Codage de l'information
Base de représentation des nombres
Cours 1

Rakotoarimalala Tsinjo

1 Base de représentation de nombre

Un nombre peut s'écrire de différentes façons selon la base choisie.

Une représentation d'un nombre dans une base $b > 1$, est une suite de chiffres/symboles dans $\{0, 1, \dots b-1\}$.

Par exemple notre base conventionnelle est la base 10. Donc tous nombres dans la base 10 est une séquence de chiffres entre $\{0, 1, \dots 9\}$.

On va écrire n_b pour préciser que le nombre n est écrit dans la base b.

La valeur d'un nombre $n_b = a_k a_{k-1} \dots a_1 a_0$ représenté dans une base b est

$$n_{10} = \sum_{i=0}^{k} a_i b^i$$

Exemple

Valeur de b	Expression en base b	Forme développée	Expression en base dix
b = 10	1101	$1 \times 10^3 + 1 \times 10^2 + 0 \times 10^1 + 1 \times 10^0$	$= 1101$
b = 8	$\overline{1101}_8$	$1 \times 8^3 + 1 \times 8^2 + 0 \times 8^1 + 1 \times 8^0$	$= 577$
b = 2	$\overline{1101}_2$	$1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$	$= 13$

1.1 Bases fréquentes

- la base deux (système binaire), en électronique numérique et informatique
- la base huit (système octal), en informatique, davantage à l'échelle humaine que la base deux
- la base douze (système duodécimal), de manière embryonnaire, a été utilisée par les Égyptiens pour le compte en heures et mois
- la base seize (système hexadécimal), en informatique, facilitant les conversions en base deux en regroupant des chiffres binaires, 16 étant une puissance de 2
- la base soixante (système sexagésimal), dans la mesure du temps et des angles, a été utilisée par les Sumériens, les Akkadiens, puis les Babyloniens

Les chiffres arabes qu'on utilise ne suffisent pas dans les bases supérieures à 10. Donc il y a deux méthodes souvent utilisées pour les représentations dans ces bases:

- **Utilisation des lettres**
  - $*$ Le système hexadécimal utilise par exemple les alphanumériques suivants :
    $0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F$
    Donc le $FF_{16} = 15 * 16 + 15 * 16^0$
  - $*$ L'utilisation des lettres est possible seulement pour les bases inférieures ou égales 36 (10 chiffres et 26 lettres) (sauf si on introduit d'autres symboles pour compléter afin de representer des nombres dans une base plus grande que 36)
- **Utilisation d'un séparateur**

  Dans cette forme de représentation, on introduit un symbole qui fera office de séparateur entre les coefficients

  Par exemple le nombre $n = 62 = 1 \times 60 + 2$ s'écrit donc $[1; 2]$ en base 60 (ici on a utilisé **;** comme séparateur).

1.2 Méthode de divisions successives

Intéressons nous à la conversion d'un nombre à base 10 vers une base $b > 1$. Cette conversion s'effectue par des divisions euclidiennes successives des quotients par b , en partant du quotient égal à n jusqu'à obtenir un quotient égale à 0.

La représentation en base b s'obtient en concaténant les restes en partant du bas.

Exemple

- Écrivons le nombre $n = 133$ en base 3

$$
\begin{align}
133 &= 3 * 44 + 2 \tag{1}\\
44 &= 3 * 14 + 2 \tag{2}\\
14 &= 3 * 4 + 2 \tag{3}\\
4 &= 3 * 1 + 1 \tag{4}\\
1 &= 3 * 0 + 1 \tag{5}
\end{align}
$$

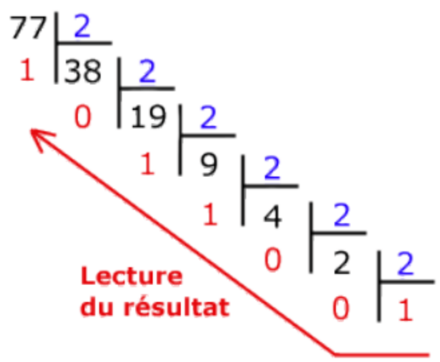
  Donc $n = 133_{10} = 11222_3$

- Écrivons $n = 133$ en base seize.

$$
\begin{align}
133 &= 16 * 8 + 5 \tag{6}\\
8 &= 16 * 0 + 8 \tag{7}\\
& \tag{8}
\end{align}
$$

  Donc $n = 133_{10} = 85_{16}$

  Ici bas une illustration pour la conversion de 77 en binaire.



