

PF1 – Contrôle n°3

14 novembre 2013

Durée : 40 minutes

Rappel : 1pouce = 2.54cm

Exercice 1. [Des pouces et des poings]

1. Combien de bits sont nécessaires pour coder une couleur avec le système RGB ? Combien y a-t-il de couleur différentes possibles ?

Réponse. *On code chaque couleur primaire sur 8 bits soit 24 bits en tout. Ce qui fait 2^{24} couleurs différentes (environ 16.7 millions).*

2. Combien pèse, en octet, une image 320×240 , couleurs RGB ?

Réponse. *L'image comporte 320×240 pixels qu'on stocke sur 24 bits soit 3 octets. Donc le poids de l'image est $320 \times 240 \times 3 = 230400$ octets.*

3. On scanne, avec un scanner 300 dpi, en noir et blanc, un texte sur une feuille de 7.62cm par 25.4cm. Combien pèsera l'image bitmap obtenue ?

Réponse. *La feuille mesure $7.62/2.54 = 3$ pouces sur $25.4/2.54 = 10$ pouces. Le nombre de pixels de l'image scannée en 300 dpi sera donc $3 * 300 * 10 * 300 = 2700000$. Or chaque pixel est stocké sur 1 bit (noir et blanc), donc l'image pèse 2700000 bits soit 337500 octets.*

4. Je souhaite représenter mon arbre généalogique sur une image. Quel format ai-je intérêt à utiliser ? Justifier brièvement.

Réponse. *Un arbre généalogique est essentiellement composé de figures géométriques, il est donc facile de le représenter avec une image vectoriel. De plus, ce choix est plus avantageux car l'image est plus simple à modifier (on peut rajouter une branche en déplaçant les objets géométriques, ce qu'on ne peut pas faire avec le bitmap) et plus simple à lire (on peut zoomer sur certaines parties sans perte de qualité).*

Exercice 2. [Échantillon de son]

1. Rappelez ce qu'est l'échantillonnage lors de la numérisation du son.

Réponse. *Un son peut être représenté par une courbe. Afin de numériser cette courbe, on l'approxime par une fonction en escalier : c'est l'échantillonnage.*

2. Combien pèsera, en octet, un enregistrement mono de 2 minutes 40 du dernier concert de Claude François échantillonné à 44kHz sur 16 bits ?

Réponse. *L'échantillonnage à 44kHz signifie qu'on a approximé chaque seconde d'enregistrement par 44000 valeurs, chacune stockée sur 16 bits soit 2 octets. L'enregistrement durant en tout 160 secondes, son poids est $44000 \times 2 \times 160 = 14080000$ octets.*

3. Un fichier son mono, échantillonné à 10kHz sur 8 bits, pèse 50 000 octets. Combien de secondes dure l'enregistrement ?

Réponse. *Chaque seconde d'enregistrement pèse $10000 \times 1 = 10000$ octets. Donc cet enregistrement dure 5 secondes.*

Exercice 3. [Leçon de Gameboy] Avant, bien avant que l'on ait des cartes sons sophistiquées, on avait seulement la possibilité d'émettre des bips à des hauteurs différentes (12 notes différentes, qu'on peut émettre à 4 octaves différentes). On pouvait alors coder "une note" avec 8 bits :

- 4 bits pour indiquer la note jouée
- 2 bits pour indiquer la hauteur de la note
- 2 bits pour la durée de la note

Un morceau est alors une succession de note de hauteur et de durée différentes.

1. La taille du fichier dépend-elle de la durée du morceau ? Justifiez.

Réponse. *Non, la taille du fichier dépend du nombre de note. Deux notes courtes seront codées avec 16 bits alors qu'une note longue avec seulement 8 bits.*

2. Combien d'octets faut-il pour jouer "Do do do ré mi ré do mi ré ré do" ?

Réponse. *Il y a 11 notes dans ce morceau, il faut donc 11 octets pour le représenter.*

3. On peut bien sûr améliorer ce système en offrant plusieurs timbres d'instruments pour chaque note, la possibilité de jouer plusieurs notes en même temps etc. C'est le principe du format MIDI. Donner un avantage et un inconvénient de ce système de représentation du son par rapport à celui vu en cours.

