aligned}$$

- On note $B(x, k)$ la boule de centre x et de rayon k : c'est l'ensemble des mots qui diffèrent de x par moins de k bits.

Distance de Hamming

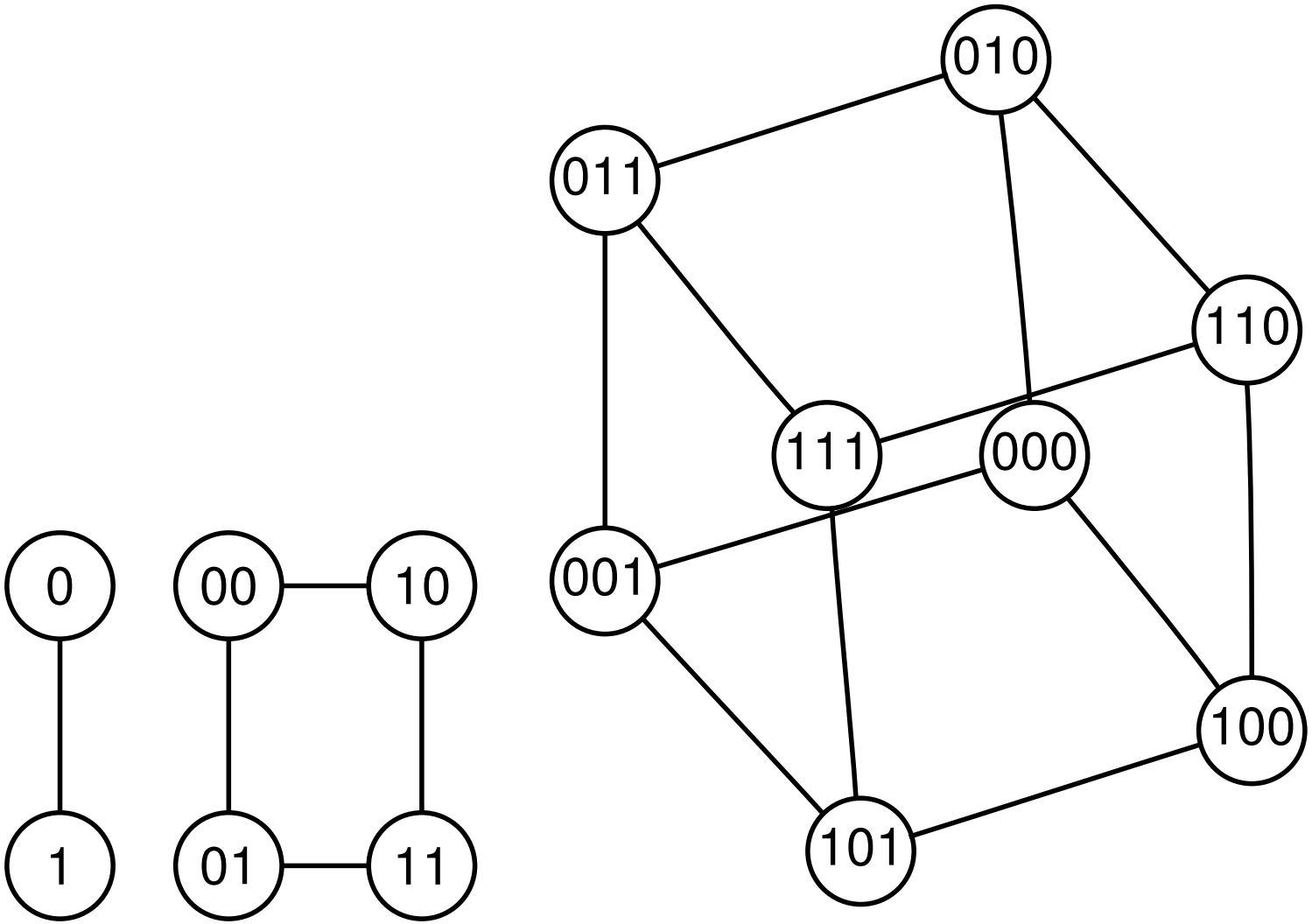
- La distance de Hamming entre deux mots x et y est le nombre de bits sur lesquels ils diffèrent.

$$\begin{aligned} d(00010, 01010) &= 1 \\ d(000110100, 000010111) &= 2 \end{aligned}$$

- On note $B(x, k)$ la boule de centre x et de rayon k : c'est l'ensemble des mots qui diffèrent de x par moins de k bits.

