Informatique

Représentation des nombres
Partie 1

Cours

Représentation des nombres – partie 1

Prenons un exemple simple. Calculons avec python :

$$A = 0{,}2 + 0{,}1$$

Le résultat est très simple, n'est-ce pas ?

$$A = 0{,}2 + 0{,}1 = 0{,}3$$

Voyons de résultat sous Python :

```
>>> 0.2+0.1
0.30000000000000004
```

OUPS

Ou encore :

```
>>> 0.3==0.1+0.2
False
```

1.I. Code binaire

1.I.1 Introduction

Un ordinateur manipule des informations binaires, c'est-à-dire à deux états : 0 ou 1

Il est donc nécessaire de traduire un nombre du système en base 10 en un nombre binaire afin de permettre à un système informatique de le manipuler.

Par exemple, le 10 de notre système en base 10 est représenté par le « nombre » 1010 en binaire.

On écrira :

$$10_{(10)} = 1010_{(2)}$$

Le principe est très simple. En base 10, on ajoute un nouveau chiffre à chaque fois que l'on dépasse la valeur 9 puis la valeur 99 puis 999 etc : on groupe les nombres « par paquets de dix », puis « par paquets de dix paquets de dix », etc.

En binaire, on compte « par paquet de 2 », on passe donc au chiffre suivant que l'on dépasse la valeur 1.

En décimal, il y a 10 chiffres : 0,1,2,3,4,5,6,7,8,9

En binaire, il y a 2 chiffres : 0 et 1

Nous parlerons aussi du système hexadécimal, dans le quel on compte par paquets de 16 et dans le quel il y a donc 16 « chiffres » : 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F.

Complétons le tableau donnant les 12 premiers entiers en base 2 :

Base 10	Base 2	Base 10	Base 2
0		6	
1		7	
2		8	
3		9	
4		10	
5		11	

Dans un système donné, les chiffres utilisés s'appellent des « digits ».

Définition à retenir :

Les chiffres d'un nombre écrit en binaire s'appellent des « **bits** » (constraction de binary digit).

Un « **mot »** est une suite de « bits ». *Un **octet** est un mot de 8 bits.*

Un nombre entier naturel codé sur n bits permettra de représenter 2^n valeurs différentes et pourra donc au maximum correspondre à la valeur en base 10 de $2^n - 1$, le 0 étant inclus.

***Exercice 1** : Quels entiers naturels peut-on représenter sur un octet ?*

1.I.2 Principe de l'écriture d'un entier dans les bases 2 et 10

Prenons un exemple dans la base 10 :

$$352_{(10)} = 3 * 10^2 + 5 * 10^1 + 2 * 10^0$$

Chaque chiffres (digit) 3, 5 et 2 correspond à une puissance de 10.

En binaire, on respecte le même principe mais avec des puissances de 2 :

$$1010_{(2)} = 1 * 2^3 + 0 * 2^2 + 1 * 2^1 + 0 * 2^0$$

1.I.3 Transcodage Entier ↔ Binaire

Soit un nombre entier positif x.

Son écriture en base 10 est : $(d_n d_{n-1} \dots d_1 d_0)_{10}$ où d_n, …, d_0 sont les digits (les chiffres).

Son écriture en binaire est : $(b_m b_{m-1} \dots b_1 b_0)_2$ où $b_m, \dots, b_0$ sont les bits de ce nombre

On a donc :

$$d_n 10^n + d_{n-1}10^{n-1} + \cdots + d_1 10^1 + d_0 10^0 = b_m 2^m + b_{m-1}2^{m-1} + \cdots + b_1 2^1 + b_0 2^0$$

Le transcodage d'un entier :

- Binaire/Base 10 consiste à trouver les digits d_i connaissant les digits b_i
- Base 10/binaire consiste à trouver les digits b_i connaissant les digits d_i

1.I.3.a Binaire → Entier

Rien de plus simple, connaissant le nombre en binaire, il suffit de procéder à la somme de chacun de ses digits pris de droite à gauche en multipliant par des puissances de 2 croissantes.