Réponse. *Le type MIDI est au son ce que le type vectoriel est à l'image. En effet, on ne pourra pas représenter un vrai enregistrement sonore en MIDI (de la même façon qu'on ne pourra pas représenter une vraie photo en vectoriel). Cependant, on peut éditer très facilement des fichiers MIDI, ajouter des instruments, ralentir le tempo, changer la tonalité, générer la partition, etc.*

Exercice 4. [Codage de texte à longueur variable] Dans le système ASCII, chaque caractère est codé par un nombre fixé de bits. On peut imaginer des systèmes où ce n'est pas le cas.

1. Quel serait l'intérêt d'un tel codage pour un texte français où la lettre e est codée sur 1 bit ?

Réponse. *Si on a un codage des lettres à longueur variable et que les lettres les plus fréquentes dans un texte ont un code court, alors le texte sera représenté avec moins de bits que dans le cas d'un codage à longueur fixée comme ASCII : on aura des fichiers plus légers. C'est le principe de la compression de Huffman.*

2. Quel est le problème du codage : $[a] = 101$, $[b] = 10$, $[c] = 110$?

Réponse. *On ne peut pas décoder de façon unique une succession de bits. En effet, $[ab] = 10110 = [bc]$.*

Pour remédier à ce problème, on peut construire des codages en utilisant des arbres (voir la figure 1). Pour trouver le code d'une lettre, on part du haut de l'arbre et on descend jusqu'à la lettre. Ce chemin peut s'écrire comme une suite de 0 et de 1 : 0 quand on est allé à gauche, 1 quand on est allé à droite : c'est le code de la lettre. Par exemple, dans la figure 1, le code de la lettre "o" est 110.

3. En utilisant la figure 1, donnez le code du mot "babacool".

Réponse. *On cherche le code de chaque lettre en parcourant l'arbre : $[b] = 100$, $[a] = 00$, $[c] = 101$, $[o] = 110$ et $[l] = 111$ d'où $[babacool] = 1000010000101110110111$. Si on avait codé nos 6 lettres sur 3 bits, le codage de babacool aurait été d'une longueur de 24 bits, alors qu'ici 22 suffisent.*

4. Quel mot est codé par 1010011101 ?

Réponse. *On suit le chemin indiqué par le mot dans l'arbre. Chaque fois qu'on tombe sur une lettre, on l'écrit et on revient à la racine du mot. On trouve alors le mot "cale".*

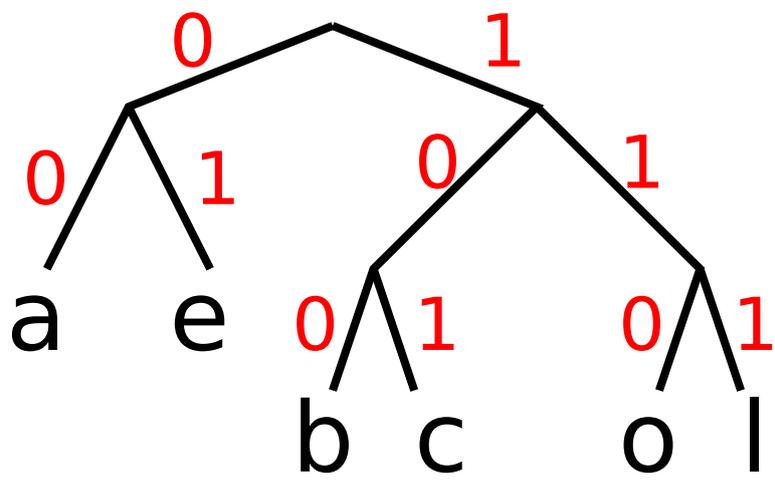


FIGURE 1 – Un arbre pour coder du texte