M1.L1: Série d'exercices sur la représentation de l'information

1 Nombre de bits

Combien de bits sont nécessaires pour disposer d'une représentation distincte pour chacun des éléments des ensembles suivants?

1. les mois d'une année.
2. les jours d'un mois.
3. l'ensemble des symboles utilisés pour les nombres romain jusqu'à mille.
4. les étudiants à l'EPFL (supposons 11'000).
5. chaque habitant de la planète selon le nombre estimé sur Worldometers.[1]

2 Représentation des entiers naturels et décimaux positifs

1. Convertir des nombres suivants exprimés dans la représentation positionnelle en base 2 vers la base 10. Les chiffres à droite du point séparant partie entière et partie fractionnaire sont facteurs des puissance négatives de 2, en commençant par 2^{-1} etc..: **101** ; **10.1** ; **1.01** ; **0.101**

2. En fait, 101 en base 2 est l'expression développée de : $1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$.
   question: par quelles multiplications simples passe-t-on du nombre binaire **101** aux autres nombres de la question 1 ? Par quelle autre méthode aurait-on pu trouver la valeur des nombres de la question 1 ?

3. Utilisez la propriété de la multiplication par une puissance de la base 2 pour trouver le plus rapidement possible la valeur en décimal du nombre binaire **0.1101** x $\mathbf{16_{10}}$

3 Domaine couvert des entiers positifs et négatifs avec la <u>représentation du complément à 2</u>, dépassement de capacité et overflow

1. Conversions : pour simplifier on suppose qu'on travaille avec des entiers représentés avec <u>8 bits</u> seulement.
   (a) Quelle valeur est représentée par ces motifs binaires : 00000110, 11111001, 10000110?
   (b) Quel est le motif binaire des nombres suivants : 0, -12_{10}, -1_{10}, 127_{10}, -128_{10} ?
   (c) Quelle conclusion tirez-vous de ces exemples?

2. (a) Toujours avec des entiers sur 8 bits, quel est le résultat en binaire de 64 + 64?
   (b) Comment appelle-t-on le phénomène observé?
   (c) Comment se traduit le dépassement de capacité avec la représentation des nombres négatifs en complément à 2 ? Illustrer sur des cas produisant un dépassement de capacité (= une retenue ignorée)
   (d) Cela pose-t-il un problème?

4 Bases 8 (octal) et 16 (hexadécimal)

Avant de faire les exercices de cette Section, il est conseillé de lire les bases théoriques ci-dessous.

4.1 Théorie

L'intérêt des bases 8 et 16 provient de ce qu'elles sont respectivement les puissances 3 et 4 de deux. De ce fait il est facile de convertir des nombres entre toutes ces bases sans avoir besoin d'une machine. Leur avantage principal est de produire une représentation compacte, ce qui réduit beaucoup les risques d'erreurs dans les manipulations par des êtres humains (ex : mise au point de systèmes embarqués).

[1] . Source : http://www.worldometers.info/world-population/.

Notons $b_7b_6b_5b_4b_3b_2b_1b_0$ la forme positionnelle d'un nombre binaire. Sa forme développée est :

$$\begin{array}{lll} b_7 * 2^7 + b_6 * 2^6 & +b_5 * 2^5 + b_4 * 2^4 + b_3 * 2^3 & +b2 * 2^2 + b1 * 2^1 + b0 * 2^0 \\ =(b_7 * 2^1 + b_6 * 2^0) * 8^2 & +(b_5 * 2^2 + b_4 * 2^1 + b_3 * 2^0) * 8^1 & +(b_2 * 2^2 + b_1 * 2^1 + b_0 * 2^0) * 8^0 \end{array}$$

Lorsqu'on met en facteur les puissances de la base 8 de ***droite à gauche***, on voit apparaître des termes comportant au plus trois bits associés aux trois premières puissances de deux. Par construction, la valeur de chaque terme entre parenthèses est comprise entre 0 et 7, ce qui correspond à un des 8 chiffres de la base 8. On voit ici qu'un groupe de 8 bits se résume ainsi à 3 chiffres en octal. Inversement, si on dispose d'un nombre en base 8 et qu'on recherche son expression en binaire, il suffit de remplacer chaque chiffre octal par sa valeur binaire sur trois bits. Par exemple, le nombre octal 351 se traduit en binaire par 11101001.
Le même raisonnement peut être fait avec la base 16 en faisant apparaître les puissances de cette base en facteur de termes comportant quatre bits. La valeur d'un terme est comprise entre 0 et quinze ce qui correspond à un symbole de la base 16 (cf liste ci-dessous). Un groupe de 8 bits se résume donc à seulement 2 chiffres en hexadécimal:

$$= (b_7 * 2^3 + b_6 * 2^2 + b_5 * 2^1 + b_4 * 2^0) * 16^1 + (b_3 * 2^3 + b_2 * 2^2 + b_1 * 2^1 + b_0 * 2^0) * 16^0$$

Inversement, si on dispose d'un nombre en base 16 et qu'on recherche son expression en binaire, il suffit de remplacer chaque chiffre hexadécimal par sa valeur binaire sur quatre bits. Par exemple, le nombre hexadécimal *C*4 se traduit en binaire par 11000100.

