Option Informatique en Spé MP et MP*
DS du mardi 13 décembre 2005 : le corrigé

Autour du lemme de l'étoile

Question 1 Chaque b devant être précédé d'au moins un a, le nombre de b d'un mot de longueur 6 est égal à 2 ou 3. Avec deux b, nous avons les mots $aaabab$, $aabaab$ et $abaaab$; avec trois b, nous avons le mot $ababab$.

Question 2 Je dis que $\varphi(\mathsf{L}) = \{a^k \mid k \geqslant 4\}$. En effet, la longueur d'un mot de L est au moins égale à 4 ; et, si $k \geqslant 4$, alors le mot $a^{k-3}bab$ est de longueur k et appartient à L.

Question 3 Non, puisque ce lemme énonce une condition *nécessaire* pour qu'un langage soit rationnel.

Question 4 Prenons $\mathsf{M} = (ab)^*$. Notons $\mathsf{R} = \mathsf{L} \cap \mathsf{M}$; R est l'ensemble des mots de la forme $(ab)^p$, où p est premier. Utilisons le lemme de l'étoile : supposons R rationnel, alors il existe N tel que tout mot u de R de longueur au moins égale à N se décompose en $u = xyz$ avec $|xy| \leqslant N$, $y \neq \varepsilon$ et $xy^*z \subset \mathsf{R}$. Nous pouvons supposer $N \geqslant 4$ sans perte de généralité. Considérons alors la décomposition associée au mot $u = (ab)^N$, de longueur $2N \geqslant N$. Si $|y|_a \neq |y|_b$, alors $|xz|_a \neq |xz|_b$ et donc $xz \notin \mathsf{R}$. Sinon, deux cas se présentent :

- $x = (ab)^i$, $y = (ab)^j$ avec $j \geqslant 1$, et $z = (ab)^{N-i-j}$. Alors le mot $xy^{N+1}z = (ab)^{2N}$ n'appartient pas à R puisque $2N$ n'est pas premier ;
- $x = (ab)^i a$, $y = (ba)^j$ avec $j \geqslant 1$, et $z = b(ab)^{N-i-j}$. Le même mot $xy^{N+1}z$ permet de conclure.

Finalement, dans tous les cas nous exhibons un mot de le forme xy^qz n'appartenant pas à R, ce qui montre que ce langage n'est pas rationnel.

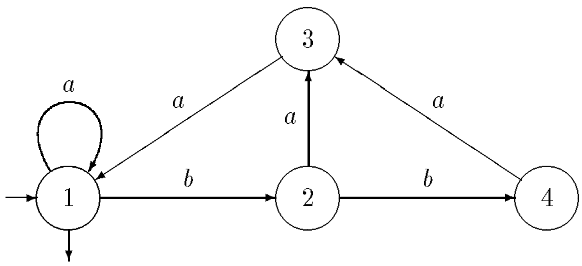
Question 5 Le langage L n'est pas rationnel : s'il l'était, alors le langage $\mathsf{R} = \mathsf{L} \cap \mathsf{M}$ le serait aussi, en tant qu'intersection de deux langages rationnels.

Question 6 La réponse est positive. Supposons L rationnel, alors il existe N tel que tout mot u de L de longueur au moins égale à N se décompose en $u = xyz$ avec $|xy| \leqslant N$, $y \neq \varepsilon$ et $xy^*z \subset \mathsf{L}$. Ici encore, nous pouvons supposer $N \geqslant 4$ sans perte de généralité. Soit p un nombre premier au moins égal à N. Considérons le mot $u = (ab)^p$: il appartient à L, et sa longueur est certainement supérieure à N. Le raisonnement tenu à la question 4 s'applique sans changement, et montre que L n'est pas rationnel.

Fibonacci revient !

Question 1 Énumérons les mots de F par longueur croissante et, à longueur égale, en ordre lexicographique ; nous trouvons ε, a, aa, aaa, baa, $aaaa$, $abaa$, $baaa$ et $bbaa$.

Question 2 L'automate $\mathcal{A}$ ci-dessous comporte quatre états ; il est déterministe, mais non complet.



Nous allons montrer que $\mathcal{A}$ reconnaît F :

- soit u reconnu par $\mathcal{A}$: u est l'étiquette d'un calcul réussi de $\mathcal{A}$. Ce calcul commence et se termine donc dans l'état 1. Il se compose donc de zéro, un ou plusieurs calculs d'étiquette non vide qui, chacun, commencent et se terminent dans l'état 1. L'étiquette de chacun de ces calculs «élémentaires» est donc a, baa ou $bbaa$: donc u appartient à F.
- soit $u \in \mathsf{F}$, distinct de ε (le cas de celui-ci étant clair). u se décompose en $p \leqslant |u|$ facteurs $u_1, \ldots, u_p$ de l'une des trois formes a, baa ou $bbaa$; comme $1 \cdot a = 1 \cdot baa = 1 \cdot bbaa = 1$, nous avons clairement $1 \cdot u = 1$ donc u est reconnu par $\mathcal{A}$.

