Codage des nombres entiers positifs

Codage des nombres entiers positifs

M. Combacau
combacau@laas.fr

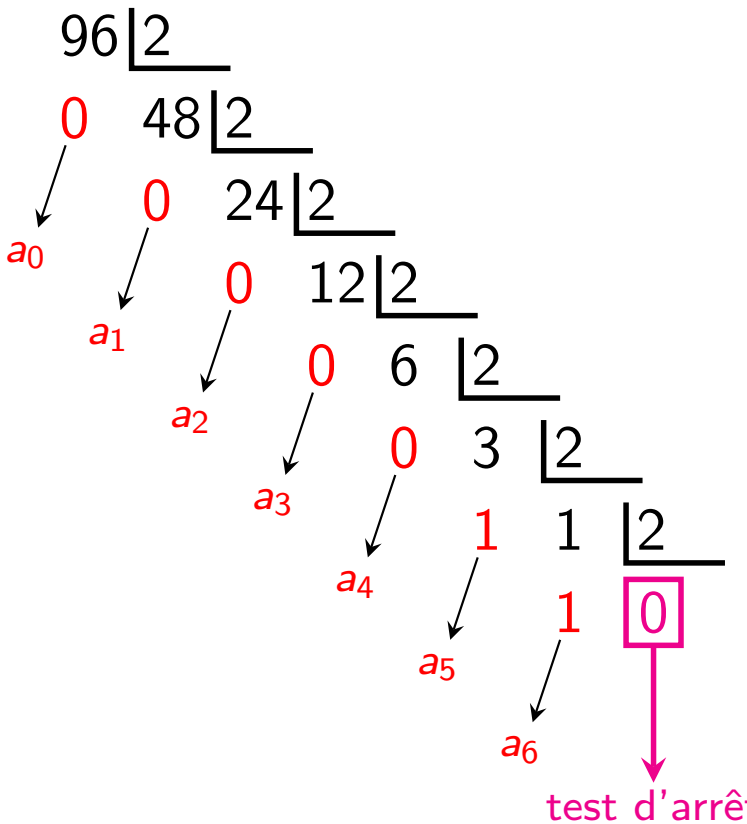
Université Paul Sabatier
LAAS-CNRS

November 12, 2024

Objectif

Exercices sur les techniques de codage des nombres entiers et fractionnaires

Codage de 96_{10} sur 8 bits



D'où $\left| \begin{array}{ll} 96_{(10)} & = [01100000] \\ & = 2^6 + 2^5 \\ & = 64 + 32 \end{array} \right.$

$\rightarrow$ autre démarche : décomposer le nombre à coder en puissances de 2 (rapide à la main)

Rq : le principe des divisions successives peut être utilisé quelles que soient la base de départ b_d et la base d'arrivée b_a pourvu que $b_d > b_a$, le calcul s'effectuant dans la base de départ.

Décodage de [11000101]

Il s'agit ici de passer de la base 2 à la base 10. Il n'est pas possible d'utiliser l'algorithme vu juste avant car le calcul en base 2 ne permet pas de traiter les nombres écrits en base 10.

	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
$A =$	1	1	0	0	0	1	0	1

Il suffit d'utiliser la formule donnant la valeur d'un nombre en binaire naturel

$$A = \sum_{i=0}^{n-1} a_i.2^i$$

D'où valeur (A)$= 2^7 + 2^6 + 2^2 + 2^0 = 128 + 64 + 4 + 1 = 197_{(10)}$

Codage de -18 suivant les trois formats (sur 8 bits)

On notera "code binaire naturel" : cbn

signe+valeur absolue

1.Signe :
$-18 < 0 \Rightarrow b_7 = 1$

2. Codage de $|-18|$
Notons que 18=16+2
codé en cbn [0001 0010]
d'où le code de -18 :[1001 0010]

avec biais (128)

On doit coder la valeur -18+128 en cbn

remarque :
128-18=127-17

127 :[0111 1111] et
17 : [0001 0001]

d'où le code de -18 :
[0110 1110]

Ca2

$-18 < 0$ on va donc coder la valeur 256-18

or 256-18=255-17

255 :[1111 1111]
17 : [0001 0001]
d'où le code de -18 :
[1110 1110]

Décodage de [11000101] suivant les trois formats

signe+valeur absolue

1.Signe :
$b_7 = 1 \Rightarrow A < 0$

2.
$$\begin{aligned} |A| &= 2^6 + 2^2 + 2^0 \\ &= 64 + 4 + 1 \\ &= 69 \end{aligned}$$

d'où A= -69

avec biais (128)

