Numérique et sciences informatiques
Première

B. Représentation de l'information
4. Représentation numérique des textes

Un préambule : la différence entre « édition de texte » et « traitement de texte ».

• Il faut bien distinguer :
* le codage « nu » d'un texte: le texte est vu comme une simple succession de caractères ;
* le codage « riche » d'un texte : qui inclut aussi
- un codage de différents éléments de mise en page (police de caractère, taille des caractères, présentation)
- des éléments de structures : débuts de chapitres, de sections, etc.
- des métadonnées : date de création, auteur du texte, mots-clés, etc.

• Pour observer cette différence, regardons le poids d'un même texte enregistré comme :
* texte brut dans un **éditeur de texte** (avec l'extension **`.txt`**) ;
* un document dans un **logiciel traitement de texte** comme *Writer* de *Libre Office* (extension **`.odt`**) ou *Word* de *Microsoft* (extension **`.docx`**) par exemple).

• Le code source d'un programme informatique (par exemple le fichier "`monProgramme.py`") doit être une simple succession de caractères pour pouvoir être interprété par l'ordinateur : on ne peut donc pas le taper dans un logiciel de traitement de texte comme *MS Word* ou *LO Writer*. Il nous faut un éditeur de texte.

• Qu'est-ce que la **coloration syntaxique** ? ...

• Ici nous n'envisageons les textes que comme une succession de caractères et donc nous nous intéresserons qu'au codage des caractères qui le composent.

I Une ancienne version du codage des caractères : le code ASCII

• Le principe général du codage numérique d'un texte est qu'un texte doit être considérer comme une succession finie de caractères. Pour représenter numériquement un texte, il suffit donc d'associer à chaque caractère un nombre entier positif : son **point de code** ou **index**. Cet entier sera à son tour « encodé » en binaire. L'index associé à chaque caractère et l'encodage de chaque index doit faire l'objet de normes très précises pour qu'ils soient partagés par tous les ordinateurs qui doivent communiquer entre eux.

• Il faut pouvoir représenter de très nombreux caractères pour pouvoir numériser des textes écrits dans toutes les langues du monde entier.

• Il faut aussi représenter des caractères « invisibles », dits **caractères non imprimables** : qui délivrent cependant des informations : la fin de ligne, la fin de fichier, etc.

• Il faut cependant concevoir un encodage le plus compact possible qui permette de limiter l'occupation de la mémoire.

I.1 Présentation de la norme ASCII

• La première tentative pour mettre de l'ordre dans les normes de codage du texte en informatique a été la norme ASCII (pour *American Standard Code for Information Interchange*) apparues en 1963.

• L'idée centrale était de coder chaque caractère sur un octet mais en réservant un bit pour le contrôle de parité (le bit de poids fort). Le codage s'effectuait donc sur 7 bits. Puisque $2^7 = 128$, ce système permet de coder *128 caractères différents*.

• Principe du codage :
* A un caractère correspond un entier compris entre 0 et 128 : son **index** ou **point de code**.
* Chacun de ces entiers est enregistré en mémoire sur un octet sous la forme d'un codage binaire : il s'agit de son **encodage**

• Les caractères représentés en ASCII :
  * 26 lettres latines minuscules : de 97 à 122 ;
  * 26 lettres latines majuscules : de 65 à 90 ;
  * 10 chiffres arabes : de 30 à 39 ;
  * des symboles de ponctuation : virgule, point, point-virgule, point d'exclamation, point d'interrogation, parenthèses ouvrante et fermante, crochet ouvrant et fermant, etc.
  * des caractères de mise en page (espace, tabulation, fin de ligne, saut de page, suppression, etc.)
  * des caractères de contrôle : début d'en-tête, début de texte, fin de texte, fin de transmission, accusé de réception, etc.

• Le bit de contrôle, dit aussi **bit de parité**, est fixé de telle façon que le nombre de '1' dans l'octet soit toujours pair.

I.3 Les limitations du code Ascii

• De très nombreux caractères utilisés dans les langues du monde entier ne sont pas représentés dans le code ASCII. Il limite donc considérablement les possibilités d'écriture de la plupart des langues autres que l'anglais, même lorsque ces langues utilisent un alphabet latin.

• Dans le cas du français, il manque notamment les caractères accentués : é, è, ê, ë, à, ô, ô mais aussi ç, œ, æ, etc..

