

Cours Méthodes Numériques M367

Optionnel L3 A-B GECTS

Grenoble Second Semestre 2009-2010.

Chapitre I: Représentation des nombres.

1.1 Développent des nrs en base β .

Prop. Soit $\beta \geq 2$ un entier. Soit $m \in \mathbb{N}$. Il existe une suite finie d'entiers $(\delta_k)_{k \in \mathbb{N}}$ telle que

$$(1) \quad 0 \leq \delta_k < \beta$$

$$(2) \quad \exists N \quad \forall k \geq N \rightarrow \delta_k = 0.$$

$$(3) \quad m = \sum_{k=0}^N \delta_k \beta^k$$

Déf (3) s'appelle la représentation de m en base β .

On écrit aussi $m = \delta_N \delta_{N-1} \dots \delta_0$

Exemple:	base 10	base 2	base 8	base 16	$(A=10$ $B=11, \dots)$
	2610	<u>11111</u> 0 11 0 1 0	3732	9BA	
		<u>56u</u>			

Preuve de la proposition

• Pas 1: Lemma $\sum_{k=0}^{N-1} \beta^k \cdot (\beta-1) = (\beta^N - 1)$. En effet $\sum_{k=0}^m q^k = \frac{q^{m+1} - 1}{q - 1}$ si $q \neq 1$

Donc, pour tout $(\delta_k)_{k \in \mathbb{N}}$ t.q. $0 \leq \delta_k \leq \beta-1 \quad 0 \leq \sum_{k=0}^{N-1} \delta_k \beta^k \leq \beta^N - 1$.

Ceci permet de définir $S_N: \left[\{0, \dots, \beta-1\}^{N-1} \xrightarrow{\sim} \{0, \dots, \beta^N-1\} \right] \text{ pour } N \in \mathbb{N}^*$
 $(\delta_k)_{k=0}^{N-1} \mapsto \sum_{k=0}^{N-1} \delta_k \beta^k$.

La proposition résulte alors du fait que S_N soit bijective pour tout $N \in \mathbb{N}^*$.

• Pas 2: On établit cette propriété par récurrence sur $N \in \mathbb{N}^*$. Le cas $N=1$ est trivial.

Supposons la propriété établie pour $N-1 \in \mathbb{N}^*$. Soit $m \in \mathbb{N}$ t.q. $0 \leq m \leq \beta^{N-1}$

Effectuons la div. euclidienne par β^{N-1} . On a:

1

$$x = \delta_N \beta^{N-1} + r_N \quad \text{avec } 0 \leq r_N < \beta^{N-1}.$$

Comme $x < \beta^N$ et $0 \leq \delta_N < \beta$. Notez que $\delta_N = E\left(\frac{x}{\beta^{N-1}}\right)$

$$\text{Comme } g_{N-1} \text{ est surjective} \quad r_N = \sum_{k=0}^{N-2} \delta_k \beta^k$$

de là $n = \sum_{k=0}^{N-1} \delta_k \beta^k$. Ainsi $n \in \text{Im}(g_N)$ et g_N est surjective.
les deux ensembles source et but de g^N ont tous deux β^N éléments donc
 g_N est bijective. La propriété est établie pour $N \in \mathbb{N}^*$. \square .

Prop: Soit $x \in \mathbb{R}^+$. Il existe une unique suite $(\delta_k)_{k \in \mathbb{Z}}$ d'entiers telle que

$$1 \quad 0 \leq \delta_k < \beta - 1$$

$$2 \quad \exists N \quad k \geq N \Rightarrow \delta_k = 0$$

$$3 \quad x = \sum_{k=-\infty}^{k=N} \delta_k \beta^k$$

$$4 \quad \forall M \quad \exists k \leq -M \quad \delta_k \neq \beta - 1$$

Exemple: $\beta = 10$ la représentation 3 n'est autre que la représentation décimale usuelle des nombres réels, par exemple

$$\pi = 3,14159 \dots$$

$$\begin{matrix} \uparrow & \uparrow \\ \delta_0(\pi) & \delta_1(\pi) \end{matrix} \quad \text{et} \quad g = \delta_2(\pi)$$

