Exercice : programmation

Lorsque l'on transmet des données (ici, une chaîne de bits 0 ou 1), il y a toujours un risque que se produise une erreur. Une façon très élémentaire d'essayer de la détecter est d'utiliser une « somme de contrôle » : il s'agit de regrouper les bits par paquets, et d'ajouter à la fin de chaque paquet par exemple la valeur de la somme modulo 2 des bits du paquet.

À la réception, la machine contrôlera que cette somme de contrôle est bien égale à celle du paquet tel qu'il lui est arrivé.

Cette méthode permet de repérer certaines erreurs, mais pas de savoir quel bit est erroné. Nous allons étudier dans cet exercice une méthode un peu plus évoluée qui permet de corriger l'erreur.

Découpons les données en paquets de 64 bits. Nous imaginerons chaque paquet comme une matrice $a = (a_{i,j})_{1\leq i,j\leq 8}$ à 8 lignes et 8 colonnes, construite en remplissant successivement la première ligne de la matrice de la gauche vers la droite, puis la deuxième ....

Nous associons à cette matrice deux vecteurs à 8 bits $l = (l_i)_{1\leq i\leq 8}$ et $c = (c_j)_{1\leq j\leq 8}$ définis par :

$$\forall i \in [|1,8|]\,, \quad l_i \equiv \sum_{k=1}^{8} a_{i,k} \text{ (modulo 2)}$$

$$\forall j \in [|1,8|]\,, \quad c_j \equiv \sum_{k=1}^{8} a_{k,j} \text{ (modulo 2)}$$

La première machine transmet à la deuxième le triplet (a, l, c). Celui-ci est reçu par la deuxième machine comme un triplet (b, m, d), b étant une matrice de $\mathcal{M}_8(\{0,1\})$, m (resp. d) des vecteurs contenant huit éléments égaux à 0 ou 1.

Aux erreurs de transmission éventuelles près, on devrait avoir :

$$a = b, l = m, c = d.$$

On supposera dans toute la suite que le triplet est transmis avec *au plus* une erreur.

1) Dans quel cas la transmission se fait-elle sans erreur ?

   Il peut arriver que l'erreur de transmission se situe dans le vecteur m ou d. Comment s'en rendre compte ?

   Lorsque l'erreur est dans la matrice b, montrer que l'on peut déterminer quel est le bit erroné, et corriger l'erreur.

Programmation en CAML

Cette partie ne s'adresse qu'aux candidats qui ont choisi le langage Caml.

Attention : CAML numérote les vecteurs à partir de 0 et non de 1. On conseille donc vivement d'indicer tous les objets de 0 à 7 plutôt que de 1 à 8.

Nous représenterons les matrices à l'aide du type CAML `int vect vect`.

On rappelle que, si a est un objet de type `int vect vect`, le terme $a_{i,j}$ de la matrice a est obtenu en CAML par `a.(i).(j)`.

2) Écrire en CAML des fonctions :

```
sommes_lignes : int vect vect -> int vect
```
et
```
sommes_colonnes : int vect vect -> int vect
```

   telles que `sommes_lignes a` (resp. `sommes_colonnes a`) renvoie le vecteur l (resp. c) associé à la matrice a.

3) Écrire une fonction CAML

```
correcteur : int vect vect -> int vect -> int vect -> int vect vect
```

   telle que `correcteur b m d` renvoie la matrice a.

Programmation en PASCAL

Cette partie ne s'adresse qu'aux candidats qui ont choisi le langage Pascal.

Nous modéliserons les matrices de ce problème par le type Pascal

```
Matrice = ARRAY[1..8,1..8] OF INTEGER
```

et les vecteurs par le type `Vecteur = ARRAY[1..8] OF INTEGER`.

On rappelle que, si A est un objet de type `Matrice`, le terme d'indices i et j est obtenu en Pascal par `A[i, j]`.

2) Écrire des fonctions :

```
function sommes_lignes (a : Matrice) : Vecteur ;
```

et `function sommes_colonnes (a : Matrice) : Vecteur ;`

telles que `sommes_lignes(a)` (resp. `sommes_colonnes(a)`) calcule le vecteur l (resp. c) associé à la matrice a.

