

## Semaines 11 & 12 : Systèmes informatiques [Solutions]

### 1 Un programme à étudier

a) Le résultat est le suivant :

r0	r1	r2	r3
17	123	365	473

- b) Ce programme trie le contenu des registres **r0**, **r1**, **r2** et **r3** pour avoir la valeur la plus petite dans **r0** et la plus grande dans **r3**. Il s'agit ici du « tri bulles » évoqué à la leçon I.1, décrit comme un des algorithmes de tri les plus simples à écrire mais aussi l'un des moins performants.
- c) D'un côté, on pourrait penser que rien ne changerait : qu'on inverse ou pas deux éléments identiques ne semble pas avoir de conséquences (à part une perte de performance due à l'exécution d'une opération inutile). Le problème est que, une fois l'échange inutile fait, on met à 1 le registre **r4** pour indiquer qu'au moins un échange a été fait et qu'il faut réitérer la boucle principale ; une fois que les valeurs contenues dans les registres **r0**, **r1**, **r2** et **r3** sont en ordre et si deux ou plus de ces valeurs sont identiques, on se retrouve dans une boucle infinie qui continue à échanger inutilement les valeurs identiques entre elles sans jamais terminer. Cette modification du programme serait donc une erreur !

### 2 Somme des valeurs absolues

Une solution possible est la suivante :

```

0: cont_ppe  0, r0, 2
1: oppose   r0          ## ou soustrait    r0, 0, r0
2: cont_ppe  0, r1, 4
3: oppose   r1
4: somme     r2, r0, r1

```

Note : « **oppose** » n'est pas difficile à imaginer : c'est le « complément à 2 » que nous avons vu en leçon I.4 !

### 3 Circuit

Le circuit réalise une porte XOR (« ou exclusif ») :

a	b	sortie
0	0	0
0	1	1
1	0	1
1	1	0

## 4 Synthèse musicale

### 4.1 Besoin de structure

*Retrouvez l'amplitude du signal sonore au temps 0.01 s dans l'information suivante stockée sur le disque :*

C'est évidemment **impossible**!...

Il faut la **structure** de ces informations pour les décoder. C'est justement le but du reste de l'exercice.

### 4.2 Echantillonnage du signal

*Ce signal peut-il être reconstitué exactement à partir des échantillons du CD ?*

**Oui sans souci** : la bande passante est de  $5 \times 440 = 2200$  Hz qui est bien inférieur à  $44100/2 = 22'050$  Hz.

*Sinon, que devrait-on faire ?*

Sinon on devrait **filtrer les hautes fréquences**, celles supérieures à 22'050 Hz.

### 4.3 Codage du signal

*Quelle précision a-t-on alors relativement à l'amplitude maximale ?*

Sur 16 bits on peut stocker 65'536 valeurs, on a donc une précision relative de... **1/32768**. Et oui ! Il ne faut pas oublier que les amplitudes peuvent être négatives, il nous faut donc un bit de signe !

*Est-ce possible de compresser à 50% ?*

On parle *bien sûr* ici de compression sans perte et sans préfixe (pour décoder), donc le th. de Shannon s'applique : on **ne peut pas** compresser en dessous de 10.5 bit par valeur, soit une compression de  $1 - 10.5/16 = 34\%$ .

*Quel taux de compression obtient-on avec Huffman (ordre de grandeur) ?*

Toujours par le th. de Shannon :  $H \leq L \leq H + 1$  donc on a entre 10.5 et 11.5 bit par valeur en moyenne,

soit entre 28% et 34% de compression. Disons **30%**.

#### 4.4 Stockage sur disque : contenu du fichier

*Combien de blocs (environ) utilise notre fichier ?*

1 seconde de musique donne 44'100 échantillons, soit environ 485'100 bits (longueur moyenne précédente), soit 60'637 octets, soit environ 60 Kio, c'est-à-dire 15 blocs.

Pour 2 s, il faut donc **30 blocs**.

*Nous avons un disque de 512 Gio : combien de blocs contient-il ?*

$$512 \text{ Gio} = 2^9 \cdot 2^{10} \cdot 2^{10} \text{ Kio} = 2^{29} \text{ Kio} = 2^{27} \text{ blocs}$$

*Combien de bits sont nécessaires pour pour l'adresse des blocs ?*

Il faut donc (au moins) **27 bits**.

*Quelle est l'adresse du bloc correspondant à l'amplitude du signal sonore au temps 0.01 s ?*

0.01 s correspond au 441<sup>e</sup> échantillon, c'est-à-dire à 4851 bits, soient 606 octets : c'est donc dans le 1<sup>er</sup> bloc.

On doit donc décoder la 1<sup>re</sup> adresse, c'est-à-dire la 2<sup>e</sup> ligne, soit 0000000000000000000000000101011, qui représente **43** en binaire.

#### 4.5 Stockage sur disque : nom absolu du fichier

*Combien de blocs utilise alors le répertoire **Musique** ?*

La taille de la table de **Musique** est de  $(1 + 30) \cdot 32$  bits (1 pour la taille et 30 adresses), soient 992 bits. **1 bloc** suffit donc.