On pourra pour cela s'aider du tableau suivant (à compléter) :

$2^0 =$	$2^4 =$	$2^8 =$
$2^1 =$	$2^5 =$	$2^9 =$
$2^2 =$	$2^6 =$	$2^{10} =$
$2^3 =$	$2^7 =$	$2^{11} =$

Exemple : $1111101000_{(2)}$

$$1111101000_{(2)} = 1.2^9 + 1.2^8 + 1.2^7 + 1.2^6 + 1.2^5 + 0.2^4 + 1.2^3 + 0.2^2 + 0.2^1 + 0.2^0$$

$$1111101000_{(2)} = 2^9 + 2^8 + 2^7 + 2^6 + 2^5 + 2^3 = 512 + 256 + 128 + 64 + 32 + 8$$

$$1111101000_{(2)} = 1.10^3 + 0.10^2 + 0.10^1 + 0.10^0$$

$$1111101000_{(2)} = 1000_{(10)}$$

***Exercice 2** : Ecrire en base 10 les nombres écrits en binaire suivants :*

$$a = (1001001)_2 \quad ; \quad b = (11111111)_2$$

1.I.3.b Entier → Binaire

Supposons qu'un nombre entier N s'écrive :

$$N = b_m 2^m + b_{m-1} 2^{m-1} + \dots + b_2 2^2 + b_1 2^1 + b_0 2^0$$

Par définition, les digits b_i ne sont pas divisibles par 2.

On peut écrire :

$$N = 2(b_m 2^{m-1} + b_{m-1} 2^{m-2} + \dots + b_2 2^1 + b_1) + b_0$$

Ce qui est la division euclidienne de N par 2. Donc **b_0 est le reste de la division euclidienne de N par 2.**

Et si on pose $q_1 = b_m 2^{m-1} + b_{m-1} 2^{m-2} + \dots + b_2 2^1 + b_1$, q_1 est le quotient de cette division euclidienne.

On peut alors continuer et écrire :

$$q_1 = 2(b_m 2^{m-2} + b_{m-1} 2^{m-3} + \dots + b_2) + b_1$$

Donc **b_1 est le reste de la division euclidienne de q_1 par 2.**

Et ainsi de suite :

Les bits de l'écriture binaire d'un entier naturel sont les restes des divisions euclidiennes successives par 2.

Rappel : pour simplifier, la division euclidienne d'un entier par un autre est celle que l'on a apprise à poser en primaire !

***Exercice 3** : Ecrire en base 2 le nombre* $a = (1000)_{10}$

Exercice 4 *: Ecrire en base 2 le nombre* $a = (2024)_{10}$

1.I.3.c En Python

Binaire → Base 10	Base 10 → Binaire
`>>> 0b1111101000` `1000`	`>>> bin(1000)` `'0b1111101000'`
`>>> 0b1111101000` `1000`	`>>> bin(1000)` `'0b1111101000'`

1.I.4 Transcodage Réel base 10 ↔ Réel base 2

En base 10, lorsque nous manipulons des nombres à virgule :

- Les chiffres avant la virgule sont des puissances de 10
- Les chiffres après la virgule sont des puissances de 1/10

Exemple :

$$5{,}375 = 5.10^0 + 3.10^{-1} + 7.10^{-2} + 5.10^{-3}$$

On peut exprimer ce nombre réel comme la somme de sa partie entière et de sa partie décimale :

$$5{,}375 = 5 + 0{,}375$$

En binaire, on va appliquer le même principe. Ainsi, le nombre $(101{,}011)_2$ représente le nombre :

$$1.2^2 + 0.2^1 + 1.2^0 + 0.2^{-1} + 1.2^{-2} + 1.2^{-3} = 4 + 0 + 1 + 0 + \frac{1}{4} + \frac{1}{8} = 5{,}375$$

Là aussi, on peut exprimer ce nombre comme somme de sa partie entière et sa partie décimale :

$$(101{,}011)_2 = (101)_2 + (0{,}011)_2$$

On pourra remarquer qu'il y a concordance entre partie entière et partie décimale entre les deux écritures :

$$(101)_2 = (5)_{10} \quad ; \quad (0{,}011)_2 = (0{,}375)_{10}$$

1.I.4.a Réel base 2 → Réel Base 10

Pour transcoder un nombre binaire réel en nombre réel en base 10, c'est très simple maintenant que l'on a compris le principe de l'écriture d'un nombre binaire réel.