Question 3 Une expression rationnelle du langage F est $e = (a + baa + bbaa)^*$. En effet :

- soit u un mot décrit par l'expression e ; il est clair que dans un tel mot, chaque b est suivi d'au moins deux caractères, dont le deuxième est un a ; donc u appartient à F ;
- réciproquement, soit $u \in \mathsf{F}$. Si u est de la forme $u = a^k$ avec $k \in \mathbb{N}$, alors u est clairement décrit par e. Sinon, u contient au moins une occurrence de b. Supposons que chaque occurrence de b est suivie immédiatement d'un a : alors elle est en fait suivie immédiatement de deux a. Ici encore, u est décrit par e puisqu'il est la concaténation de mots de la forme a ou baa. Enfin, si u contient des occurrences de bb, chacune d'elles est suivie immédiatement de deux a : dans ce cas, u est la concaténation de mots de l'une des trois formes a, baa ou $bbaa$ et est donc décrit par e.

Question 4 • Voici une rédaction qui respecte strictement les spécifications de l'énoncé:

```
let rec dans_L = function
 | A::q -> dans_L q
 | B::t::A::q -> dans_L (t::A::q)
 | [] -> true
 | _ -> false ;;
```

Question 5 Soit $k \geqslant 3$. Les mots de F de longueur k sont de l'une des trois formes suivantes :

- av, avec $v \in \mathsf{F}$; il y en a n_{k-1} de cette forme ;
- $baav$, avec $v \in \mathsf{F}$; il y en a n_{k-3} de cette forme ;
- $bbaav$, avec $v \in \mathsf{F}$; il y en a n_{k-4} de cette forme.

Ceci nous donne la relation de récurrence $n_k = n_{k-1} + n_{k-3} + n_{k-4}$.

Question 6 Le tableau ci-dessous donne les résultats demandés, et en prime les nombres de Fibonacci :

k	0	1	2	3	4	5	6	7	8	9	10
n_k	1	1	1	2	4	6	9	15	25	40	64
F_k	0	1	1	2	3	5	8	13	21	34	55

Question 7 Au vu du tableau précédent, nous avons déjà $n_0 = 1 \geqslant F_0 = 0$ et $n_1 = 1 \geqslant F_1 = 1$. Supposons l'inégalité $n_j \geqslant F_j$ acquise pour tout $j \leqslant k$. Alors $n_{k+1} = n_k + n_{k-2} + n_{k-3} \geqslant F_k + F_{k-2} + F_{k-3} = F_k + F_{k-1} = F_{k+1}$ ce qui établit l'inégalité au rang $k+1$. Par récurrence, cette inégalité est donc vraie pour tout $k \in \mathbb{N}$.

Question 8 La réponse est négative, puisque la suite de terme général n_k domine la suite de Fibonacci, laquelle majore une suite géométrique de raison $\varphi > 1$.

Question 9 $\beta = \varphi$ convient puisque $F_k = \varphi^k + \mathcal{O}(1)$, et $n_k \geqslant F_k$.

Question 10 Pour $k \geqslant 0$, notons $\mathcal{P}(k)$ l'assertion $n_{k+2} + n_k = F_{k+3}$. Au vu du tableau dressé à la question 6, nous avons $n_2 + n_0 = 2 = F_3$ et $n_3 + n_1 = 3 = F_4$; ceci établit $\mathcal{P}(0)$ et $\mathcal{P}(1)$. Supposons $\mathcal{P}(k)$ et $\mathcal{P}(k+1)$ acquises ; alors $n_{k+4} + n_{k+2} = n_{k+3} + n_{k+1} + n_k + n_{k+2} = F_{k+4} + F_{k+3} = F_{k+5}$, ce qui établit $\mathcal{P}(k+2)$. Donc, par récurrence, $\mathcal{P}(k)$ est vraie quel que soit $k \geqslant 0$.

Question 11 $\gamma = \varphi$ convient, puisque n_k est majoré par F_{k+3}, lequel est équivalent à $\varphi^{k+3}/\sqrt{5}$.

Question 12 L'équation caractéristique associée à la relation de récurrence est $r^4 = r^3 + r + 1$, que l'on écrit $r^4 - r^3 - r - 1 = 0$. Un peu de flair fait apparaître le facteur $r^2 + 1$ (ou, si l'on veut, les solutions i et $-i$). L'autre facteur est $r^2 - r - 1$, dont les racines sont φ et $-1/\varphi$. Il reste à déterminer les quatre coefficients a, b, c et d tels que $n_k = ai^k + b(-i)^k + c\varphi^k + d(-1/\varphi)^k$.

Autour du casse-tête de l'été 2005

Question 1 Observons le bloc 0 : il y a déjà un 6 dans la colonne 1 et dans la colonne 2, donc le 6 ne peut être que dans la colonne 0. La case $(0, 0)$ est occupée, et il y a un 6 dans la ligne 2. Donc le 6 de ce bloc est dans la case $(1, 0)$.

