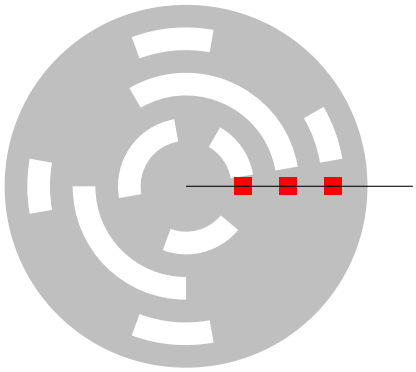
Codage

Emmanuel Jeandel
http://www.lif.univ-mrs.fr/~ejeandel/

24 janvier 2011

1 Codeur angulaire

Un codeur angulaire est un mécanisme qui permet de transformer un angle en une information binaire.



La barre est fixe et contient 3 leds. La roue peut bouger. Derrière le disque se trouvent 3 capteurs, positionnés en face des 3 leds. Les capteurs recoivent, ou non, la lumière provenant des leds : Si la roue est opaque (représenté en gris sur le dessin), sur le chemin entre le capteur et la led, le capteur ne reçoit rien. Si la roue est transparente (le blanc), le capteur reçoit la lumière. On code par 1 la présence de lumière sur le capteur et par 0 son absence.

Q 1) On fait subir à la roue une rotation de 360 degrés dans le sens des aiguilles d'une montre. Donner l'information fournie par chacun des capteurs correspondant à la roue représentée ci-dessus pendant toute la durée de la rotation.

Q 2) Construire une roue de sorte que l'information reçue lors de la rotation de 360 degrés soit successivement 000, 001, 010 ..... 111 puis 000.

En pratique, on ne peut évidemment pas obtenir l'information des capteurs à tous les instants, on les interroge fréquemment, par exemple toutes les 10 millisecondes. Il est pratiquement impossible d'obtenir l'information des 3 capteurs exactement au même moment, il y aura forcément un (très léger) décalage entre le temps où la mesure a été prise sur chacun des 3 capteurs.

Q 3) Pour la roue obtenue précédemment, expliquer l'information reçue par les capteurs pendant le court instant où la roue effectue une rotation entre 179 et 181 degrés. Quels sont les problèmes qui se posent ?

Q 4) Proposer un autre codage des 8 secteurs anguleux qui évite ce problème.

Les codes de *Gray* de niveau n sont définis de la façon suivante. Le code de Gray de niveau 1 est $0, 1$.

On note $a_1, \ldots, a_p$ le code de Gray d'ordre n. Alors le code de Gray d'ordre $n+1$ est défini par $0a_1, 0a_2, 0a_3, \ldots 0a_p, 1a_p, 1a_{p-1}, 1a_{p-2}, \ldots 1a_1$.

Ainsi le code de Gray de niveau 2 est $0a_1, 0a_2, 1a_2, 1a_1$, où $a_1, a_2 = 0, 1$ désigne le code de niveau 1, donc le code de niveau 2 est $00, 01, 11, 10$.

Q 5) Calculer les codes de Gray de niveau 3 et 4.

Q 6) Prouver que les codes de Gray ont la propriété recherchée : deux codes consécutifs ne diffèrent que par un seul bit. On utilisera une récurrence.

Q 7) Donner, en fonction de n, la valeur du dernier code de Gray (c'est à dire du code a_8 du code de Gray niveau 3, du code a_{16} du code de Gray niveau 4 etc.)

2 Codes ambigus

Q 1) Déterminer, parmi les ensembles suivants, ceux qui sont des codes non-ambigus :

1. $\{0, 10, 11\}$
2. $\{0, 01, 11\}$
3. $\{0, 01, 10\}$
4. $\{110, 11, 10\}$
5. $\{110, 11, 100, 00, 10\}$
6. $\{11001, 010, 001, 110, 0\}$

Q 2) Construire un code non ambigu contenant 3 mots de longueur 2, et 2 mots de longueur 3.

Q 3) Construire un code non ambigu contenant 4 mots de longueur 3, 5 mots de longueur 4 et 6 mots de longueur 5.

Q 4) Donner un code non-ambigu tels qu'un mot *infini* puisse se décoder de plusieurs façons différentes.

Q 5) On suppose qu'on transmet un message avec un code C. On prend le message en cours de route, de sorte qu'il nous manque les premiers bits du message. De même, la connexion est interrompue, donc il nous manque aussi les derniers bits du message. Pour lesquels de ces codes (qui sont non-ambigus) peut-on quand même retrouver la portion du message qu'on a lu ?

1. $\{0, 01, 011, 0111\}$
2. $\{01, 11, 10, 00\}$
3. $\{0111, 0100\}$
4. $\{0, 01, 11\}$

Q 6) Montrer que si un code est non-ambigu, le code miroir (celui où on lit les codes à l'envers, par exemple 0100 devient 0010) est aussi non-ambigu

Q 7) Montrer que si on enlève un mot à un code non-ambigu, il reste non-ambigu. Donner un exemple simple d'un code non-ambigu qui devient ambigu après ajout d'un mot.

