TD noté de ILOG Solver, Scheduler

Durée : 4 h

Les solutions sont à envoyer en pièces attachées à `pascalDOTbrissetATenacDOTfr`
Utiliser le nouveau `Makefile` : `ln -s /usr/local/serveur/ILOG/Makefile`

1729 (3 points)

De `www.nrich.maths.org.uk/conference/reports/mcbride.html` :

This number has become associated with Hardy's taxi. The story goes that the remarkable young Indian mathematician Ramanujan came to England to discuss his work with G.H. Hardy. However, Ramanujan's health was not good and he landed up in hospital in Putney. Hardy went to visit him and arrived in a taxi with plate number 1729. Hardy remarked that he thought this was rather a dull number, whatever that was supposed to mean. Ramanujan immediately got very excited and told Hardy that, far from it being boring, 1729 was a very interesting number, being the smallest positive integer which can be written as the sum of two cubes in two different ways, i.e.

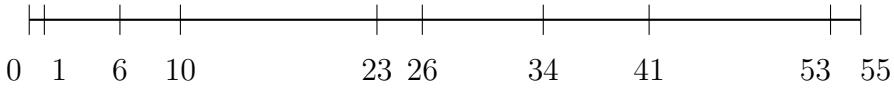
$$n = a^3 + b^3 = c^3 + d^3$$

Retrouver ce nombre magique avec ILOG-Solver (supposer qu'il est inférieur à 10^6).

Règle de Golomb (5 points)

Une règle de Golomb est une règle graduée telle que les distances entre deux graduations quelconques sont toutes différentes. Une règle de Golomb est optimale si la longueur totale de la règle (la distance entre la première graduation et la dernière) est minimale.

Voici un exemple de règle de Golomb optimale à 10 graduations.



Les tailles optimales sont les suivantes :

Nb graduations	2	3	4	5	6	7	8	9	10
Longueur optimale	1	3	6	11	17	25	34	44	55

Écrire un programme de génération de règle de Golomb optimale à n graduations.

Girouette (6 points)

1. Un codage de Gray est un codage binaire des entiers tel que deux codes successifs ne diffèrent que par un seul bit. Le codage binaire standard (00 pour 0, 01 pour 1, 10 pour 2, 11 pour 3) ne respecte pas cette règle car le changement unitaire de 1 à 2 change 2 bits.

   La table suivante est un exemple de code de Gray pour 3 bits

	0	1	2	3	4	5	6	7
1er bit	0	0	0	0	1	1	1	1
2ème bit	0	0	1	1	1	1	0	0
3ème bit	0	1	1	0	0	1	1	0

   Écrire un programme qui fabrique un code de Gray pour n bits en mettant donc en œuvre les contraintes suivantes :
   - chaque valeur entre 0 et $2^n - 1$ est codée (une fois et une seule) ;
   - deux valeurs consécutives diffèrent seulement par 1 bit ;
   - ajouter une contrainte pour fabriquer un code « circulaire » : les codes de 0 et $2^n - 1$ diffèrent également d'un seul bit.

2. On désire mesurer la position angulaire d'un mécanisme tournant : une girouette, un radar sol, ... Pour cela, on fixe sur l'axe du système un disque à encoches, celles-ci pouvant être «lues» par un capteur (optique). Une dent correspond à un 1, une encoche à un 0.

   Le système simple de la figure **??** avec un seul capteur permet une mesure angulaire relative (en comptant l'alternance des 0 et 1)

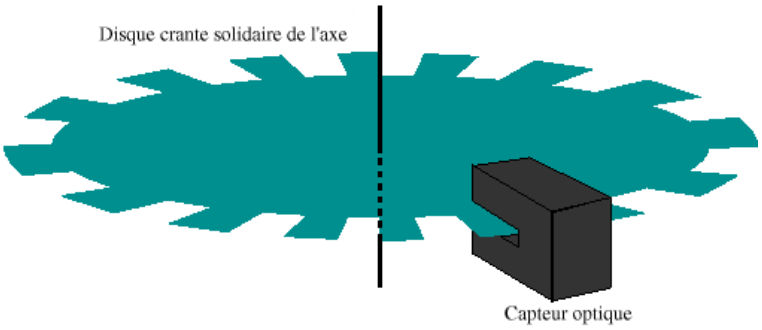


FIG. 1 – Un disque à encoches régulières

   On veut fabriquer un système de mesure angulaire absolue en utilisant plusieurs disques solidaire et plusieurs capteurs ; il s'agit de coder les secteurs angulaires en binaire, chaque bit correspondant à un disque et à un capteur. Par exemple, une précision de 45 degrés sera obtenue avec 3 disques pour $2^3 = 8$ valeurs possibles. Il est important pour un tel système que chaque changement unitaire n'affecte qu'un bit à la fois (ceci pour éviter des phases transitoires fausses : avec le codage standard, le

passage de 1 à 2 se fait via un état transitoire, dû à l'impossibilité matérielle d'aligner exactement les encoches des disques, 00 ou 11). Il faut donc utiliser un code de Gray. La figure **??** correspond au codage de la table à 3 bits donnée précédemment.

