

Codages d'information, système de numération

M. Dubacq

S1D 2009

1 Préparatifs

Objectif : *Connaître les puissances de 2 et les bases de la numération hexadécimale.*

Écrivez la liste de toutes les puissances de 2, de 2^{-4} à 2^{16} .

0,06125 - 0,125 - 0,25 - 0,5 - 1 - 2 - 4 - 8 - 16 - 32 - 64 - 128 - 256 - 512 - 1024 - 2048 - 4096 - 8192 - 16384 - 32768 - 65536

Et jusqu'à 1024, c'est à savoir par cœur. Aucune discussion à avoir. Écrivez une table de conversion des chiffres hexadécimaux et octaux vers le codage natuel écrit en binaire (4 bits ou 3 bits).

0 - 0000, 1 - 0001, 2 - 0010, 3 - 0011, 4 - 0100, 5 - 0101, 6 - 0110, 7 - 0111, 8 - 1000, 9 - 1001, A - 1010, B - 1011, C - 1100, D - 1101, E - 1110, F - 1111. Et pour les octaux, la même chose de 0 à 7 (on peut mettre seulement sur 3 bits pour l'octal, en supprimant le 0 initial).

C'est à savoir par cœur. Aucune discussion à avoir.

2 De la fonction à l'algorithme

Objectif : *Maîtriser la différence entre fonction et algorithme. Concevoir un algorithme simple. Examiner l'influence de la représentation sur l'algorithme utilisé pour une même fonction.*

La numération grecque (simple) est proche de la numération romaine que vous connaissez : on note les nombres comme suit :

1	5	10	50	100	500	1 000	5 000	10 000	50 000
I	Γ	Δ	Γ _Δ	X	Γ _X	H	Γ _H	M	Γ _M

C'est à la différence près que l'on a pas de règle soustractive : le nombre 4 s'écrit IIII, pas IIΓ. La position des chiffres n'a théoriquement aucune importance, mais on les classait dans l'ordre décroissant de valeur.

- Ce système est-il un système de numération positionnelle? *Non*
- Écrivez votre âge et votre date de naissance en numération grecque.
HΓ_XXXXXΓ_ΔΔΔIII et ΔΔΔΓI pour moi.
- Écrivez un algorithme d'addition des nombres représentés en numération grecque. Est-ce que cet algorithme est le même qu'en décimal?
*On prend les nombres à additionner, on les colle les uns aux autres, on rassemble les chiffres identiques. Ensuite on répète en partant de la droite : tant que l'on a cinq (ou deux) chiffres identiques, on les enlève et on fabrique un chiffre suivant à leur place.
C'est un exemple, il y en a plusieurs possibles.*
- Faites l'addition de votre âge et de votre année de naissance avec votre algorithme (vous devriez obtenir HHΓIII ou HHΓIII). De quelle représentations partez-vous? *On part des représentations grecques, bien sûr. Ou alors des représentations en décimal, que l'on convertit, mais 1- ça a déjà été fait et 2- ça ne doit pas faire partie de l'algorithme.*
- Faites la même chose en décimal. De quelles représentations partez-vous? Est-ce que l'algorithme est le même? Est-ce que la fonction calculée est la même?
La représentation en décimal sert de départ, l'algorithme est différent (même si un peu similaire), mais le résultat (et donc la fonction calculée) est identique.

3 Quantité d'information

Objectif : *Comprendre la nature logarithmique de la mesure en bits.*

- Dans votre classe de TD, jusqu'à combien devez-vous aller si vous donnez un numéro unique à chaque étudiant en partant de 0? Écrivez ces nombres en binaire. Combien de bits d'information sont nécessaires pour désigner un élève parmi tous les autres?
Normalement, on numérote entre 0 et environ 22, ce qui doit faire en binaire 10110 et leur faire comprendre qu'il faut 5 bits pour chaque étudiant.