Distance de Hamming

Représentation graphique : On relie deux mots s'ils diffèrent par un bit : La distance de Hamming est alors la distance dans le graphe



Codes

- Un code f détecte k erreurs si la boule de centre $f(x)$ et de rayon k ne contient aucun autre mot de code que $f(x)$.
- Un code f corrige k erreurs si toute boule de rayon k ne contient qu'au plus un mot de code.
- Dit autrement : Un code f corrige k erreurs si deux boules de centre $f(x)$ et $f(y)$ et de rayon k ne s'intersectent pas.
- Exemple : Les codes de paramètre $(1,3)$
- Exemple : Les codes de paramètre $(2,3)$

Codes corrigeant une erreur

- Soit un code de paramètre (n, k).
- Un code f corrige une erreur si deux boules de centre $f(x)$ et $f(y)$ et de rayon 1 ne s'intersectent pas.
- Il y a ... boules
- Chaque boule contient ... points
- Il faut donc ...
- Pour $n = 1$, il faut donc $k \geq \ldots$
- Pour $n = 2$, il faut donc $k \geq \ldots$
- Pour $n = 3$, il faut donc $k \geq \ldots$
- Pour $n = 4$, il faut donc $k \geq \ldots$

Codes corrigeant une erreur

- Soit un code de paramètre (n, k).
- Un code f corrige une erreur si deux boules de centre $f(x)$ et $f(y)$ et de rayon 1 ne s'intersectent pas.
- Il y a 2^n boules
- Chaque boule contient ... points
- Il faut donc ...
- Pour $n = 1$, il faut donc $k \geq \ldots$
- Pour $n = 2$, il faut donc $k \geq \ldots$
- Pour $n = 3$, il faut donc $k \geq \ldots$
- Pour $n = 4$, il faut donc $k \geq \ldots$

Codes corrigeant une erreur

- Soit un code de paramètre (n, k).
- Un code f corrige une erreur si deux boules de centre $f(x)$ et $f(y)$ et de rayon 1 ne s'intersectent pas.
- Il y a 2^n boules
- Chaque boule contient $k + 1$ points
- Il faut donc ...
- Pour $n = 1$, il faut donc $k \geq \ldots$
- Pour $n = 2$, il faut donc $k \geq \ldots$
- Pour $n = 3$, il faut donc $k \geq \ldots$
- Pour $n = 4$, il faut donc $k \geq \ldots$

Codes corrigeant une erreur

- Soit un code de paramètre (n, k).
- Un code f corrige une erreur si deux boules de centre $f(x)$ et $f(y)$ et de rayon 1 ne s'intersectent pas.
- Il y a 2^n boules
- Chaque boule contient $k + 1$ points
- Il faut donc $2^n(k + 1) \leq 2^k$
- Pour $n = 1$, il faut donc $k \geq \ldots$
- Pour $n = 2$, il faut donc $k \geq \ldots$
- Pour $n = 3$, il faut donc $k \geq \ldots$
- Pour $n = 4$, il faut donc $k \geq \ldots$

Codes corrigeant une erreur

- Soit un code de paramètre (n, k).
- Un code f corrige une erreur si deux boules de centre $f(x)$ et $f(y)$ et de rayon 1 ne s'intersectent pas.
- Il y a 2^n boules
- Chaque boule contient $k + 1$ points
- Il faut donc $2^n(k + 1) \leq 2^k$
- Pour $n = 1$, il faut donc $k \geq 3$
- Pour $n = 2$, il faut donc $k \geq \ldots$
- Pour $n = 3$, il faut donc $k \geq \ldots$
- Pour $n = 4$, il faut donc $k \geq \ldots$

Codes corrigeant une erreur

- Soit un code de paramètre (n, k).
- Un code f corrige une erreur si deux boules de centre $f(x)$ et $f(y)$ et de rayon 1 ne s'intersectent pas.
- Il y a 2^n boules
- Chaque boule contient $k + 1$ points
- Il faut donc $2^n(k + 1) \leq 2^k$
- Pour $n = 1$, il faut donc $k \geq 3$
- Pour $n = 2$, il faut donc $k \geq 5$
- Pour $n = 3$, il faut donc $k \geq \ldots$
- Pour $n = 4$, il faut donc $k \geq \ldots$

Codes corrigeant une erreur

- Soit un code de paramètre (n, k).
- Un code f corrige une erreur si deux boules de centre $f(x)$ et $f(y)$ et de rayon 1 ne s'intersectent pas.
- Il y a 2^n boules
- Chaque boule contient $k + 1$ points
- Il faut donc $2^n(k + 1) \leq 2^k$
- Pour $n = 1$, il faut donc $k \geq 3$
- Pour $n = 2$, il faut donc $k \geq 5$
- Pour $n = 3$, il faut donc $k \geq 6$
- Pour $n = 4$, il faut donc $k \geq 7$

Leitmotiv

- Pour vérifier qu'un code corrige une erreur, il suffit de vérifier que deux boules de rayon 1 ne s'intersectent pas.
- Pour vérifier qu'un code corrige une erreur, il suffit de vérifier que deux mots de code diffèrent par au moins trois coordonnées.
- On appelle distance d d'un code la distance minimale entre deux mots de code. Si $d = 3$, il corrige une erreur. Si $d = 5$ il en corrige 2.

