Représentation binaire d'un entier relatif

Bruno Darid

20 novembre 2019

Plan

1 Encodage des entiers relatifs

1.1 Valeur maximale et dépassement de capacité

On suppose dans cette leçon que l'on travaille avec une capacité fixe de n bits.
Rappel : avec n bits on peut représenter 2^n valeurs comprises dans l'intervalle $[0; 2^n - 1]$. Que se passe-t-il si on ajoute une unité à la valeur maximale ? Pour $n = 4$ bits par exemple :

$$\begin{array}{cccc} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 0 \end{array}$$

La valeur obtenue 2^n n'est pas représentable sur n bits (on ignore la retenue) : on obtient que des zéros ! Ce phénomène traduit un **dépassement de capacité**.

1.2 Propriétés à vérifier et première solution

L'encodage choisi doit respecter deux propriétés essentielles :

✓ $a + (-a) = 0$ (propriété 1)

✓ $-(-a) = a$ (propriété 2)

La première idée est d'utiliser un bit (généralement **le bit de poids fort**) pour représenter le signe et les autres bits pour représenter la valeur absolue du nombre. Par convention, le signe + est codé par 0 et le signe - par 1.
Avec n bits, on aurait ainsi 1 bit pour le signe et $n - 1$ bits pour coder les 2^{n-1} valeurs différentes.

1.3 Exemple

Soit à coder -5 sur 4 bits. On a $|-5| = 5 = 101_2$, le bit de signe valant 1 car $-5 < 0$. On aurait donc le codage suivant : $-5 \rightarrow 1\ 1\ 0\ 1$. La première propriété ci-dessus est-elle vérifiée ? Ajoutons les représentations de 5 et -5 :

$$\begin{array}{cccc} 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ \hline 0 & 0 & 1 & 0 \end{array}$$

La valeur obtenue est différente de zéro : cette solution n'est pas satifaisante.
Remarque : un problème supplémentaire peut être mis en évidence : l'existence de **deux** représentations pour zéro.

1.4 Une solution satisfaisante : le complément à deux

Cette solution exploite le dépassement de capacité. Sur n bits, le plus grand nombre représentable est $2^n - 1$. Ce qui implique que $2^n - 1 + 1 = 2^n$ sera représenté par 0 0 ... 0 (dépassement de capacité).

$$(2^n - 1) + 1 \rightarrow 0$$

La quantité $2^n - 1$ peut être considérée comme **une représentation** de -1.
Généralisons ce résultat : pour une capacité de n bits et pour un entier x, $2^n - x$ est une représentation valable de $-x$.

Vérifions que les propriétés citées précédemment sont vérifiées.

$$x + (2^n - x) = 2^n$$

Or, 2^n est représenté par 0 0 ... 0, la propriété (1) est vérifiée. En ce qui concerne la propriété (2) :

$$\begin{aligned} -(-x) &= -(2^n - x) \\ &= 2^n - (2^n - x) \\ &= x \end{aligned}$$

Conclusion :
pour une capacité de n bits et pour un entier naturel x, la quantité $2^n - x$ appelée **complément à deux** de x est une bonne représentation de $-x$.

2 Comment utiliser le complément à deux ?

Il s'agit de présenter une méthode permettant d'obtenir une représentation de $2^n - x$ soit l'opposé de x.

$$\begin{aligned} 2^n - x &= 2^n - 1 + 1 - x \\ &= (2^n - 1) - x + 1 \end{aligned}$$

Or, réaliser $(2^n - 1) - x$ (*appelé aussi complément à un*) consiste à **inverser tous les bits** de x (sauriez-vous le montrer ?).

D'où la méthode pour représenter l'opposé d'un entier relatif :

- Écrire la valeur absolue du nombre en binaire naturel avec le nombre de bits spécifié (compléter si besoin avec des zéros) ;
- Inverser tous les bits (complément à un) ;
- Ajouter 1 au résultat précédent

3 Domaine couvert avec une capacité de n bits

3.1 Entiers positifs

On a des valeurs allant de 0 0 ... 0 à 0 1 ... 1 soit de 0 à $2^{n-1} - 1$.

3.2 Entiers négatifs

Si on ajoute 1 au nombre positif maximal, on obtient : 1 0 ... 0, soit un nombre négatif ! Ce phénomène est appelé overflow et est à l'origine de bug célèbre comme celui ayant entrainé la destruction d'Ariane5 pour son vol inaugural.
Les entiers négatifs sont donc étalés de 1 1 ... 1 soit -1 (montrez le !) à 1 0 ... 0 soit -2^{n-1}.

3.3 Conclusion

Avec n bits, en arithmétique signée, on peut coder les entiers appartenant à l'intervalle :

$$[-2^{n-1}, ..., -1, 0, ..., 2^{n-1} - 1]$$

4 À retenir

Pour coder un entier relatif, on utilise la méthode du complément à deux. Pour cela, si la capcité est fixée à n bits :

— Écrire la valeur absolue du nombre en binaire naturel avec le nombre de bits spécifié (compléter si besoin avec des zéros) ;

— Inverser tous les bits (complément à un) ;

— Ajouter 1 au résultat précédent.

Avec n bits, on peut coder les entiers appartenant à l'intervalle $[-2^{n-1}, ..., -1, 0, ..., 2^{n-1} - 1]$. Si une opération conduit à un nombre se situant en dehors de cet intervalle, on a un phénomène d'overflow se traduisant par un résultat aberrant.