3) Écrire une fonction :

```
function correcteur (b : Matrice ; m, d : Vecteur) : Matrice ;
```

telle que `correcteur (b, m, d)` calcule la matrice a.

Problème : circuits logiques, automates

Partie A : circuits logiques

Le but de cette partie est de concevoir un circuit logique qui calcule le reste modulo 3 d'un entier.

Portes logiques élémentaires. On pourra utiliser les portes suivantes :



Porte Non



Porte Et



Porte Ou

Nous représenterons les entiers en base 2.

Ainsi $a_{n-1} \dots a_1 a_0$, où les a_i valent 0 ou 1, représente l'entier $A = \sum_{k=0}^{n-1} a_k 2^k$.

1) Montrer que A est égal modulo 3 à $B = \sum_{k=0}^{n-1} (-1)^k a_k$.

2) Concevoir un circuit logique noté M_1 qui dispose de deux entrées a_0 et a_1, deux sorties s_0 et s_1, et est tel que $a_0 - a_1$ soit égal à $s_0 + 2s_1$ modulo 3 (c'est-à-dire que le nombre $s_1 s_0$ est, en base 2, le reste modulo 3 de $a_1 a_0$).

3) Concevoir un circuit logique, noté D, qui dispose de quatre entrées a_0, a_1, b_0, b_1 et de deux sorties s_0, s_1, tel que $S = s_0 + 2s_1$ soit congru modulo 3 à $A + B$, en notant $A = a_0 + 2a_1$ et $B = b_0 + 2b_1$. On pourra se contenter ici de donner des formules logiques qui expriment s_0 et s_1 en fonction de a_0, a_1, b_0, b_1, sans dessiner le circuit D.

4) Supposons que l'on dispose d'un circuit M_n à 2^n entrées a_0, …, a_{2^n-1} et deux sorties s_0, s_1 tel que $S = s_0 + 2s_1$ soit congru à $A = \sum_{k=0}^{2^n-1} a_k 2^k$ modulo 3. À l'aide de M_n et D, concevoir un circuit M_{n+1} qui résout le problème pour un nombre à 2^{n+1} bits en entrée.

5) Déterminer un équivalent lorsque n tend vers $+\infty$ du nombre de portes élémentaires utilisées dans le circuit M_n, en fonction de n et du nombre a de portes élémentaires utilisées dans le circuit D.

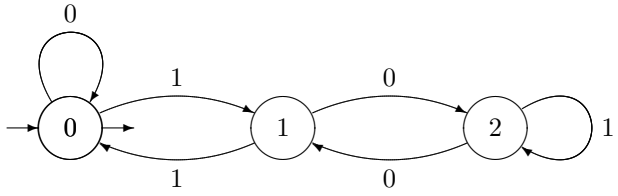
Partie B : automates

Soit les alphabets $A = \{0, 1\}$ et $B = \{0, 1, +, =\}$.

À tout mot $u = a_p a_{p-1} \dots a_0$ de A^*, on associe l'entier $\sigma(u) = \sum_{k=0}^{p} a_k 2^k$.

Soit L_0 l'ensemble des mots u de A^* tels que $\sigma(u)$ est multiple de 3.

Considérons l'automate $\mathcal{C}$ suivant :



Dans cet automate, l'état 0 est à la fois état initial et état final.

1) Montrer que $\mathcal{C}$ reconnaît le langage L_0.
   On admettra que les ensembles L_1 (resp. L_2) des mots u de A^* tels que $\sigma(u)$ est congru à 1 (resp. 2) modulo 3 sont des langages reconnaissables.

2) Un test élémentaire pour repérer certaines erreurs de calcul consiste à vérifier que le résultat est exact modulo 3.
   Soit M l'ensemble des mots de B^* de la forme $u + v = w$, où u, v et w sont des mots de A^*. Ce mot représente l'addition :
   $$\sigma(u) + \sigma(v) = \sigma(w).$$
   Montrer que l'ensemble L des mots de M qui représentent des additions exactes modulo 3 est un langage reconnaissable.