~~Notons que~~ $\sum_{k=1}^{\infty} 10^{-k} \cdot g = \frac{10^{-1} \cdot 1}{1 - 1/10} \cdot g = \frac{1}{10} \cdot \frac{9}{9/10} = 1$

Ainsi l'écriture 0,9999... ne vérifie pas la condition 4 qui a pour effet d'éliminer la redondance $1 = 0,999\dots$

Preuve: La preuve est constructive. Supposons que $x \neq 0$ et $\beta^{N+1} \leq x < \beta^{N+2}$
i.e.: $N = E\left(\frac{\log(x)}{\log(\beta)}\right)$. On fixe $\delta_{N+1} = E\left(\frac{x}{\beta^{N+1}}\right) \in \mathbb{N}$

et on a $0 \leq \delta_{N+1} < \beta$. Ceci fournit

$$x = \delta_N \beta^N + x_{N+1} \quad \text{avec } \beta^{N+1} \leq x_{N+1} < \beta^{N+2}$$

$$\text{En étant } x = \delta_N \beta^N + \delta_{N-1} \beta^{N-1} + \dots + \delta_{N-p} \beta^{N-p} + x_p \quad 0 \leq x_p < \beta^{N-p}$$

donc $x_p \rightarrow 0$ et $x = \sum_{k=-\infty}^{k=N} \delta_k \beta^k$. Ceci établit l'existence d'une suite vérifiant 1, 2, 3. La propriété 4 s'établit en remarquant

que $\sum_{k=-\infty}^{k=-1} (\beta - 1) \beta^k = \beta^0 - 1$ et on laisse en exercice ainsi que l'unicité.

1.2 Arithmétique des nombres flottants

Dans les ordinateurs (calculatrices) les nombres sont représentés comme des nombres flottants (= à virgule flottante).

Définition Soit $\beta \in \mathbb{N}$ $\beta \geq 2$ Soient $m, N \in \mathbb{N}^*$
Soit $r \in \mathbb{N}^*$

l'ensemble $F = \{\beta, r, -m, M\}$ est l'ensemble des nombres réels de la forme $\pm \beta^r \cdot \sum_{k=-n}^{n-1} \delta_k \beta^k$ avec $0 \leq \delta_k < \beta-1$
 $1 \leq \delta_0 \leq \beta-1$
 $-m \leq r \leq +M$.

3 composantes \rightarrow le signe \pm
 \rightarrow l'exposant : r compris entre $-m$ et $+M$.

\rightarrow la mantisse $m = \sum_{k=-1}^{n-1} \delta_k \beta^k$

Ex: $\beta=10$ $r=4$ $0,3146 \cdot 10^4$ est la représentation flottante standard de π

r = nombre de chiffres significatifs

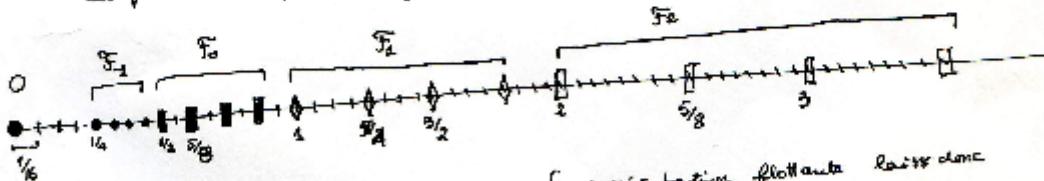
Remarque Si on autorisait $\delta_0=0$ on aurait une redondance car π a une précision de $r+1$ chiffres significatifs.

$$\sum_{k=-2}^{n-1} \delta_k \beta^k = \beta^1 \sum_{k=-1}^{n-2} \delta_k \beta^k \cdot \beta^{k+1}$$