Si C est un code, on note par C^2 l'ensemble des concaténations de deux mots de code de C. Par exemple si $C = \{1, 10\}$, alors $C^2 = \{11, 110, 101, 1010\}$.

Q 8) Montrer que si C est un code non-ambigu, alors C et C^2 n'ont pas d'éléments en commun.

Q 9) Montrer que si C est un code non-ambigu, alors C^2 aussi.

Q 10) Montrer que si C^2 est un code non-ambigu, alors C aussi.

3 TP : `tar`

Le programme `tar` est un logiciel d'archivage. Il permet de regrouper en un seul fichier plusieurs fichiers et répertoires afin, par exemple, de les stocker ensuite plus facilement, ou de les compresser.

On crée le fichier `toto.tar` contenant les fichiers `a.c`, `bc.z` et `16.64` par la commande :

```
tar cvf toto.tar a.c bc.z 16.64
```

On peut extraire les fichiers à partir de l'archive à l'aide de la commande

```
tar xvf toto.tar
```

On se contentera de ces deux opérations dans le TP.

Q 1) Créer deux fichiers `a.txt` et b.txt contenant quelques lignes de texte, et archivez-les dans le fichier `toto.tar`

Un fichier `tar` contenant des fichiers `1.txt`, `2.txt`, ... `n.txt` est constitué :
- D'une entête de 512 bits décrivant le fichier `1.txt`
- Du contenu du fichier `1.txt`
- D'une entête de 512 bits décrivant le fichier `2.txt`
- Du contenu du fichier `2.txt`
- ...
- D'une entête de 512 bits décrivant le fichier `n.txt`
- Du contenu du fichier `n.txt`

Afin de permettre une lecture du fichier plus facile (et également pour des raisons historiques), le contenu du fichier est **aligné sur un multiple de 512** : Par exemple, si le fichier fait 3678 octets, on lui ajoutera artificiellement 418 octets de façon à arriver à 4096, un multiple de 512.

Décrivons maintenant le format des 512 octets de l'entête :

Octets	Contenu	Précisions
0-99	Nom du fichier	Terminé par l'octet `'\0'`
100-107	Permissions	Codé en base 8
108-115	Propriétaire	Codé en base 8
116-123	Groupe	Codé en base 8
124-135	Taille	Codé en base 8
136-147	Date de modification	Codé en base 8
148-155	Checksum	*Voir plus loin*
156-156	Type	Toujours le caractère `'0'` pour nous [1]
157-511	Divers	Rien du tout

Q 2) Exécutez la commande `hd -v toto.tar` qui vous permet de visualiser le contenu (en ascii et hexadécimal) du fichier `toto.tar`

Vérifiez que le format du fichier est similaire à ce qui a été indiqué plus haut. Remarquez en particulier qu'à la fin de chaque champ indiqué ci-dessus, il y a un caractère \0 (ne pas oublier de le mettre).

La checksum est un entier permettant de contrôler la validité de l'entête du fichier. Elle est calculée comme suit :
- On suppose que les caractères 148 à 155 de l'entête sont des symboles espace `' '` (code ascii 32)
- On calcule la somme de tous les (512) caractères de l'entête.
- On remplace les caractères 148 à 155 de l'entête par la somme, écrite en base 8 (en octal).

1. `'0'` signale que le fichier est bien un fichier normal et non pas un répertoire, ni un lien symbolique, etc.

Q 3) Écrire un programme `montar` qui permet d'archiver un ensemble de fichiers.
La commande

```
montar a.c b.txt
```

créera ainsi un fichier test.tar contenant les fichiers `a.c` et `b.txt`. Il est possible soit de créer le fichier `toto.tar`, soit de renvoyer le résultat directement sur l'entrée standard, au choix.

Quelques commandes utiles :

- Si n est un nombre, on peut le convertir en une chaîne de caractère en base 8 utilisant (par exemple) exactement 6 caractères avec la commande

  ```
  s =  "%06o" % n
  ```

- `ord(c)` renvoie le code ascii du caractère `c`.
- La commande `stat` permet de récupérer la plupart des informations dont vous avez besoin. Voilà un exemple très simple d'utilisation (testez-le !)

  ```
  import os
  infos = os.stat("apple.pdf")
  print "Permissions: ",  "%0o" % infos.st_mode
  print "Proprietaire: ", infos.st_uid
  print "Groupe: ", infos.st_gid
  print "Taille: ", infos.st_size
  print "Date: ", infos.st_mtime
  ```

- Pour lire un fichier (binaire), en mettant le résultat à chaque étape dans une chaîne s

  ```
  f = open("apple.pdf", "rb")

  s = f.read(100)
  while s != "":
      faire un truc avec s
      s=f.read(100)
  ```

- Pour écrire dans un fichier (binaire) la chaîne `s` :

  ```
  f = open("test", "w")
  f.write(s)
  ```