Représentation des nombres entiers

Définition Écriture d'un nombre dans une base

Dans un système de numération en base B, un nombre noté N_B peut s'écrire sous la forme : $N_B = \sum_{k=0}^{n} a_k \cdot B^k$ s'écrit symboliquement sous la forme : $N_B = \underbrace{(a_n a_{n-1} \cdots a_2 a_1 a_0)_B}_{n+1 \text{ chiffres}}$ On note :

- B : la base ou nombre de chiffres différents qu'utilise le système de numération ;
- a_k : chiffre de rang k ;
- B^k la pondération associée à a_k.

Représentation des nombres entiers relatifs

Méthode Pour représenter l'opposé d'un nombre positif par son complément à deux, on inverse les bits 0 et 1 et on ajoute 1 au mot binaire obtenu.

Représentation des nombres réels

Méthode Conversion d'une partie fractionnaire en binaire

1. On multiplie la partie fractionnaire par 2.
2. La partie entière obtenue représente le poids binaire (limité aux seules valeurs 0 ou 1).
3. La partie fractionnaire restante est à nouveau multipliée par 2.
4. On procède ainsi de suite jusqu'à ce qu'il n'y ait plus de partie fractionnaire ou que le nombre de bits obtenus correspond à la taille du mot mémoire dans lequel on stocke cette partie.

Pour représenter des réels, nombres pouvant être positifs, nuls, négatifs et non entiers, on utilise la représentation en virgule flottante (*float* en anglais) qui fait correspondre au nombre 3 informations :

$$-243,25_{(10)} = \underbrace{-}_{1} \; \underbrace{0,24325}_{2} \cdot 10^{\overbrace{3}^{3}}$$

On appelle alors :

1. le signe (positif ou négatif) ;
2. la mantisse (nombre de chiffres significatifs) ;
3. l'exposant : puissance à laquelle la base est élevée.

Sous cette forme normalisée, il suffit de mémoriser le signe, l'exposant et la mantisse pour avoir une représentation du nombre en base 10. Il n'est pas utile de mémoriser le 0 avant la virgule puisque tous les nombres vont commencer par 0. En faisant varier l'exposant, on fait « flotter » la virgule décimale.

C'est cette méthode que l'on va adapter pour coder les réels en binaire naturel. Il faut au préalable les écrire sous la forme (norme IEEE 754 – Institute of Electrical and Electronics Engineers) :

$$\text{signe 1, mantisse x } 2^{\text{exposant}}$$

Le mot binaire obtenu sera la juxtaposition de 3 parties :

Le tableau décrit la répartition des bits selon le type de précision : la taille de la mantisse (m bits) donne la précision mais suivant la valeur de l'exposant, la précision sera totalement différente.

	Signe	Exposant	Mantisse
Simple précision – 32 bits	1	8	23
Double précision – 64 bits	1	11	52
Précision étendue – 80 bits	1	15	64

Ainsi :

- erreur relative : 2^{-m} (poids du dernier bit)
- erreur absolue : erreur relative * 2^{exposant}

Simple précision : $2^{-23} = 1,192... * 10^{-7}$ Double précision : $2^{-52} = 2,220... * 10^{-16}$

Procédure de conversion de réel en binaire (hexadécimale)

Méthode

1. Convertir en binaire les partie entière et fractionnaire du nombre sans tenir compte du signe.
2. Décaler la virgule vers la gauche pour le mettre sous la forme normalisée (IEEE 754).
3. Codage du nombre réel avec les conventions suivantes :
   - signe = 1 : Nombre négatif (Signe = 0 : Nombre positif) ;
   - le chiffre 1 avant la virgule étant invariant pour la forme normalisée, il n'est pas codé ;
   - on utilise un exposant décalé au lieu de l'exposant simple (complément sur octet). Ainsi, on ajoute à l'exposant simple la valeur 127 en simple précision et 1023 en double précision (c'est à dire $2^{n-1}-1$ où n est le nombre de bits de l'exposant) ;
   - la mantisse est complétée à droite avec des zéros.

Représentation chaînes de caractères

Les fichiers texte stockés sur un périphérique de stockage sont encodés avec un encodage particulier (ANSI, UTF-8, ASCII...). L'encodage est une table de correspondance entre une séquence de bits et un caractère (visible – une lettre – ou non – une tabulation–).

Par exemple, l'ASCII est un encodage fréquemment utilisé où les caractères sont codés sur 7 bits et permet donc d'encoder 127 caractères. Cependant, les informations étant très souvent regroupées par octets, un caractère ASCII occupe donc 8 bits.

Code décimal	Caractère ASCII	Description	Décimal	Caractère	Décimal	Caractère	Décimal	Caractère
0	NUL	Null	32	Space	64	@	96	‘
1	SOH	Start of heading	33	!	65	A	97	a
2	STX	Start of text	34	”	66	B	98	b
3	ETX	End of text	35	#	67	C	99	c
4	EOT	End of transmission	36	$	68	D	100	d
5	ENQ	Enquiry	37	%	69	E	101	e
6	ACQ	Acknowledge	38	&	70	F	102	f
7	BEL	Bell	39	’	71	G	103	g
8	BS	Backsapce	40	(	72	H	104	h
9	TAB	horizontal tab	41	)	73	I	105	i
10	LF	New line feed, new line	42	*	74	J	106	j
11	VT	Vertical tab	43	+	75	K	107	k
12	FF	NP form feed, new page	44	‘	76	L	108	l
13	CR	Carriage return	45	-	77	M	109	m
14	SO	Shift out	46	.	78	N	110	n
15	SI	Shift in	47	/	79	O	111	o
16	DLE	Data link espace	48	0	80	P	112	p
17	DC1	Device control 1	49	1	81	Q	113	q
18	DC2	Device control 2	50	2	82	R	114	r
19	DC3	Device control 3	51	3	83	S	115	s
20	DC4	Device control 4	52	4	84	T	116	t
21	NAK	Negative acknowledge	53	5	85	U	117	u
22	SYN	Synchronous idle	54	6	86	V	118	v
23	ETB	End of trans. block	55	7	87	W	119	w
24	CAN	Cancel	56	8	88	X	120	x
25	EM	End of medium	57	9	89	Y	121	y
26	SUB	Substitute	58	:	90	Z	122	z
27	ESC	Escape	59	;	91	[	123	{
28	FS	File separator	60	<	92	\	124	|
29	GS	Group separator	61	=	93	]	125	}
30	RS	Record separator	62	>	94	^	126	~
31	US	Unit separator	63	?	95	_	127	DEL