2. S'il y a un élève de plus, combien de bits supplémentaires seraient nécessaires? *Probablement 0.*
3. Combien d'élèves sont nécessaires pour qu'on ait besoin de 6 bits? Et 7 bits? et 10 bits? *6 bits : au moins 33 étudiants (pas 32, ni 64). Ensuite, c'est 65, et pour 10 bits, c'est 513 étudiants.*

4 Conversions

Objectif : Comprendre la différence entre quantité d'information et longueur de codage, savoir faire des conversions bits \leftrightarrow octets, savoir faire des multiplications sans calculatrice, comprendre la différence entre l'échelle binaire et l'échelle décimale pour les unités.

1. En Syldavie, les plaques d'immatriculation comportent 3 chiffres (décimaux) et une lettre (l'alphabet syldave comporte 32 lettres). Combien de plaques différentes peuvent exister? Combien de bits d'information représente une voiture (parmi les autres)?
32000, $\lceil \log_2(32000) \rceil = 15$ ($2^{15} = 32768$).
2. Combien de bits faut-il pour coder un chiffre décimal? Une lettre syldave? *4 bits et 5 bits, respectivement.*
3. Si on met bout à bout les codages de trois chiffres décimaux et d'une lettre syldave, quelle est la longueur du codage d'une voiture syldave? Comment expliquez-vous la différence avec la réponse à la question 1?
17 bits, ce qui est plus que les 15 annoncés. En effet, il y a à chaque fois qu'on code un chiffre décimal des combinaisons dont on ne se sert pas. À force, ça finit par faire une différence. En fait, on pourrait coder en binaire plus de voitures, mais on ne pourrait pas les traduire en un nombre à 3 chiffres et une lettre.
4. Le nouveau ministre des transports syldave décide de changer les plaques pour utiliser trois lettres. Est-ce que c'est possible, sachant que toutes les plaques sont utilisées?
 $32 \times 32 \times 32 = 32768 = 2^{15}$, ce qui est plus grand que 32000, donc c'est possible.
5. Convertissez 24×10^8 bits en Go. *$24 \times 10^8 / 8 = 3 \times 10^8 = 0,3\text{Go}$*
6. Convertissez 2^{16} octets en Mib. Donnez une approximation en Mb. Quel est l'ordre de grandeur de l'approximation faite? *$8 \times 2^{16} = 2^{19} = 0,5\text{Mib}$, soit environ 0,5 Mb, à 5% près.*
7. Un élément d'ordinateur est capable d'émettre 1024 bits en 0,5 nanosecondes. Quel est le débit (quantité d'information divisée par le temps) de cet élément en bits par secondes? Quelle est la bonne unité pour ce débit?
 $1024 \text{ bits en } 0,5 \text{ nanosecondes} = 1024 / (0,5 \times 10^{-9}) = 2048 \times 10^9 \text{ bits/secondes}$. L'unité appropriée est sans doute le Tb/s (et pas le Tib/s, car si on a un 2048, on a pas une pure puissance de 2, et la division du résultat par 2^{40} n'est pas un entier du tout...).

5 Conversion en binaire et hexadécimal

Objectif : Savoir convertir des nombres de binaire/hexadécimal en décimal et inversement, maîtriser l'écriture des réels dans d'autres bases, comprendre la notion de poids des chiffres dans un système positionnel

1. Écrivez en binaire et en hexadécimal les nombres décimaux suivants : 28 ; 149 ; 1285 ; 0,3125 ; 164,3125.
1 1100 ; 111 1010 ; 100 0000 0001 1001 ; 0,0101 ; 1010 0100,0101. 0x1C ; 0x7A ; 0x4019 ; 0x0,5 ; 0x54,A.
2. Convertissez en décimal les nombres suivants : 0x48 ; 0xA1C ; 0b1010010010011111 ; 0b1010,0011.
72 ; 2588 ; 42143 ; 10,1875.
3. Comment trouver midi à quatorze heures?
En base 8. Ne pas insister sur cette question.