Example: $F(2, 3, -1, 2) \cap \mathbb{R}^+$

positifs
Les flottants d'exposant 0 sont $0,100$, $0,101$, $0,110$, $0,111$ et font l'ensemble F_0
 $0,100$, $0,101$, $0,110$, $0,111$, $1/2$, $5/8$, $3/4$, $7/8$

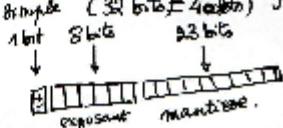
positifs
Les flottants d'exposant 1 forment l'ensemble $F_1 = 2^1 \cdot 5_0$



de représentation flottante laisse donc
les mal de "têtes"!

Arithmétique flottante en

Example: Précision simple (32 bits) = 4 octets $F(2, -127, 128, 24)$.



Les 8 bits peuvent coder $2^8 = 256$ entiers compris entre -127 et 128

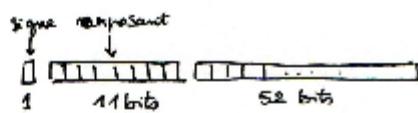
Même si le code de la mantisse a 23 bits, il y a 24 chiffres significatifs binaires puisque le premier est toujours 1.

Notez que: $2^{-127} \approx 1,701418 \cdot 10^{-38}$ est le plus petit flottant de précision simple.

Exemple Arithmétique flottante en double précision = 64 bits.

$$F(2, \dots, -1023, +1024; 53).$$

$$2^{11} = 2048$$



Dans le processeur du calculateur est implementée une arithmétique machine d'où à des opérations $\oplus, \ominus, \otimes, \oslash$ qui implémentent aussi précisément que possible les quatre opérations de l'arithmétique élémentaire. Faisant de la machine l'implémentation de l'arithmétique machine dépend de la machine en question et ~~et de sa longueur~~ comme suit :

Soit $A: \mathbb{R} \rightarrow F$ une application "arrondi" avec les propriétés suivantes

$$\forall z \in F \quad A(z) = z \quad A(-z) = -A(z)$$

$$\forall z \in \mathbb{R}, f, f' \in F \quad f = \sup_{k \geq 0} \{ k \in F \mid k \leq z \}$$

$$f' = \inf_{k < 0} \{ k \in F \mid k \geq z \}$$

si $f \neq +\infty$ on a $z \in [f, f']$ et $A(z) =$ le plus proche pair de f avec $A(\frac{f+f'}{2})$ dépend de la machine

si $f = +\infty$ $A(z) =$ le plus grand flottant

si $f = -\infty$ $A(z) =$ dépend de la machine

On pose alors :

$$f \oplus f' = A(f + f')$$

$$f \ominus f' = A(f - f')$$

$$f \otimes f' = A(f \cdot f')$$

$$f \oslash f' = A(f / f')$$

1.3 Remarques sur l'arithmétique flottante.

$$x, y, z, r = 3$$

$$\begin{aligned} x &= 1.12 \\ y &= 0.00300 \\ z &= 0.00400 \end{aligned}$$

$$x + y + z = 1.127$$

$$x \otimes y = 1.12$$

$$(x \otimes y) \otimes z = 1.12$$

$$y \otimes x = 0.00300$$

$$x \otimes (y \otimes z) = 1.12$$

→ \oplus n'est pas associative

→ erreur minimale en sommant les plus petits termes d'accord.

Dépassement de capacité supérieure / inférieure: Si le résultat d'une opération est $>$ au plus grand flottant en valeur absolue, l'arithmétique flottante perd toute pertinence. Ceci est une erreur par dépassement de capacité supérieure (overflow). De même s'il est non nul et $<$ au plus petit flottant (underflow).

Instabilité numérique: cumul des erreurs d'arrondi dans les processus itératifs. Peut amener à des catastrophes numériques.

L'étude détaillée de ces sources d'erreurs (qui peuvent se combiner) sont des objectifs de ce cours. Elles sont analysées dans les méthodes standard et leur solution.