Codage numérique : du nombre au pixel - Cours 4

Codage des réels

L1 – Université de Lorraine
B. Girau et N. de Rugy Altherre

Transparents disponibles sur la plateforme de cours en ligne

Codage des réels
Virgule fixe

Virgule fixe

Principes du codage en virgule fixe

- codage entier + position virgule
- si f bits en partie fractionnaire, alors $n - f$ bits en partie entière
- codage de x = codage entier de $x.2^f$ (éventuellement négatif)

Exemple

nombre 3,4375 codé en virgule fixe avec 3 chiffres binaires en partie entière et 5 en partie fractionnaire (on parle de codage Q3.5) : 01101110_2 (110_{10}), car $110/32 = 3,4375$.

On peut noter directement avec la virgule, mais celle-ci n'est que virtuelle, elle n'apparaît pas dans le code : 011.01110

Virgule fixe

Opérateurs

- opérateurs arithmétiques : utilisation des opérateurs entiers, puis "placement" de la virgule
- multiplication :
  - taille de la partie entière/fractionnaire = somme des tailles des parties entières/fractionnaires
  - troncature en fonction de la position de la virgule

Virgule fixe

Exemple

nombre de taille 2.6 $\times$ nombre de taille 3.5 = nombre de taille 5.11

								1	1.	0	1	1	1	0	0	
						x		0	0	0.	1	0	1	0	0	
0	0	0	1	0.	0	0	1	0	0	1	1	0	0	0	0	

i.e. : 3,4375 (220/64) x 0,625 (20/32) = 2,1484375 (4400/2048)

Virgule fixe

Exemple 2

nombre de taille 2.6 $\times$ nombre de taille 2.6 = nombre de taille 4.12
que se passe-t-il si on suppose maintenir un codage en taille 2.6

							1	1.	0	1	1	1	0	0
						x	0	0.	0	1	0	1	0	0
0	0	0	1.	0	0	0	1	0	0	1	1	0	0	0	0

i.e. : 3,4375 (220/64) x 0,3125 (20/64) = 0,07421875
(4400/4096) $\simeq$ 1,0625

N.B. : on positionne la virgule (en comptant à partir des poids faibles) puis on tronque la partie fractionnaire.

Codage des réels
Virgule flottante

Virgule flottante

Norme IEEE 754

- norme adoptée en 1985
- définit la simple et la double précision
- principe général :
  - signe
  - exposant de taille k (simple : 8, double : 11)
  - mantisse de taille m (simple : 23, double : 52)

$$x = (-1)^s \, x_0.x_{-1}x_{-2}\ldots x_{-m+1} \, 2^e$$

est codé par $s \, e_{k-1}e_{k-2}\ldots \; e_1e_0 \, x_0x_{-1}\ldots \; x_{-m+2}x_{-m+1}$

Virgule flottante, norme IEEE 754

Mantisse normalisée

- on impose (sauf cas particulier) $x_0 = 1$
- il devient inutile de l'expliciter dans la mantisse

donc

$$x = (-1)^s \, 1.x_{-1}x_{-2}\ldots x_{-m} \, 2^e$$

est en fait codé par $s \, e_{k-1}e_{k-2}\ldots \; e_1e_0 \, x_{-1}x_{-2}\ldots \; x_{-m+1}x_{-m}$

Virgule flottante, norme IEEE 754

Exposant biaisé

- pour ne pas avoir de signe, on code e+biais
- biais= $2^{k-1} - 1$
- valeur min réservée pour le codage de 0
- valeur max réservée pour le codage des infinis et de `NaN`
- simple précision, biais= 127, exposant biaisé va de 1 pour $e = -126$ à 254 pour $e = 127$
- double précision, biais= 1023, exposant biaisé va de 1 pour $e = -1022$ à 2046 pour $e = 1023$

Virgule flottante, norme IEEE 754

Cas général

En résumé, en norme IEEE 754, si $e = e_{k-1} \dots e_0$ est différent de 0 et de $2^k - 1$, alors $s\, e_{k-1} e_{k-2} \dots e_1 e_0\, x_{-1} x_{-2} \dots x_{-m+1} x_{-m}$ code le réel

$$x = (-1)^s\, 1.x_{-1}x_{-2}\dots x_{-m}\, 2^{e-(2^{k-1}-1)}$$

Exemple

En simple précision, 1 10000001 01100000000000000000000, code le réel $-1,375.2^{129-127} = -5,5$

Virgule flottante, norme IEEE 754

Codage de 0

- exposant biaisé min $= 000\ldots000$
- mantisse nulle $= 000\ldots000$
- signe 0 ou 1, pour 0^+ et 0^-
- attention : $\sqrt{0^-} = 0^-$

Codage des infinis

- exposant biaisé max $= 111\ldots111$
- mantisse nulle $= 000\ldots000$
- signe 0 ou 1, pour $+\infty$ et $-\infty$

Virgule flottante, norme IEEE 754

NaN

- “not a number”
- symbolise un résultat invalide ou indéterminé
- exposant biaisé max $= 111\ldots111$
- mantisse non nulle
- signe 0 ou 1