6 Calcul en binaire et hexadécimal

Objectif : Maîtriser les conversions et les additions en base 2 et 16, maîtriser la multiplication en base 2, comprendre la notion de poids des chiffres dans un système positionnel

1. Faites les additions en binaire : 0b1101 0101+0b1110 0101 ; 0b1, 1+0b110+0b100, 1+0b111, 1+0b1010, 1+0b100, 1.
1 1011 1010 et 100010, 1 (ils peuvent vérifier en décimal).
2. Faites les opérations suivantes en hexadécimal : 0x122 + 0x233, 0x87 + 0x54, 0x18 + 0x9, 2 × 0xED, 0x100 − 0x3.
355,DB,21,1DA,FD
3. Faites la multiplication suivante : 17 × 129 à la fois en décimal et en binaire.
2193 et 100010010001.
4. Faites la multiplication suivante en binaire : 110110 × 1101.
*1010111110 (54*13=702)*

7 Codage des entiers

Objectif : Comprendre la différence entre codage et écriture d'un nombre, maîtriser les quatre codages classiques des entiers, comprendre les limites des codages

Ce tableau comporte des cases inutilisées (la première ligne est un exemple). Complétez-le :

Décimal	Écriture Binaire	Type de codage	Codage (binaire)	Codage (hexa)
-18	-1 0010	VA+S (8 bits)	1001 0010	0x92
424	110101000	NAT (16 bits)	0000000110101000	0x01A8
-138	-1000 1010	C2 (16 bits)	1111111101110110	0xFF76
-115	-111 0011	C1 (8 bits)	10001100	0x8C
-4197	-10000 01100101	VA+S (24 bits)	100000000001 000001100101	0x801065
-84	-101 0100	C1 (8 bits)	1010 1011	0xAB
341	101010101	NAT (8 bits)	Impossible!	XXX

Rappel : pour décoder un complément à 2, il faut : inverser les bits si signe=1, ajouter 1 à la valeur obtenue, on a la valeur absolue. Pour encoder un complément à 2, il faut soustraire 1 à la valeur à encoder, inverser les bits (si le signe est égal à 1). Si le signe est positif, rien à faire de différent de NAT.

8 Codage des nombres flottants

Objectif : Comprendre la notation scientifique, le procédé de codage par concaténation de champs, la conversion binaire/hexadécimal

Ce tableau comporte des cases inutilisées. Complétez-le :

Décimal	Binaire	Virgule flottante	E	Codage IEEE754			Hexa
				S	E(8b)	M(23b)	
19,5	10011,1	$1,00111 \times 2^4$	131	0	10000011	00111 $\underbrace{0\dots0}_{18 \text{ fois}}$	419C0000
-7,5	111,1	$1,111 \times 2^2$	129	1	10000001	111 $\underbrace{0\dots0}_{20 \text{ fois}}$	C0F00000
-46,25	101110,01	$1,0111001 \times 2^5$	132	1	10000100	0111001 $\underbrace{0\dots0}_{16 \text{ fois}}$	C2390000
0,3125	0,0101	$1,01 \times 2^{-2}$	125	0	01111101	01 $\underbrace{0\dots0}_{21 \text{ fois}}$	3EA00000
-0,1875	0,0011	$1,1 \times 2^{-3}$	124	1	01111100	1 $\underbrace{0\dots0}_{22 \text{ fois}}$	BE400000
$+\infty$	---	---	255	0	11111111	$\underbrace{0\dots0}_{23 \text{ fois}}$	7F800000
0	0	---	0	0	00000000	$\underbrace{0\dots0}_{23 \text{ fois}}$	00000000
$-26,375 \times 2^{40}$	$-11010,011 \times 2^{40}$	$-1,1010011 \times 2^{44}$	171	1	10101011	1010011 $\underbrace{0\dots0}_{16 \text{ fois}}$	D5D30000