I.4 Une extension de la norme Ascii : ISO 8859

• Le développement de techniques de contrôle plus efficaces que le bit de parité ont permis de concevoir des extensions du code Ascii qui utilisent le 8ème bit de l'octet. Ceci permet de multiplier par deux le nombre de caractères codés et donc de coder 256 caractères.

• La norme **ISO 8859** apparue en 1986 proposait différentes extensions pour différentes langues. Elle comportait 16 tables différentes, chacune correspondant à un groupe de langues particulier (dont 10 pour les langues utilisant l'alphabet latin que l'on désignait par « latin-n »). Parmi celles-ci , la norme ISO 8859-1 dite « latin-1 » était la norme utilisée pour les langues d'Europe occidentale comme le français ou l'allemand.

• L'idée était de conserver les 128 premières valeurs aux caractères représentés en Ascii et d'utiliser les 128 suivantes pour des caractères plus spécifiques au groupe de langues concerné.

• L'avantage de ces extensions est de maintenir un codage de chaque caractère sur un octet.

I.5 Importance historique du code Ascii

• L'importance historique du codage Ascii dans le monde informatique se manifeste dans le fait qu'il continue encore aujourd'hui d'imposer des limites aux caractères utilisables dans certains domaines : nom de domaine, adresse mail, caractère disponible dans le BIOS, etc.

II Unicode

II.1 La norme ISO 10646

• L'idée est de créer un code qui attribue un point de code à tous les caractères de systèmes d'écriture du monde entier, les symboles de monnaie, des emojis, etc.

• Aujourd'hui la norme (**ISO 10646**) intègre plus de 143 000 caractères. La capacité maximale est d'envirion 4,3 milliards de caractères.

• Les 256 premières positions coïncident avec la norme latin-1, donc les 128 premières correspondent à la norme Ascii.

• On écrit le point de code d'un caractère sous la forme U+xxxx, où chaque x représente un chiffre en hexadécimal. Par exemple, le point de code de la lettre 'o' est U+006F car le point de code est '6F' en hexadécimal, soit '111' en décimal.

• Associer un point de code (un entier positif) à chaque caractère ne suffit pas à définir la représentation d'un caractère dans la mémoire de l'ordinateur. Il faut aussi choisir **l'encodage** : c'est-à-dire préciser comment chaque point de code va être représenté dans la mémoire de l'ordinateur.

II.2 L'encodage UTF-8

• La norme Unicode vise à définir l'encodage de chacun des caractères représentés par la norme ISO 10646. Elle a élaboré différents encodages appelés « formats de transformation universel » (en anglais **UTF** pour « *Universal Transformation Format* »). L'encodage le plus courant d'Unicode est **UTF-8**, ce qui signifie qu'il s'agit d'un encodage qui utilise *au minimum* 8 bits soit un octet pour coder un point de code.

• L'idée générale est en effet que tous les points de code ne seront pas exprimés par le même nombre de bits. Les plus courants seront codés sur 1 octet, les moins courants sur 4 octets.

• Une caractéristique majeure de l'encodage UTF-8 est qu'il est compatible avec le code ASCII. En effet, les caractères qui étaient déjà pris en charge par Ascii sont codés sur un seul octet et avec la même représentation en mémoire.

• Le principe de UTF-8 :

Point de code	Nb d'octets utilisés	Nb de bits significatifs	Forme du codage
De 0 à 127	1	7	**0********
De 128 à 2047	2	11	**110***** **10******
De 2048 à 65 535	3	16	**1110**** **10****** **10******
De 65 536 à 2 097 152	4	21	**11110*** **10****** **10****** **10******

• Quelques exemples :

Le caractère	Nombre d'octets	Point de code en décimal	Point de code en binaire	Point de code en hexadécimal
'A'	1	65	100 0001	41
'é'	2	233	1110 1001	C3 A9
'€'	3	8364	10 0000 1010 1100	E2 82 AC

• Les avantage de l'UTF-8 :
  * compatibilité avec d'ancien traitement fondé sur le code Ascii ;
  * gain en mémoire si le texte est écrit dans une langue faisant appel aux caractères latins car les caractères les plus fréquents seront codés sur un seul octet.

• Un inconvénient : il faut lire le premier octet pour savoir sur combien d'octets le caractère est codé, puis extraire les « bits significatifs » de chaque octet. Cela implique une certaine lenteur de lecture.

• D'autres encodages de Unicode :
  * UTF-16
  * UTF-32