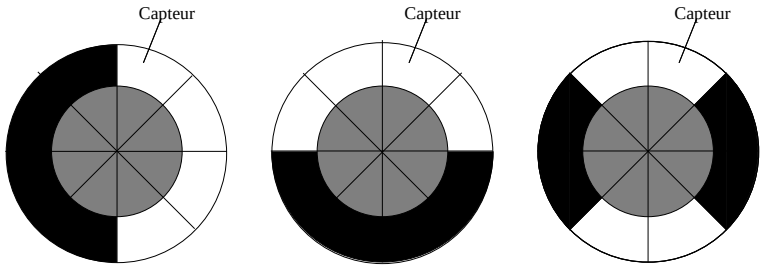


FIG. 2 – Positionnement angulaire absolu à 3 bits

On remarque sur cette figure que les deux premiers disques sont identiques, à un quart de tour prêt. On peut donc «économiser» un disque en positionnant les deux capteur sur le même disque, à 90 degrés l'un de l'autre. On obtient la solution de la figure **??**.

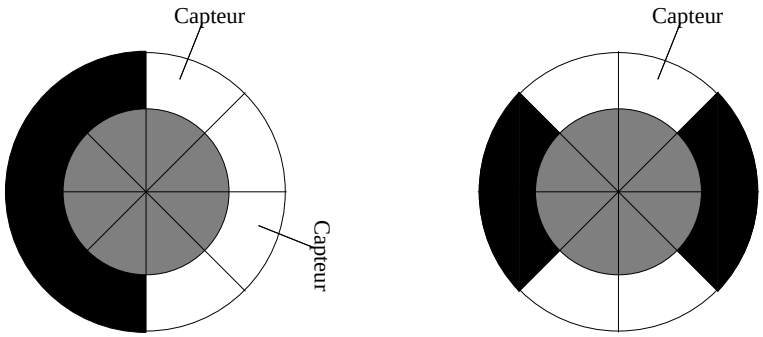


FIG. 3 – Positionnement angulaire absolu à 3 bits avec 2 disques

Fabriquer un codage pour un système de positionnement angulaire à 22,5 degrés (4 bits, ou $2k$) utilisant 2 (ou k) disques et 2 capteurs (à un quart de tour l'un de l'autre) par disque. Indication : utiliser l'opérateur modulo (**%**) pour exprimer les contraintes supplémentaires sur le code de Gray de la question précédente.

Une solution et sa mise en œuvre sont données en figure **??** (disques concentriques et positions des capteurs marquées par des points).

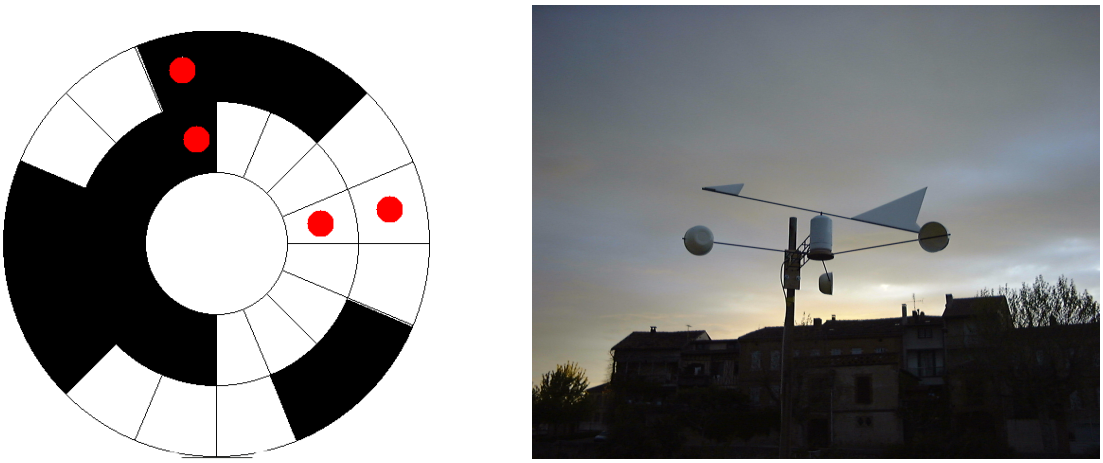


FIG. 4 – Solution 2 disques et 2 bits par disques

Jobshop (6 points)

Soit n *jobs* à effectuer sur m machines. Chaque job est constitué de m tâches dont la durée est fixée et qui doivent être exécutées dans un ordre fixé. Chaque tâche doit être effectuée sur une machine donnée. Le problème est de choisir une date pour les $n \times m$ activités et de minimiser la date de fin de la dernière tâche.

En utilisant `Ilog Scheduler`, résoudre ce problème pour les instances données dans les fichiers `jobshop6.txt` et `jobshop10.txt` dans `/usr/local/serveur/ILOG`. Dans chaque fichier on trouve

- n le nombre de jobs ;
- m le nombre de machine ;
- pour chaque job i (n lignes), pour j de 1 à m (m colonnes), le numéro de la machine devant effectuer la j-ème tâche du job i.
- pour chaque job i, pour j de 1 à m, la durée de la j-ème tâche du job i.

Indication : utiliser des ressources unitaires et appeler la méthode `setEdgeFinder` pour chacune de ces ressource à la définition et la méthode `close` avant la recherche. Utiliser le but `IlcRank` pour la recherche. Documentation dans `/usr/local/ilog/sched43/html/ref/schedule.htm`