On a $77_{10} = 1001101_2$

1.3 Évaluation d'une écriture

Pour évaluer la valeur d'une écriture $a_k a_{k-1} \dots a_1 a_0$ dans une base b, il suffit de revenir sur la définition c'est-à-dire calculuer directement

$$n = \sum_{i=0}^{k} a_i b^i$$

2 Exercices

2.1 Entrées

- Convertir en binaire les nombres en base 10 suivants : $255, 1023, 2050$
- Convertir en hexadécimal les nombres binaires suivants: $101_2, 1010001_2, 11111111_2$
- Convertir en hexadécimal les nombres en base 4 suivants : $33, 3030, 2022$

2.2 Plats

1. Écrire un algo et le programme correspondant de conversion d'un nombre entier naturel n en base 10 vers une base $b \geqslant 2$ quelconque à partir de la méthode de divisions successives.
2. Écrire un algo et le programme correspondant de conversion d'un nombre dans une base b vers la base décimale (base 10).
3. Écrire un algo et le programme correspondant de conversion d'un nombre binaire vers la base hexadécimale sans passer par la base décimale
4. Écrire un algo et le programme correspondant de conversion d'un nombre binaire vers la base octale sans passer par la base décimale
5. Écrire un algo et le programme correspondant de conversion d'un nombre en base b vers une base b_1

2.3 Desserts

1. Que pensez vous de la base 1?
2. En vous inspirant du fait que le nombre 0.25 s'écrit en binaire 0.01_2,le nombre 0.625 s'écrit en binaire 0.101, écrire les nombres $0.8125, 0.1$ en binaire.
3. En vous inspirant de ci-dessus et du cours, écrire un algorithme qui convertit un nombre décimal avec des chiffres après la virgule finis en binaire avec virgule
4. D'après vous, pourquoi les ordinateurs utilisent le système binaire?

3 Problème

Le but du problème est de trouver la longueur d'une représentation d'un nombre n dans une base $b \geqslant 2$.

Par exemple $77 = 1001101_2$, donc la longueur d'une représentation de 77 notée $|77|_2$ vaut 7.

i) Montrer qu'entier n a une taille égale à p si et seulement s'il appartient à l'intervalle $[b^{p-1}, b^p - 1]$ Autrement dit,

$$|n|_b = p \Leftrightarrow n \in [b^{p-1}, b^p - 1]$$

ii) En déduire que

$$|n|_b = p \Leftrightarrow p - 1 \leqslant \log_b n < p$$

avec $\log_b n = \frac{\ln n}{\ln b}$

iii) En déduire finalement que

$$p = \lfloor \log_b n \rfloor + 1$$

avec $\lfloor x \rfloor$ est la partie entière de x

4 Encore un problème

Reprenons la question 3 de la section 3.2. On s'intéresse en détail sur la conversion d'un nombre dans une base b vers la base décimale.

La valeur d'un nombre $n_b = a_k a_{k-1} \dots a_1 a_0$ est

$$n_{10} = \sum_{i=0}^{k} a_i b^i$$

1. La formule nous laisse à penser qu'il faut calculer chacune des puissances de b.
   - Écrire l'algorithme pour calculer la puissance p-ième d'un nombre m de façon itérative.
   - Écrire l'algorithme pour calculer la puissance p-ième d'un nombre m de façon récursive.
   - Écrire l'algorithme pour calculer la puissance p-ième d'un nombre m en utilisant l'exponentiation rapide définie ci-dessus.

   $$\text{puissance}(m,p) = \begin{cases} 1 & \text{si } p = 0 \\ m & \text{si } p = 1 \\ \text{puissance}(m^2, p/2) & \text{si } p \text{ est pair} \\ m * \text{puissance}(m^2, (p-1)/2) & \text{si } p \text{ est impair} \end{cases}$$

   - Pour chacune des fonctions puissances ci-dessus, quelle est la complexité de l'évaluation dans la base décimale d'un nombre en base b $n_b = a_k a_{k-1} \dots a_1 a_0$ (en fonction de k et en comptant le nombre de multiplication)?

2. La valeur en base 10 de n_b n'est rien d'autres que l'évaluation d'un polynôme P en b avec

   $$P(X) = a_k X^k + a_{k-1} X^{k-1} + \dots + a_1 X + a_0$$

   . ***Ruffini-Horner*** fait la remarque qu'on peut écrire ce polynôme sous la forme suivante

   $$P(X) = ((\dots(a_k X + a_{k-1})X + a_{k-2})X + a_{k-3})\dots)X + a_0$$

   Donc en evaluant P en b

   $$P(b) = ((\dots(a_k b + a_{k-1})b + a_{k-2})b + a_{k-3})\dots)b + a_0$$

   Ce qui nous donne un nouvel algorithme:

   - On calcul d'abord $v_0 = a_k b + a_{k-1}$

   $$P(b) = ((\dots\underbrace{(a_k b + a_{k-1})}_{v_0} b + a_{k-2})b + a_{k-2})\dots)b + a_0$$

   - Puis $v1 = v_0 * b + a_{k-2}$

   $$P(b) = ((\dots\underbrace{(v_0)b + a_{k-2})}_{v_1} b + a_{k-3})\dots)b + a_0$$

   - Ainsi de suite

   Écrire alors l'algorithme d'évaluation dans la base décimale d'un nombre en base n $n_b = a_k a_{k-1} \dots a_1 a_0$ en utilisant la méthode de ***Ruffini-Horner***. Quelle est alors la complexité de cet algorithme en fonction de k en comptant le nombre de multiplications.