*Quel est le début du fichier représentant le répertoire **musche** ?*

```
00000000000000000000000000000001 <- taille = 1
00000000000000000000000000000101010 <- 42
```

## 4.6 Décodage du « message » de départ

*Quelle est la valeur de l'amplitude de notre signal au temps  $2/44100$  s ?*

On décode : en 42 on trouve la table de notre fichier de musique. Le temps  $2/44100$  s correspond au 3<sup>e</sup> échantillon, qui est donc dans le 1<sup>er</sup> bloc, dont l'adresse est 43 (décodée plus haut).

On doit donc décoder le bloc 43, qui se décompose comme suit en utilisant le code de Huffman : 11001100110, 10101010, 0001110001

ce qui correspond donc aux amplitudes (relatives, codées sur 16 bits) : 0, 5929 et 11056.

Celle qui nous intéresse est 11056, qui nous donne au final une amplitude (physique) de  $4 \cdot 11056 / 32768 = 1.35$ .

## 5 Couches Internet

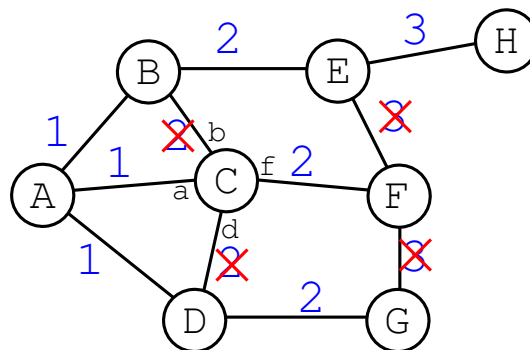
Dans l'Internet, « IP » est un protocole de la couche niveau 3 (réseau) utile pour une fonction de routage.

Dans l'Internet, « TCP » est un protocole de la couche niveau 4 (transport) utile pour une fonction de connexion entre machines/services.

Dans l'Internet, « HTTP » est un protocole de la couche niveau 5 (application) utile pour une fonction de communication de contenus formatés (pages Web).

Dans l'Internet, « DNS » est un protocole de la couche niveau 5 (application) utile pour une fonction de résolution de noms en adresses ( $\simeq$  bottin).

## 6 Routage IP

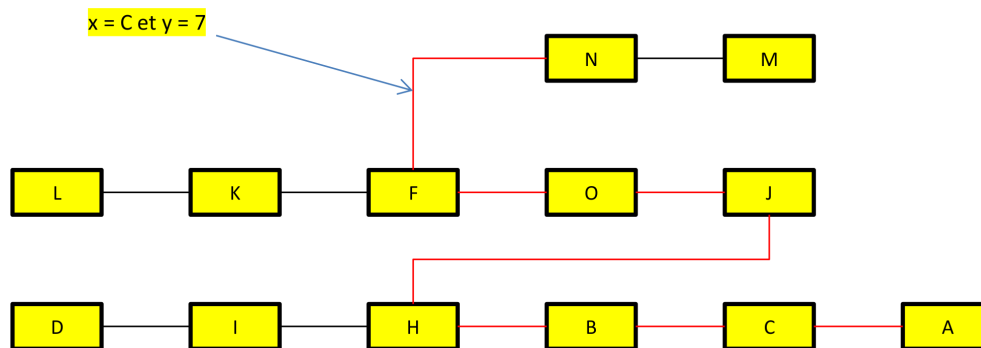


Par exemple (solution non unique) :

table de C

A	1	a
B	1	b
D	1	d
E	2	b
F	1	f
G	2	d
H	3	f

## 7 Routage encore



## 8 Routage toujours

a)

C			D			E		
dest.	dir.	dist.	dest.	dir.	dist.	dest.	dir.	dist.
A	A	1	A	C	2	A	C	2
K	E	4	K	E	4	K	G	3

F			G			H			I		
dest.	dir.	dist.	dest.	dir.	dist.	dest.	dir.	dist.	dest.	dir.	dist.
A	E	3	A	E	3	A	G	4	A	G	4
K	G	3	K	I	2	K	I	2	K	K	1

b)

C			D			E		
dest.	dir.	dist.	dest.	dir.	dist.	dest.	dir.	dist.
K	E	5	K	F	4	K	F	4

F			G			H			I		
dest.	dir.	dist.	dest.	dir.	dist.	dest.	dir.	dist.	dest.	dir.	dist.
			A	F	4	A	F	4	A	G	5

c)

C			D			E		
dest.	dir.	dist.	dest.	dir.	dist.	dest.	dir.	dist.
K	D	5	K	F	4			

F			G			H			I		
dest.	dir.	dist.	dest.	dir.	dist.	dest.	dir.	dist.	dest.	dir.	dist.
A	D	3	A	F	4	A	F	4	A	G	5

## 9 Encore un peu plus de routage ?

oui et la longueur est 4 :