Ecrivons un nombre binaire réel sous la forme : $x = (b_n \dots b_0, c_1 \dots c_m)_2$

Avec b_i et c_i les bits du mot binaire associé au nombre représenté.

On a alors : $(x)_{10} = b_n.2^n + \cdots + b_0.2^0 + c_1.2^{-1} + \cdots + c_m.2^{-m}$

Exercice 5 *: Ecrire en base 10 le nombre* $a = (10101{,}0101)_2$

1.I.4.b Réel base 10 → Réel base 2

Soit un nombre réel en base 10 de la forme : $x = (e_n \dots e_0, f_1 \dots f_m)_{10}$

On sait qu'il doit s'écrire sous la forme : $b_{n'} . 2^{n'} + \dots + b_0 . 2^0 + d_1 . 2^{-1} + \dots + d_{m'} . 2^{-m'}$

Séparons partie entière et partie décimale : $Ent = e_n \dots e_0 \quad ; \quad Dec = f_1 \dots f_m$

1.I.4.b.i Partie entière binaire

Pour trouver la partie entière du nombre binaire associé, il suffit de transcrire le Ent en binaire.

1.I.4.b.ii Partie décimale binaire

Concernant sa partie décimale, prenons un exemple pour comprendre : $x = (0,375)_{10}$

- On calcule $10 * 0,375 = 3,75$: la partie entière est le chiffre des dizièmes.
- On calcule $3,75 - 3 = 0,75$ puis $0,745 * 10 = 7,5$: la partie entière est le chiffre des centièmes.
- Etc.

Pour les obtenir en binaire, on va procéder de la même manière mais en multipliant par 2 :

$$0,375 = c_1 * 2^{-1} + c_2 * 2^{-2} + c_3 * 2^{-3} + \cdots$$

On multiplie par 2 :

$$0,750 = c_1 + \underbrace{c_2 * 2^{-1} + c_3 * 2^{-2} + \cdots}_{\in [0,1[}$$

Donc c_1 est la partie entière de $0,750$: $c_1 = 0$

On retranche c_1 à $0,750$:

$$0,750 - 0 = 0,750 = \ c_2 * 2^{-1} + c_3 * 2^{-2} + \cdots$$

On multiplie par 2 :

$$1,5 = c_2 + \underbrace{c_3 * 2^{-1} + \cdots}_{\in [0,1[}$$

Donc c_2 est la partie entière de $1,5$: $c_2 = 1$.

Et on continue ainsi de suite.

Principe	Présentation améliorée
$0,375 * 2 = 0,750 \rightarrow \begin{cases} Bit\ 0 \\ Nouvelle\ partie\ décimale\ 0,75 \end{cases}$ $0,750 * 2 = 1,5 \rightarrow \begin{cases} Bit\ 1 \\ Nouvelle\ partie\ décimale\ 0,5 \end{cases}$ $0,50 * 2 = 1 \rightarrow \begin{cases} Bit\ 1 \\ Nouvelle\ partie\ décimale\ 0 \end{cases}$	<table><tr><td>0,375</td><td>* 2 =</td><td>0,</td><td>750</td></tr><tr><td>0,750</td><td>* 2 =</td><td>1,</td><td>5</td></tr><tr><td>0,50</td><td>* 2 =</td><td>1,</td><td>0</td></tr></table>

C'est terminé : $(0,375)_{10} = (0,011)_2$

Remarque : Tout comme on pourrait dire que $0,375 = 375.10^{-3}$, on verra dans la suite qu'en passant par une écriture scientifique binaire, ce transcodage est aussi possible : $(0,011)_2 = (11)_2 . 2^{-3} = \frac{3}{8} = 0,375$

Exercice 6 *: Ecrire en base 2 le nombre* $a = (45{,}625)_{10}$

Exercice 7 *: Ecrire en base 2 le nombre* $b = (0{,}1)_{10}$

On s'aperçoit qu'en base 2 le nombre $0{,}1$ a une infinité de bits : il ne peut pas s'écrire de façon précise.

De même que beaucoup de nombre réels ne sont pas décimaux (n'ont pas une écriture décimale finie), de nombreux réels n'ont pas d'écriture binaire fini.

Remarque : les réels qui ont une écriture binaire finie sont appelés les nombres dyadiques.