Arithmétique modulo 2

- On peut considérer un mot comme un vecteur dont les coordonnées sont 0 ou 1.
- On peut additionner deux vecteurs modulo 2, composante par composante
- Le résultat de $x \oplus y$ est le vecteur dans lequel tous les bits de x pour lesquels le bit correspondant de y est à 1 sont inversés.

$$\begin{pmatrix}1 & 0 & 0 & 1\end{pmatrix} \oplus \begin{pmatrix}0 & 1 & 0 & 1\end{pmatrix} = \begin{pmatrix}1 & 1 & 0 & 0\end{pmatrix}$$

- Corollaire : $d(x, y) = d(x \oplus z, y \oplus z)$
- Aussi : $d(x, y) = d(0, x \oplus y)$: Le nombre de bits qui diffèrent entre x et y est le nombre de bits à 1 dans $x \oplus y$.
- $x \oplus x$?

Code linéaire

- Un code linéaire est un code tel que la somme de deux mots de code est encore un mot de code.
- Exemple : $000, 111$ (tous les codes de répétition)
- Exemple : $101, 010, 000, 111$
- Exemple : $000, 011, 101, 110$ (bit de parité)
- Comme $x \oplus x = 0$, 0 est toujours un mot de code

Code linéaire

- Pour vérifier qu'un code linéaire corrige une erreur, il suffit de vérifier qu'aucun mot de code différent de $00\cdots0$ a strictement moins de 3 bits à 1.
- Preuve : On prend deux mots de code x et y. On suppose qu'ils diffèrent par moins de 3 coordonnées. Alors 0 et $x \oplus y$ diffèrent aussi par moins de 3 coordonnées. $d(x,y) = d(0, x \oplus y)$.
- Comme le code est un code linéaire, $x \oplus y$ est aussi un mot de code, donc on a trouvé un mot de code qui diffère par moins de 3 coordonnées de $00\cdots0$.
- Retour sur les exemples :
  - $000, 111$
  - $101, 010, 000, 111$
  - $0000, 0111, 1011, 1100$

Code linéaire

- On peut donner un code linéaire par une matrice H qui permet de vérifier si un mot est correct.
- Les mots de codes sont les x tels que $Hx = 0$.
- Premier exemple

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

- Deuxième exemple

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

Code de Hamming

- Le code de Hamming correspond à la matrice suivante :

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

- On peut vérifier qu'il corrige une erreur
- Il y a 2^4 mots de codes possibles
- On a un code $(4, 7)$ qui corrige une erreur.
- Se généralise : on construit la matrice H contenant tous les mots non nuls de longueur n, et on obtient un code $(2^n - n - 1, 2^n - 1)$

Combiner des codes (non nécessairement linéaires)

- On part d'un code A de paramètre (n, k) de distance d
- On part d'un code B de paramètre (m, k) de distance $d' \geq 2d$
- On construit le code $A \star B$ de paramètre $(m + n, 2k)$ dont les mots de code sont les mots de la forme $x \cdot x \oplus y$ pour x de A et y de B.
- Exemple : $A = 0000, 0110, 1100, 1010$
- Exemple : $B = 0000, 1111$
- On obtient ...
- Le code obtenu est de distance $2d$.

Reed-Solomon

- Pour changer, on va supposer qu'on travaille sur des octets (entre 0 et 255) plutôt que sur des bits.
- On code 2 caractères par 5 de la façon suivante :
- On voit les 2 entiers a et b comme deux points $(1, a)$ et $(2, b)$.
- On ajoute les points $(3, c)$, $(4, d)$, $(5, e)$ qui sont sur la même droite.
- Exemple : $20, 40$.
- Exemple : $20, 5$.
- Correction des erreurs ?
- Généralisation : au lieu de prendre une droite, on prend un polynôme de degré un peu plus grand.
- Exemple : $20, 40, 80$

Codes correcteurs d'erreur

- Plusieurs types de codes
- Plus on corrige d'erreurs, mieux c'est
- Plus on corrige d'erreur, plus on va avoir besoin de bits supplémentaires.
- Un compromis à trouver.
- En pratique : Les codes sont des généralisations des techniques précédentes
- Au lieu de travailler modulo 2, on travaille modulo un polynôme.