PS : Si vous voulez vous entraîner chez vous, utilisez le programme C suivant (sans explications) :

```
#include <stdint.h>
#include <stdio.h>
main () { float f; union {uint32_t integer;float ieeeseptcinquatre;} n;
printf("Entrez un nombre en base 10: ");scanf("%f",&f);
n.ieeeseptcinquatre=f;printf("IEEE754: 0x%08X\n",n.integer); }
```

9 Codage de texte

Objectif : Comprendre le codage du texte

1. Soit le texte (guillemets non-compris) « Les sanglots longs des violons de l'automne, blessent mon cœur d'une langueur monotone. » Est-il possible de représenter ce texte dans le jeu de caractères ASCII?

Non, à cause du œ. Le œ n'est d'ailleurs pas une lettre en français; par contre le ch en espagnol l'est (voir n'importe quel dictionnaire tchèque, où ch n'est pas entre cg et ci, mais juste après hz; c'était aussi le cas en espagnol jusqu'à une réforme en 1994).

2. Dans le jeu de caractère ISO-8859-15 (dit latin-9), il est possible de coder ce texte. Chaque caractère est alors codé par un octet unique. Quelle est la taille du fichier qui contient uniquement ce texte? *85 octets (et non 86)*

3. Un polonais lit sur son vieil ordinateur le texte précédent. Il voit qu'une des lettres a été remplacée par " (c'est un double accent aigu, comme dans Erdős, et pas un tréma comme dans Gwenaël). Laquelle et pourquoi? S'il renvoie le texte tel quel a son correspondant français du début, que verra le français et pourquoi?

En vérité, il y a de bonnes chances qu'il lise son texte comme étant du ISO-8859-2, et non pas du ISO-8859-15, donc il verra un double accent aigu à la place de sa lettre. Mais le contenu du fichier est inchangé; s'il est renvoyé au français, le texte apparaîtra normalement.

L'encodage d'un fichier ne peut pas être deviné simplement comme ça (il faut faire une analyse des mots pour déterminer la langue et donc l'encodage probable).

NB : bien sûr, il peut y avoir des problèmes; les logiciels de courrier indiquent parfois l'encodage des pièces jointes, même s'il a été mal deviné; certains éditeurs de texte sauvegardent les textes dans un encodage différent de celui qui a été deviné pour l'ouverture... bref, les problèmes peuvent exister. Mais le fichier n'est a priori pas modifié sauf logiciels qui ne fonctionnent pas bien.

10 Le format UTF-8

Objectif : Comprendre le codage du texte, comprendre un format, comprendre la différence entre jeu de caractères et codage des caractères.

Le format UTF-8 est un codage des nombres entre 0x0000 et 0x1FFFFFF (prévu pour coder le jeu de caractère Unicode). C'est le codage suivant (les lettres en italique représentent des bits, qu'il faut reporter dans la troisième colonne) :

Valeurs	Écriture binaire	Codage UTF-8 (binaire)	taille UTF-8
0x0–0x7F	<i>abc defg</i>	0abc defg	1 octet
0x80–0x7FF	<i>abc defg hijk</i>	110a bcde 10fg hijk	2 octets
0x800–0xFFFF	<i>abcd efgh ijkl mnop</i>	1110abcd 10ef ghij 10kl mnop	3 octets
0x10000–0x1FFFFFF	<i>a bcde fghi jklm nopqrstu</i>	11110abc 10de fghi 10jk lmno 10pq rstu	4 octets

Par exemple, le numéro 0x5D0 est encodée par la deuxième ligne (à cause de sa valeur). Traduit en binaire, il vaut 0b101 1101 0000. Il est donc codé sur deux octets : Le premier vaut 11010111, et le deuxième vaut 10010000. C'est-à-dire qu'il est codé par les deux octets 0xD7 et 0x90 (on écrit très souvent les octets en hexadécimal pour gagner de la place et facilité de lecture).

