Codes correcteurs d'erreurs

Les mathématiques sont partout dans l'encodage de l'information, laquelle se fait en transformant les informations en suites de *bits*, soit des 0 et des 1. Mais, lors du transfert de l'information, certains bits peuvent être corrompus, un 0 étant transformé en 1, ou le contraire. Comment récupérer l'information ?

Christiane Rousseau

Université de Montréal

Les ordinateurs fonctionnent à l'aide de transistors que l'on peut modéliser simplement comme des interrupteurs qui ont deux positions : *ouvert* et *fermé*, que l'on peut représenter par 0 et 1, d'où l'idée de coder l'information en utilisant des bits.

Il est courant que des bits soient corrompus lors du transfert de l'information. Pourtant, il est souvent essentiel que le message puisse être décodé correctement. C'est le cas, par exemple, lorsqu'on communique avec une sonde interplanétaire. La solution retenue est d'utiliser un *code correcteur d'erreurs*. Le principe d'un tel code est d'ajouter de l'information redondante. Ainsi, si une partie de l'information est perdue, on peut la récupérer ailleurs.

Prenons un exemple. On veut envoyer quatre bits : 0010. On pourrait choisir de répéter chaque bit deux fois : 00 00 11 00. Ainsi, si on reçoit 00 10 11 00, on détecte qu'il y a eu une erreur. Par contre, on ne peut corriger… même s'il y a eu exactement une erreur. En effet, le mot envoyé aurait pu être 00 00 11 00, ou encore 00 11 11 00. Un tel code est un *code détecteur d'erreurs*. Si on choisit plutôt de répéter chaque bit trois fois, et donc d'encoder notre mot comme

000 000 111 000,

alors on peut corriger une erreur. Ainsi, si on reçoit 000 010 111 000, on déduit que le mot envoyé est 0010. Mais, si on reçoit 000 011 111 000, on corrige à 0110 et on a quand même reçu un mot erroné.

Un code correcteur d'erreurs ne permet de récupérer le mot initial que si un nombre pas trop grand d'erreurs se sont produites.

Sinon, l'information est perdue. Certains codes correcteurs d'erreurs peuvent corriger plus d'une erreur. Le choix du code correcteur utilisé dépend de la fiabilité du canal de transmission.

Mais, revenons à notre exemple. On est parti d'un mot de quatre bits et on l'a allongé à 12 bits pour s'assurer de toujours pouvoir corriger une erreur. Peut-on faire mieux ? Oui, à condition d'utiliser de la sophistication mathématique.

Le code de Hamming C(7, 4)

Ce code transforme les mots de 4 bits en mots de 7 bits et corrige une erreur. Pour cela, on va introduire une addition sur les bits donnée par la table suivante :

+	0	1
0	0	1
1	1	0

C'est l'addition modulo 2, que l'on peut se représenter ainsi : 0 représente un nombre pair et 1, un nombre impair. L'addition de deux nombres pairs ou de deux nombres impairs est paire, alors que celle d'un nombre pair et d'un nombre impair est impaire.

Prenons un mot de quatre bits :

$$u = u_1u_2u_3u_4.$$

Nous allons l'encoder en un mot de 7 bits où

$$v_1 = u_1$$
$$v_2 = u_2$$
$$v_3 = u_3$$
$$v_4 = u_4$$
$$v_5 = u_1 + u_2 + u_4$$
$$v_6 = u_1 + u_3 + u_4$$
$$v_7 = u_2 + u_3 + u_4$$

Envoyons le mot v. On reçoit le mot

$$w = w_1w_2w_3w_4w_5w_6w_7.$$

Calculons :

$$W_5 = w_1 + w_2 + w_4$$
$$W_6 = w_1 + w_3 + w_4$$
$$W_7 = w_2 + w_3 + w_4$$

Si le mot a été transmis sans erreur on devrait avoir $W_5 = w_5$, $W_6 = w_6$, $W_7 = w_7$.

Regardons maintenant ce qui se passe si une erreur s'est produite :

Bit erroné	Incompatibilités
w_1	$W_5 \neq w_5$, $W_6 \neq w_6$
w_2	$W_5 \neq w_5$, $W_7 \neq w_7$
w_3	$W_6 \neq w_6$, $W_7 \neq w_7$
w_4	$W_5 \neq w_5$, $W_6 \neq w_6$, $W_7 \neq w_7$
w_5	$W_5 \neq w_5$
w_6	$W_6 \neq w_6$
w_7	$W_7 \neq w_7$

On voit que les incompatibilités sont toutes différentes ! On peut donc corriger l'erreur selon les incompatibilités observées. Ainsi, si on veut envoyer u=0010, on l'encode en

$$v = 0010011.$$

Si on reçoit w = 0011011, on voit que $W_5 = 1 \neq w_5 = 0$, $W_6 = 0 \neq w_6 = 1$ et $W_7 = 0 \neq w_7 = 1$. On conclut qu'il y a eu erreur car il y a des incompatibilités, et que, si exactement une erreur s'est produite, alors elle s'est produite en w_4. On change w_4 pour récupérer v, et ensuite u.

On peut illustrer le code $C(7, 4)$ avec la figure ci-contre.

Corriger plus d'une erreur ?

Il arrive souvent qu'on doive corriger plus d'une erreur. On doit alors utiliser des codes correcteurs d'erreurs plus sophistiqués. Les codes de Reed-Solomon forment une famille de tels codes. Dans ces codes, les « lettres » sont des ensembles de m bits et les « mots » des ensembles de k lettres que l'on encode dans des mots de $2^m - 1$ lettres. Ces codes corrigent s erreurs ou moins, où

$$s = \left\lfloor \frac{1}{2}(2^m - k - 1) \right\rfloor,$$

et $\lfloor x \rfloor$ représente la partie entière de x. Comme une erreur représente ici une lettre, soit un ensemble de m bits, en corrigeant une erreur, un tel code pourrait corriger jusqu'à m bits de la lettre selon le nombre de bits erronés. Les codes de Reed-Solomon sont donc très performants pour corriger les erreurs groupées, ce qui arrive souvent quand il y a une interférence dans un canal de transmission.

Les codes de Hamming $C(2^k-1, 2^k-k-1)$

Pour $k = 3$ le code $C(7, 4)$ est le premier d'une grande famille de codes de Hamming qui corrigent une erreur. Si on prend $k = 7$, on obtient le code $C(127, 120)$ qui allonge des mots de 120 bits à 127 bits et corrige une erreur. Ce code très économique a été utilisé dans le Minitel en France de 1980 à 2012. Mais, il est inefficace s'il se produit souvent plus d'une erreur par suite de 127 bits.