Il faut soustraire le biais du codage

Cela peut être fait en binaire (sur 8 bits)

$$\begin{array}{rr} & 11000101 \\ - & 10000000 \\ \hline = & 01000101 \end{array}$$

d'où A=69

où en décimal
[11000101 = 128+64+4+1 = 197
et finalement
A= 197-128 = 69

Ca2

1.Signe :
$b_7 = 1 \Rightarrow A < 0$

2. pour calculer $|A|$, il faut "prendre" le complément à 2 du code

1100010|1
0011101|1
d'où
$$\begin{aligned} |A| &= 32+16+8+2+1 \\ &= 59 \end{aligned}$$
et finalement : A =-59

Codage de 14.25 en virgule fixe (6,2)

$[b_7$	b_6	b_5	b_4	b_3	b_2	b_1	$b_0]$	le code
$[2^5$	2^4	2^3	2^2	2^1	2^0.	2^{-1}	$2^{-2}]$	son interprétation en virgule fixe (m,f)=(6,2)

Pour le codage, deux démarches

1. calculs de la partie entière (divisions successives par 2) et de la partie fractionnaire (multiplications successives par 2)
   $14 = 8+4+2 \rightarrow 001110$
   $0.25 = 2^{-2} \rightarrow 01$
   d'où le code : [00111001]
2. on code la partie entière de $A \times 2^f$
   ici f=2, on code donc $14,25 \times 2^2 = 57 = 32 + 16 + 8 + 1$
   d'où le code [00111001]

Remarque : si la valeur $A \times 2^f$ n'est pas entière, on code l'entier le plus proche de $A \times 2^f$ et une erreur de codage existe.

Décodage de [11010101] en virgule fixe (6,2)

Ici $b_7 = 1$ donc la valeur codée est négative ($A < 0$)

on recherche la valeur de $|A|$ en "prenant" le complément à 2 du code.

1101010|1
0010101|1

Comme lors du codage, deux démarches existent

1. Calcul partie entière et partie fractionnaire
   Le code de la partie entière = 001010 et $\lfloor A \rfloor = 8 + 2 = 10$
   Le code de la partie fractionnaire = 11 qui code la valeur $2^{-1} + 2^{-2} = 0.75$
   d'où $|A| = 10.75$ et finalement $A = -10.75$
2. Décodage de [00101011] en 32+8+2+1=43=$|A| \times 2^f$
   puis multiplication par 2^{-f} pour obtenir $|A|$
   $|A| = 43/2^2 = 10.75$ et finalement $A = -10.75$

Codage de 14.25 en simple précision (IEEE p754)

Rappel : simple précision = 32 bits dont

- b_{31} : signe
- $b_{30} \dots b_{23}$: exposant sur 8 bits biaisé de 127 $(2^7 - 1)$
- $b_{22} \dots b_0$: partie fractionnaire appelée mantisse (1,mantisse)

Application : 14,25

- $14,25 > 0 \rightarrow b_{31} = 0$
- 14,25 : $1110.01 = 1.11001(\times 2^3)$! formellement incorrect ! (base 2 et base 10)
  D'où la matisse m=11001 et $b_{22} \dots b_0 = 1100100 \dots 0$
- exposant = 3+127 = 128+2 codé en BN par $b_{30} \dots b_{23} = 10000010$

et finalement le code [0 10000010 11001000000000000000000]

Décodage de [11000010010000000000000000000000] en simple précision (IEEE p754)

- $b_{31} = 1$ donc $a < 0$
- $b_{30} \dots b_{23} = 10000100$ codant la valeur 128+4=127+5 donc exposant $= 5$
- $b_{22} \dots b_0 = 1000 \dots 0$ codant la valeur 0.5 d'où la mantisse m=0.5

Et finalement le nombre codé $A = -1.5 \times 2^5 = -48$

Précision en virgule fixe

- Codons la valeur : 14.25
  On a vu que 14.25 était codable sans erreur. $E_a = 0$ et $E_r = 0$ (parfait !)
- Codons la valeur : 14,26
  $14.26 \times 2^2 = 47,04$ n'est pas une valeur entière, erreur de codage !
  La valeur entière la plus proche est 47 qui correspond à 14,25
  L'erreur absolue commise vaut $14,26 - 14,25 = 0.01$
  L'erreur relative commise vaut $0.01/14,25 \approx 0.07\%$ (acceptable)
- Codons la valeur 0.12
  $0.12 \times 2^2 = 0.48 \rightarrow$ valeur non entière
  On va coder l'entier le plus proche : 0
  l'erreur absolue commise vaut $E_a = |0 - 0.12| = 0.12$
  et l'erreur relative commise vaut $E_r = 0.12/0.12 = 100\%$ (inacceptable)

L'erreur relative commise sur les valeurs "proches de 0" est importante.
Seules les valeurs multiples de 2^{-f} sont codables sans erreur.