Vous veillerez à systématiquement écrire les résultats binaires aussi en hexadécimal !

1. Qu'est-ce qu'un jeu de caractère? Que représente-t-on avec?

Un jeu de caractère, c'est un codage numérique de caractères (lettres, signes divers) qui permet de représenter un texte.

2. Le caractère de numéro 0x0041 (A) est codé par quel(s) octet(s) en UTF-8? *0x41*

3. Le caractère de numéro 0x00E9 (é) est codé par quel(s) octet(s) en UTF-8? *0xc3 0xa9*

4. Le caractère de numéro 0x0F03 (ཨླ) est codé par quel(s) octet(s) en UTF-8? *0xe0 0xbc 0x83* C'est le caractère *GTER YIG MGO 'IM GTER SHEG MA* en tibétain (à vos souhaits).

5. Le caractère de numéro 0x12084 (𐎠𐎢𐎡𐎣) est codé par quel(s) octet(s) en UTF-8? *0xf0 0x92 0x82 0x84* (*4 octets*) C'est le caractère *DOUN* en cunéiforme (babylonien).

6. Dans un fichier codé en UTF-8, on trouve les six octets suivants. Combien de caractères sont réellement codés dans ce texte?

0xE6 0x9D 0x8c 0xDE 0xBC 0x43

3 caractères (un sur trois octets, un sur deux, un sur un)

Ce code a l'avantage que l'on peut aussi trouver facilement en lisant une suite d'octets représentant de l'UTF-8 combien d'octets occupe chaque caractère codé : on les écrit en binaire, et on sait automatiquement avec le premier octet dans quelle ligne on se trouve, et donc combien d'octets sont utilisés pour le caractère. On peut alors sauter au caractère suivant facilement.

- L'anglais n'utilise que des caractères dont le numéro est dans la première ligne, et est codé traditionnellement en ISO-8859-1 (1 caractère = 1 octet). Le français utilise 5% de caractères de la deuxième ligne (le reste de la première), et est codé pareil (1 caractère = 1 octet). L'arabe (le russe, l'hébreu, le grec) sont aussi codés traditionnellement par 1 caractère = 1 octet, et comportent 95% de caractères de la deuxième ligne (le reste de la première ligne). Le chinois, en revanche est traditionnellement codé en BIG5 (1 caractère = 2 octets). Les textes chinois sont à 99% des caractères de la troisième ligne (le reste de la première ligne).
Depuis quelques années, les codages changent : on utilise de moins en moins le codage traditionnel et de plus en plus le codage UTF-8.
- Pour un texte de 1000 caractères codé en UTF-8, combien d'octets seront utilisés en moyenne pour un texte anglais, français, russe et chinois? *Anglais : 1000 octets. Français : 1050 octets. Russe : 1950 octets. Chinois : 2980 octets.*
- Quel est en chinois le pourcentage d'augmentation de la taille du texte par rapport au codage traditionnel? $(2980 - 1000)/1000 = 1980/1000 = 198\%$

11 Comprendre un format

Objectif : *Maîtriser les conversions hexadécimal/binaire, comprendre un format complexe d'après sa description, connaître un format simple d'image noir et blanc*

Cet exercice est l'étude du codage PBM qui code des images en noir et blanc. Une image est décrite comme un rectangle largeur × hauteur de carrés qui peuvent être soit noirs, soit blanc.

Le format est le suivant : D'abord le caractère P (codé comme dans la table ASCII donnée en cours) sur un octet, suivi du caractère 4. Ensuite, on trouve une série de caractères blancs (un ou plusieurs, voir ci-dessous pour la définition d'un caractère blanc). Ensuite vient la largeur de l'image, sous forme décimale, écrite en ASCII (un octet par chiffre), puis un ou plusieurs caractères blancs. Ensuite vient la hauteur de l'image (comme pour la largeur, en décimal et en ASCII) suivi d'un unique caractère blanc. La partie décrite jusque là s'appelle *l'entête* du format.

