Représentation des nombres réels

BRUNO DARID

28 novembre 2019

PLAN

1 Représentation approximative des nombres réels

1.1 Conversion d'un nombre décimal en binaire - virgule fixe

La méthode consiste à décomposer la partie entière et la partie fractionnaire suivant les puissances de deux (puissances positives pour la partie entière et puissances négatives pour la partie fractionnaire). Ces deux parties étant séparées par la virgule (qui est fixe dans ce cas). La conversion binaire → décimale est alors évidente :

2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}
1	0	1	1	1	0	1

soit 5,8125.

L'algorithme décimal → binaire consiste à réaliser les étapes suivantes :

1) Convertir la partie entière (*voir cours précédents*) ;
2) Convertir la partie fractionnaire en adoptant l'algorithme suivant :
   — multiplier la partie fractionnaire par deux ;
   — extraire la partie entière qui donne un des bits de la partie fractionnaire ;
   — répéter tant que la partie fractionnaire restante est différente de zéro.

1.2 Conversion d'un nombre décimal en binaire - virgule flottante

1.2.1 Le format

La deuxième façon d'encoder un nombre décimal est inspirée de la notation scientifique :

$$\pm m \times 10^n$$

mais en base deux, c'est-à-dire

$$(-1)^s \times 1, f \times 2^{e-biais}$$

Il s'agit des **nombres à virgule flottante**.
Ce format est constitué de trois parties essentielles :

— 1 bit de signe s ;

— un exposant e ; pour éviter d'avoir que des grandes valeurs, on décale cet exposant d'une certaine valeur (*biais*) ;

— une partie fractionnaire f appelé encore mantisse.

signe (s)	exposant (e)	partie fractionnaire ou mantisse (f)

La représentation des nombres à virgule flottante est entièrement définie dans la **norme IEEE 754**. Celle-ci prévoit une représentation **simple précision sur 32 bits** ou **double précision sur 64 bits**.

	exposant (e)	fraction (f)	valeur
32 bits	8 bits	23 bits	$(-1)^s \times 1, f \times 2^{e-127}$
64 bits	11 bits	52 bits	$(-1)^s \times 1, f \times 2^{e-1023}$

1.2.2 Exemple de conversion décimal $\rightarrow$ flottant

Le choix a été fait ici de travailler avec un nombre N positif. Dans le cas d'un nombre N négatif, il suffit juste de changer le bit de signe $s = 1$ et de faire les calculs avec la valeur absolue $|N|$.

Soit à convertir $N = 17,25$ en flottant, format simple précision sur 32 bits. Il s'agit de touver les trois composantes s, e et f de la représentation en virgule flottante.
Le nombre est positif, donc $s = 0$.
Pour trouver e, écrivons N en binaire suivant la méthode de la virgule fixe :

$$17,25 = 10001,01$$

On va maintenant faire apparaître une puissance de 2 (comme en base 10) en décalant la virgule :

$$10001,01 = 1,000101 \times 2^4 \tag{1}$$

Comme on travaille en simple précision, on a $e - 127 = 4$, soit $e = 4 + 127 = 131$, soit encore $e = 10000011$ en binaire sur 8 bits.
La relation (1) donne aussi la partie fractionnaire (*mantisse*) f. On a :

$$f = 000101$$

que l'on complète à 23 bits avec des zéros à droite. Finalement, le nombre $N = 17,25$ s'écrit en format simple précision :

0	10000011	00010100 00000000 0000000

1.2.3 Exemple de conversion flottant $\rightarrow$ décimal

Soit à convertir :

1 10000110 10101101100000000000000

Signe : $s = (-1)^1 = -1$;
Exposant : $e - 127$, soit $10000110_2 = 134 - 127 = 7$;
Mantisse : $1, f$ avec $f = 2^{-1} + 2^{-3} + 2^{-5} + 2^{-6} + 2^{-8} + 2^{-9}$, soit $1,677734375$.
Finalement, ce nombre vaut en décimal :

$$-1,677734375 \times 2^7 = -214,75$$

2 Cas particuliers

Lorsque **tous les bits sont à zéros**, cela correspond à la **valeur zéro**.
Lorsque **tous les bits de l'exposant sont à 1 et que la partie fractionnaire est nulle**, cela correspond à **l'infini** (*plus ou moins, cela dépend du bit de signe*).
Enfin, pour les nombres très petits (inférieurs à 2^{-126}) il existe une forme **dénormalisée**, qui ne sera pas étudiée ici.

3 Caractère approchée de la représentation

Travail à réaliser : donner les représentations binaire de 0.2 et 0.3
Un ordinateur qui ne peut stocker qu'un nombre fini de chiffre, ne peut représenter correctement ces nombres et utilise par conséquent une valeur approchée. Par exemple :

- 1.2* 3 donne 3.5999999999999996 ;
- 0.1 + 0.2 == 0.3 donne False

Par ailleurs, des propriétés comme l'associativité de l'addition ne sont plus valables :

```
In [14]: 1.6 +(3.2 + 1.7)

Out[14]: 6.5

In [15]: (1.6 + 3.2) + 1.7

Out[15]: 6.500000000000001
```

4 À retenir

Les nombres flottants sont une représentation approximative des nombres réels dans les ordinateurs. Une norme internationale IEEE754 définit un encodage en simple ou double précision (32 ou 64 bits).

Les opérations arithmétiques sur les nombres flottants n'ont pas toujours les mêmes propriétés que ces mêmes opérations sur les nombres réels.