Voici la correspondance pour les chiffres de la base 16 (valeur décimale sur la troisième ligne). On utilise les premières lettres de l'alphabet de *A* à *F* pour les quantités de dix à quinze car ces quantités doivent être représentées par un symbole unique :

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

4.2 Exercices

1. Convertir le nombre 613_8 de l'octal vers le binaire.

2. Convertir le nombre $CAFE_{16}$ de l'hexadécimal vers le binaire.

3. Convertir les motifs binaires en octal et en hexadécimal : 10111001 ; 10001010

4. Quelle méthode proposez-vous pour convertir rapidement un nombre de l'octal vers l'hexadécimal et réciproquement ?

5 Représentation binaire des entiers et multiplication égyptienne

Dans cet exercice nous allons redécouvrir la méthode de la multiplication égyptienne qui repose seulement sur la connaissance de l'addition, de la liste des puissances de 2 et de la table de 2 (pas plus).

Supposons qu'on veuille calculer à la main **53_{10} x 37_{10}** .

La solution utilisant seulement l'addition serait d'effectuer 37_{10} additions de la quantité 53_{10} pour obtenir le résultat.

Maintenant supposons qu'on connaisse la liste des premières puissances de 2 et la table de 2. La méthode égyptienne réduit de beaucoup le nombre d'additions ; la voici:

1) Ecrivez côte à côte sur deux colonnes, à gauche les puissances de deux à partir de 1, et à droite la valeur du plus grand opérande qui est la quantité 53_{10}. Cette valeur est doublée d'une ligne à la suivante.
2) Complétez les colonnes jusqu'à la plus grande puissance de 2 présente dans 37_{10} :

1	53
2	106
4	212
…	…
…	…

3) Convertissez 37_{10} en une somme de puissances de 2
4) Barrez les lignes dont les puissances de 2 ne sont pas dans cette somme
5) Additionnez ce qui reste.

A vous de jouer, combien obtenez-vous ? Combien d'additions avez-vous effectuées ?

Voilà une méthode très efficace pour la multiplication d'entiers si on n'a pas de calculatrice. Merci au scribe Ahmès qui l'a recopié sur le papyrus Rhind d'une source plus ancienne maintenant perdue.

Nous allons en reparler de cette approche dans les séries sur les algorithmes.

6 utilisation d'une représentation à virgule fixe pour représenter la mesure d'un thermomètre numérique familial

Un thermomètre numérique familial est destiné à mesurer la température du corps humain sur un intervalle utile de 35 à 43 degrés. Supposons que ce thermomètre dispose seulement d'un octet pour représenter une telle température.

1. Entier naturels : quel est l'intervalle des températures mesurées si on utilise l'octet pour comme représenter les entiers naturels ; quel est le min, le max, l'erreur absolue ?

2. Virgule fixe sans décalage : supposons maintenant qu'on répartisse les 8 bits de l'octet entre partie entière et partie fractionnaire ; quelle répartition faites-vous ? quelle erreur absolue maximum faites-vous avec cette répartition (approximation par troncation) ?

3. Virgule fixe avec décalage : même question que la précédente excepté qu'on représente cette fois une quantité qui est ajoutée à la valeur minimum implicite de béta =35 degrés. Quelle répartition faites-vous entre partie entière et partie fractionnaire ? Quelle erreur absolue faites-vous ?

7 Représentation des nombres à virgule flottante, erreur absolue, erreur relative, précision

Rappel: soit une valeur **x** qui est représentée (par troncation) avec la valeur **x'**. On appelle :

- **erreur absolue** δx la quantité: $\delta x = |x - x'|$
- **erreur relative** la quantité: $\delta x \ / \ |x|$
- La **précision ε** est une quantité qui caractérise un format de représentation ; elle est donnée par la valeur du *bit de poids faible de la mantisse*. C'est un majorant de l'erreur relative sur le domaine couvert par la représentation utilisée. Dans le pire des cas, l'erreur relative sur le domaine couvert est légèrement inférieure à la valeur donnée par la précision. Le pire des cas tend vers la valeur de la précision lorsque le nombre de bits de la mantisse augmente.

Méthode de travail:

- si on exploite un format à virgule flottante simple tels que ceux vus en cours et dans l'exercice 7.1, on utilise les définitions ci-dessus pour calculer les erreurs absolue et relative.

