Le codage numérique des nombres entiers (activité sans ordinateur).

Dominique Larrieu, Pr. de Mathématiques, Lycée Régional de Valbonne, exercices pour classes de 2^{nd}.

Vous allez voir que le codage des nombres est de nature différente, car ce n'est pas un code arbitraire « standard » comme le code ASCII pour les caractères, mais un choix issu de l'arithmétique.

En effet, il serait possible de représenter un nombre tel que 274 comme la succession des caractères '2', '7', '4', puis de coder chacun de ces caractères par son code ASCII pour obtenir ainsi : 00110001 00110111 00110100 .

Mais, si un tel codage représente bien la **chaîne de caractères '274'**, il ne représente pas vraiment **le nombre 274** et permet difficilement d'effectuer dessus des calculs arithmétiques tels que l'addition ou la multiplication.

Pour obtenir une représentation utilisable dans des calculs, il suffit cependant de passer de la numération en base 10 classique (10 chiffres) à **la numération en base 2 où on n'utilise que 2 chiffres : 0 et 1**.

Principe de ce changement de base.

Le tableau suivant donne les représentation en base 2 des dix premiers nombres entiers :

Base 10	Base 2	Principe de conversion	
0	0	$0\times 8 + 0 \times 4 + 0 \times 2 + \mathbf{0} \times 1$	$0\times 2^3 + 0 \times 2^2 + 0 \times 2^1 + \mathbf{0} \times 2^0$
1	1	$0\times 8 + 0 \times 4 + 0 \times 2 + \mathbf{1} \times 1$	$0\times 2^3 + 0 \times 2^2 + 0 \times 2^1 + \mathbf{1} \times 2^0$
2	10	$0\times 8 + 0 \times 4 + \mathbf{1} \times 2 + \mathbf{0} \times 1$	$0\times 2^3 + 0 \times 2^2 + \mathbf{1} \times 2^1 + \mathbf{0} \times 2^0$
3	11	$0\times 8 + 0 \times 4 + \mathbf{1} \times 2 + \mathbf{1} \times 1$	$0\times 2^3 + 0 \times 2^2 + \mathbf{1} \times 2^1 + \mathbf{1} \times 2^0$
4	100	$0\times 8 + \mathbf{1} \times 4 + \mathbf{0} \times 2 + \mathbf{0} \times 1$	$0\times 2^3 + \mathbf{1} \times 2^2 + \mathbf{0} \times 2^1 + \mathbf{0} \times 2^0$
5	101	$0\times 8 + \mathbf{1} \times 4 + \mathbf{0} \times 2 + \mathbf{0} \times 1$	$0\times 2^3 + \mathbf{1} \times 2^2 + \mathbf{0} \times 2^1 + \mathbf{1} \times 2^0$
6	110	$0\times 8 + \mathbf{1} \times 4 + \mathbf{1} \times 2 + \mathbf{0} \times 1$	$0\times 2^3 + \mathbf{1} \times 2^2 + \mathbf{1} \times 2^1 + \mathbf{0} \times 2^0$
7	111	$0\times 8 + \mathbf{1} \times 4 + \mathbf{1} \times 2 + \mathbf{1} \times 1$	$0\times 2^3 + \mathbf{1} \times 2^2 + \mathbf{1} \times 2^1 + \mathbf{1} \times 2^0$
8	10000	$\mathbf{1}\times 8 + \mathbf{0} \times 4 + \mathbf{0} \times 2 + \mathbf{0} \times 1$	$\mathbf{1}\times 2^3 + \mathbf{0} \times 2^2 + \mathbf{0} \times 2^1 + \mathbf{0} \times 2^0$
9	10001	$\mathbf{1}\times 8 + \mathbf{0} \times 4 + \mathbf{0} \times 2 + \mathbf{1} \times 1$	$\mathbf{1}\times 2^3 + \mathbf{0} \times 2^2 + \mathbf{0} \times 2^1 + \mathbf{1} \times 2^0$
10	…….	…..	….
11	…….	…..	….
12	…….	…..	….

On a donc choisi comme codage binaire d'un nombre, son écriture en base 2.

Du nombre entier positif à son codage binaire

Il existe un algorithme pour obtenir le code binaire de n'importe quel nombre entier.

Exemple avec le nombre 9 :
Si nous notons 9 % 2 = 1 le reste de la division de 9 par 2 ou 4 % 2 = 0 le reste de la division de 4 par 2, etc., alors le calcul suivant, où nous calculons successivement le reste de la division par deux puis divisons par deux nous donne... le codage binaire du nombre 9 :

$$\begin{array}{l} 9 \,\%\, 2 = \mathbf{1} \\ (9 \;/\; 2 = 4) \,\%\, 2 = \mathbf{0} \\ \qquad (4 \;/\; 2 = 2) \,\%\, 2 = \mathbf{0} \\ \qquad\qquad (2 \;/\; 2 = 1) \,\%\, 2 = \mathbf{1} \end{array}$$

on peut poser $\begin{array}{c|c} 9 & 1 \\ 4 & 0 \\ 2 & 0 \\ 1 & 1 \\ 0 & \end{array}$ et lire de bas en haut la colonne de droite. 9 en base 2 s'écrit : …….

Ainsi 274 en base 2 s'écrit : …………………….

Du codage binaire au nombre entier positif

Voici le calcul qui permet de retrouver un nombre à partir de son code binaire :

$$\left.\begin{array}{l} \mathbf{1} \times 8 + \mathbf{0} \times 4 + \mathbf{0} \times 2 + \mathbf{1} \times 1 = \\ \mathbf{1} \times 2^3 + \mathbf{0} \times 2^2 + \mathbf{0} \times 2^1 + \mathbf{1} \times 2^0 = \\ ((\mathbf{1} \times 2 + \mathbf{0}) \times 2 + \mathbf{0}) \times 2 + \mathbf{1} \quad = \end{array}\right| 9$$

- À la première ligne, nous avons multiplié respectivement par 1, 2, 4, 8 de droite à gauche les bits qui représentent 9.
- À la seconde ligne, nous nous sommes aperçus que 1, 2, 4, 8 sont en fait les puissances de 2 : 2^0, 2^1, 2^2...
- À la troisième ligne, nous factorisons 2 autant que possible ; cette écriture avec ses parenthèses bien placées nous montre qu'il suffit de multiplier par deux puis d'additionner les bits de gauche à droite pour trouver le nombre en base 10.

Voilà donc un calcul, une méthode mécanique que l'on appelle algorithme, qui permet de passer du codage binaire au nombre dans sa représentation usuelle en base 10.

Représenter des entiers positifs de toutes tailles

Entiers	0 à 1	0 à 3	0 à 7	0 à 15	0 à 255	0 à 1023	0 à environ 4 milliards	0 à plus de 1.8×10^{20}
Codés sur	1 bit	2 bits	3 bits	4 bits	8 bits	10 bits	32 bits	64 bits

Nos ordinateurs utilisent aujourd'hui souvent 32 ou 64 bits pour coder les nombres entiers, ils peuvent donc coder directement des nombres très grands et, si besoin était, en utilisant plus de bits... des nombres vertigineusement grands !