Exemples (génération de NaN)

- $+\infty \ - \ +\infty$
- $0 \times \infty$
- $0/0$
- $+\infty \ / \ +\infty$

Virgule flottante, norme IEEE 754

NaN (suite)

- propagation des `NaN`
- résultats logiques pas toujours intuitifs

Exemples

- $x + \texttt{NaN} = \texttt{NaN}$
- $\texttt{NaN} - \texttt{NaN} = \texttt{NaN}$
- si $x = \texttt{NaN}$ alors $x == x$ est faux et $x! = x$ est vrai
- si $x = \texttt{NaN}$ ou $y = \texttt{NaN}$ alors $x < y$, $x <= y$, $x == y$, $x >= y$ et $x > y$ sont faux

Virgule flottante, norme IEEE 754

Nombres dénormalisés

- exposant minimal
- mantisse non normalisée (pas de '1' implicite)
- but : échantillonner mieux autour de 0

Exemples

$0\,00000000\,00100000000000000000000 = 2^{-129}$

$1\,00000000\,10000000000000000000000 = -2^{-127}$

$1\,00000000\,00000000000000000000001 = -2^{-149}$

Virgule flottante, norme IEEE 754

Arrondis

Notion d'arrondi :

- ensemble des réels échantillonné par valeurs représentables en machine
- arrondi déterministe
- 4 modes possibles :
  - au plus proche (arrondi "pair" si on est au milieu)
  - par excès (vers $+\infty$)
  - par défaut (vers $-\infty$)
  - vers zéro

Arrondis (suite)

Notion d'arrondi "exact" ou "correct" :

- principe : le système doit se comporter comme si le calcul était fait en précision infinie sur les opérandes exactes, puis arrondi
- comportement prévisible et reproductible
- indispensable pour portabilité des logiciels, reproductibilité des calculs, interopérabilité des systèmes
- difficile à satisfaire pour les opérations élémentaires

Virgule flottante, norme IEEE 754

Arrondis, cas des fonctions élémentaires

- si on a m bits de mantisse, on calcule d'abord une approximation de précision n bits, avec $m < n$, puis on arrondit
- question fondamentale : quel n faut-il pour que l'approximation obtenue soit la même que l'arrondi du résultat exact ?
- pour les opérateurs arithmétiques, $n = m + 3$ suffit toujours
- théorème/algo de 1975 : pour calculer *log* et *exp* en double précision, il faut $n = 10^{244}$! ! !
- théorème/algo de 1995 : on se ramène à environ $n = 1\,000\,000$ bits
- optimal ? ? ?

Virgule flottante, norme IEEE 754

Opérations arithmétiques : cas de l'addition $(s_x, e_x, m_x) + (s_y, e_y, m_y)$

- traiter les cas particuliers : $x_1 = NaN$ ou $x_2 = NaN$, $x_1 = 0$ ou $x_2 = 0$, $x_1 = +/-\infty$ ou $x_2 = +/-\infty$

addition	$-\infty$	$x \in \mathbb{F}_-^*$	0^-	0^+	$x \in \mathbb{F}_+^*$	$+\infty$	NaN
$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$	NaN	NaN
$y \in \mathbb{F}_-^*$	$-\infty$	$x+y$ ou $-\infty$	y	y	$x+y$	$+\infty$	NaN
0^-	$-\infty$	x	0^-	0^+	x	$+\infty$	NaN
0^+	$-\infty$	x	0^+	0^+	x	$+\infty$	NaN
$y \in \mathbb{F}_+^*$	$-\infty$	$x+y$	y	y	$x+y$ ou $+\infty$	$+\infty$	NaN
$+\infty$	NaN	$+\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$	NaN
NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Virgule flottante, norme IEEE 754

Opérations arithmétiques : cas de l'addition $(s_1, e_1, m_1) + (s_2, e_2, m_2)$

- différence des exposants : $d = e_x - e_y$
- échange des valeurs : $x \longleftrightarrow y$ et $d = -d$ si $d < 0$
- alignement des mantisses : $m_y = m_y >> d$
- calcul du type d'opération :
  $op = +$ si $(add, s_x = s_y)$ ou $(sub, s_x \neq s_y)$
  $op = -$ si $(add, s_x \neq s_y)$ ou $(sub, s_x = s_y)$
- addition/soustraction : $m_r = m_x \ op \ m_y$
- correction du signe : $m_r = -m_r$ si $m_r < 0$

Virgule flottante, norme IEEE 754

Opérations arithmétiques : cas de l'addition $(s_1, e_1, m_1) + (s_2, e_2, m_2)$

- détection du 1 de poids fort (t =indice MSO, most significant one)
- normalisation (et correction exposant) : $m_r = m_r << t$ et $e_r = e_r + t$
- dénormalisation
- arrondi (en fonction des 3 bits de garde)
- renormalisation si l'arrondi provoque une propagation de retenue

Opérations arithmétiques : cas de l'addition