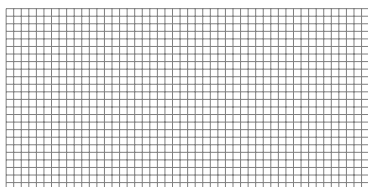
Après l'entête, on va coder bit par bit les pixels de l'image. Un pixel blanc se code par un 0, un noir par un 1. Si la largeur de la ligne n'est pas divisible par 8, on fait comme si l'image était plus large que réellement (on va jusqu'au multiple de 8 immédiatement supérieur, et tout ce qui ne fait pas partie de l'image est considéré blanc), mais bien sûr on ne modifie pas dans l'entête la largeur de l'image (on la laisse à sa valeur réelle).

Les caractères blancs sont les caractères de code hexadécimal 0x09, 0x0A, 0x0D ou 0x20. Ils correspondent au caractère de tabulation (TAB), au caractère de nouvelle ligne (LF), au caractère de retour chariot (CR) et à l'espace (SP). Votre enseignant pourra vous expliquer l'utilisation habituelle de ces caractères.

- En utilisant votre cours, trouvez le code du caractère P et le code du caractère 4. Donnez-le en hexadécimal, en binaire, en octal, en décimal.

P vaut 0120 en octal, 0b1010000, 80, 0x50. 4 vaut 064, 0b0110100, 52, 0x34.

- Combien de bits sont nécessaires pour coder de cette façon une image de taille 8 pixels de large et 1 de haut? Et pour une image de 1024 pixels de large et 1024 de haut? Donnez à chaque fois la taille de l'entête et la taille totale. *L'entête d'une image a une entête qui a une taille minimale de 5 plus le nombre de chiffres de largeur/hauteur de l'image (exemple : P4 8 1). Ensuite, il faut compter 1 octet par 8 pixels, sachant qu'on arrondit la largeur de l'image au multiple de 8 supérieur. Donc, 8 × 1 donne une image de taille 8 octets (64 bits), et 1024 × 1024 donne une image de taille 5 + 2 × 4 + 1024 × 1024/8 = 131085 octets, soit environ 128 kio (ou ko abusivement), ou alors 131 ko (en comptant un kilo-octet=1000 octets). Ils doivent pouvoir trouver 128 kio seuls (rappeler que 1024 = 2¹⁰), pour le 131 leur donner (et qu'ils constatent que ça fait une différence non négligeable).*



- Le contenu d'un fichier (représenté ici en hexadécimal) est comme suit :

ADRESSE	OCTETS EN HEXADÉCIMAL
00000000	50 34 0a 31 36 20 31 36 0a 00 04 06 ca 09 55 11
00000010	23 10 04 20 04 40 04 40 04 46 08 49 30 28 a0 68
00000020	a0 38 e0 34 e0 3f fc 07 ee

Transformez chacun des cinq premiers octets en utilisant la table ASCII du cours pour faire le début de déchiffrement du fichier. Est-ce qu'il y a des octets de ce fichier qu'on ne peut pas transformer ainsi? *P,4,(passage à la ligne),1,6. Il y en a, par exemple fc, qui ne sont pas dans la table puisque la table est initialement prévue pour 7 bits de codage. C'est parce que tout le fichier n'est pas codé selon la table ASCII.*

- Quelle est la dimension de cette image? *Les premiers octets se décodent en ASCII comme étant P4<NL>16 16<NL>. Donc l'image est de taille 16 par 16.*
- Quelle est la taille de ce fichier? Comment se répartit entête et corps? *41 octets dont 9 d'entête et 32 d'image.*
- En utilisant le quadrillage ci-dessus, décidez le fichier. Que représente cette image? *Un dromadaire..*