- Lorsqu'on travaille avec une représentation "*professionnelle*" telle que **IEEE 754** (virgule flottante simple précision sur 32 bits ou double précision sur 64 bits) présentée dans l'exercice 7.2, il devient très fastidieux de savoir quelle est la valeur représentée x' pour une valeur x quelconque. Dans un tel contexte, on utilise la précision **ε** comme une approximation de l'erreur relative maximum par excès (sur tout le domaine couvert sauf la plage des valeurs qui inclut zéro). A partir de cette approximation de l'erreur relative maximum on en déduit l'erreur absolue maximum **δx** qui peut être faite sur un nombre x comme suit:
    - sachant que $\varepsilon = \delta x / x$ alors $\delta x = \varepsilon . x$

7.1 Soit une représentation simplifiée des nombres flottants positifs à l'aide de **5** bits de la manière suivante : **2** bits pour l'exposant de la base 2, **3** bits pour la mantisse.

1) Indiquez le minimum et le maximum représentables [min, max].

2) indiquer toutes les autres valeurs représentées par ce format en les regroupant par valeur de l'exposant ; montrer l'endroit où se situe la virgule dans la représentation binaire, pour chaque valeur de l'exposant.

3) Déterminer L'erreur absolue maximum pour chaque ensemble de valeurs représentées ayant une même puissance de la base deux; ces erreurs absolues sont-elles identiques sur l'intervalle [min, max] ?

4) Pour les valeurs **x** suivantes, indiquez si elles sont exactement représentables avec ce format et si ça n'est pas le cas, indiquez la valeur représentées x' obtenue par troncation, ainsi que l'erreur absolue et l'erreur relative pour chaque cas : **1,12** ; **1,375** ; **8,75** ; **12,75** .

5) Dans quelle partie d'une plage de valeurs représentées avec une puissance constante de la base trouve-t-on l'erreur relative la plus importante ?

6) Quelle est la **précision** de cette représentation ? (l'erreur relative maximum tendrait vers cette valeur de **précision** si le nombre de bits de la mantisse augmentait)

7.2 Soit la représentation en **virgule flottante simple précision** définie par le standard **IEEE 754** sur **32 bits**. Cette représentation utilise *un bit pour le signe du nombre, **8** bits pour **l'exposant** et **23** bits pour la **mantisse**.*

Le nombre représenté dans ce format est donné par l'expression:

$$\textbf{Nombre} = \text{signe}\ 2^{\text{exposant}-127}.1,\text{mantisse}$$

On remarque en particulier que la puissance de la base 2 est obtenue est soustrayant **127** à l'entier naturel obtenu avec le motif binaire que nous appelons "**exposant".** Cela permet de représenter des puissances négatives pour représenter les nombres entre 0 et 1.

A propos des quantités négatives dans la représentation en virgule flottante IEEE 754:

- il n'y a pas de complément à deux dans la représentation de l'exposant; tous les bits sont utilisés pour représenter une quantité positive. Cette méthode de "la soustraction de la quantité 127" est justifiée par le fait qu'elle permet de représenter la quantité zéro lorsque tous les bits sont à 0.

- il n'y a pas non plus de complément à deux sur la mantisse si le bit de signe est à 1. La représentation travaille avec la valeur absolue. Cette approche est différente des entiers car au niveau du processeur ce sont des circuits spécialisés qui traitent les entier d'une part et les nombres à virgule flottante d'autre part.

Questions:

1) Soit le motif binaire suivant pour l'exposant: 01111111. quelle est la valeur représentée si on a seulement des 0 pour la mantisse ?

2) Soit le motif binaire suivant pour l'exposant: 10000000. quelle est la valeur représentée si on a seulement des 0 pour la mantisse ?

3) Soit le motif binaire suivant pour l'exposant: 10000000. quelle est la valeur représentée si on a un 1 pour le poids fort de la mantisse et des 0 pour le reste ?

4) Soit le nombre hexadécimal : $C0400000_{16}$. Quelle valeur est représentée si on vous dit qu'il s'agit du motif binaire d'un nombre représenté en virgule flottante simple précision ?

5) En théorie quelle est le plus grand nombre représentable avec cette représentation lorsque la mantisse contient des zéros partout ? (en pratique le standard IEEE 754 utilise ce motif binaire pour coder des cas particuliers comme nous le verrons en programmation)

6) Quelle est la valeur du plus petit nombre positif représentable strictement supérieur à zéro ?

7) Quelle est la précision de cette représentation ? Convertir la puissance négative de 2 en décimal (avec une calculatrice) pour en déduire le nombre de *chiffres significatifs* maximum que l'on peut obtenir en base dix avec cette représentation. Combien y en a-t-il ?

8) Quel est le nombre X_1 à partir duquel l'erreur absolue maximum vaut 1 ? Il est suffisant d'exprimer ce nombre par une puissance de 2.

9) Quel est le nombre X_2 à partir duquel l'erreur absolue maximum vaut 2 ? Il est suffisant d'exprimer ce nombre par une puissance de 2.

10) Sur l'axe réel indiquez X_2 , ainsi que des valeurs représentées qui le précèdent et qui le suivent dans ce format. reprenez l'exemple du cours concernant la perte de l'associativité et expliquez-le avec votre dessin.

11) A partir de quelle puissance de 2 a-t-on une perte d'associativité lorsqu'on ajoute deux fois la quantité 8 à cette puissance de 2 ?