Question 2 Le bloc 1 peut être rempli de $4! = 24$ façons ; le bloc 2 et le bloc 3 de quatre façons chacun ; il ne reste plus qu'un seul choix pour le bloc 4. Ceci nous donne un total de 384 sudokus d'ordre 2.

9	8	7	6	5	4	3	2	1
6	5	4	6	5	4	3	2	1
3	2	1	3	2	1	3	2	1
6	5	4	6	5	4	3	2	1
5	5	4	5	5	4	3	2	1
4	4	3	3	2	1	3	2	1
3	3	3	3	3	3	3	2	1
2	2	2	2	2	2	2	2	1
1	1	1	1	1	1	1	1	1

Question 3 Le tableau ci-contre donne, pour chaque case, un majorant du nombre de choix au cours d'un remplissage par lignes successives. Le produit de ces nombres est $2^{44} \times 3^{24} \times 5^9 \times 7 = 10^9 \times 9^{12} \times 2^{28} \times 2^7 \times 7 < 10^9 \times 10^{12} \times 10^9 \times 10^3 = 10^{33}$.

▶ À l'URL `http://www.shef.ac.uk/~pm1afj/sudoku/`, vous trouverez un article de Bertram Felgenhauer et Frazer Jarvis intitulé *There are 6670903752021072936960 Sudoku grids*. En clair, le nombre de sudokus est environ $6{,}67\,10^{21}$.

Question 4 • Vous avez le choix entre le très concis `it_list (prefix @) []` et le très classique :

```
let rec flat = function
 | [] -> []
 | t::q -> t @ (flat q) ;;
```

Question 5 • La fonction `intervalle` permet une grande concision :

```
let list_of_bande h g d m =
 map (fun j -> m.(h).(j)) (intervalle g d) ;;
```

Question 6 • Pour chaque valeur i de l'indice de ligne, nous construisons la liste des cases de la bande définie par i, g et d ; ensuite, il reste à remettre à plat la liste de listes obtenue, en utilisant `falat` :

```
let list_of_zone h b g d m =
 flat (map (fun i -> list_of_bande i g d m) (intervalle h b)) ;;
```

Question 7 • Il suffit d'appliquer la fonction `list_of_zone` aux bons argument. Le bloc numéro k contient les cases (i, j) avec $g \leqslant i \leqslant g + 2$ et $h \leqslant j \leqslant h + 2$, où $g = 3\lfloor k/3 \rfloor$ et $h = 3(k \bmod 3)$.

```
let list_of_bloc k m = let g = 3*(k/3) and h = 3*(k mod 3) in
 list_of_zone (g) (g+2) (h) (h+2) m ;;
```

Question 8 • Le plus simple est de trier la liste ℓ et de la comparer à la liste ≪canonique≫ :

```
let bonne_liste l = sort__sort (prefix <) l = intervalle 1 9 ;;
```

Question 9 • Il suffit de vérifier que chacune des trois listes associées respectivement à la ligne, à la colonne, et au bloc de la case visée, est correcte.

```
let bon_sudoku m = let l08 = intervalle 0 8 in
 let lignes = map (fun i -> list_of_bande i 0 8 m) l08
 and colonnes = map (fun j -> list_of_zone 0 8 j j m) l08
 and blocs = map (fun k -> list_of_bloc k m) l08 in
 for_all bonne_liste lignes && for_all bonne_liste colonnes
 && for_all bonne_liste blocs ;;
```

Personellement, j'aime beaucoup la variante suivante, qui n'est pas plus courte, mais est certainement plus mystérieuse . . .

```
let f = it_list (prefix &) true ;;

let bon_sudoku m = let l08 = intervalle 0 8 in
 let lignes = map (fun i -> list_of_bande i 0 8 m) l08
 and colonnes = map (fun j -> list_of_zone 0 8 j j m) l08
 and blocs = map (fun k -> list_of_bloc k m) l08 in
 f(map f (map (map bonne_liste) [lignes;colonnes;blocs])) ;;
```

Celle-ci est plus courte et plus claire, ce qui prouve que l'on peut avoir le beurre *et* l'argent du beurre :

```
let bon_sudoku m =
 let f1 = fun i -> list_of_bande i 0 8 m
 and f2 = fun j -> list_of_zone 0 8 j j m
 and f3 = fun k -> list_of_bloc k m in
 for_all bonne_liste (flat(map (fun f -> map f l08) [f1;f2;f3])) ;;
```

▶ Nous dirons que deux cases s et t sont voisines si elles sont situées sur une même ligne, ou sur une même colonne, ou dans un même bloc (le «ou» n'étant pas exclusif).

Question 10 • Numérotons les cases vides de 0 à $n-1$. Les nœuds situés à la profondeur k sont associés à la case vide numéro k. Un nœud du niveau k possède au plus neuf fils, associés aux neuf choix possibles de la couleur de la case $k+1$. Chaque feuille correspond à une solution. Un simple parcours en profondeur permet de découvrir ces solutions. Bien entendu, le problème est que cet arbre est gigantesque ...

Question 11 • Considérons le graphe suivant : ses sommets sont les 81 cases ; il existe une arête reliant s et t ssi s et t sont voisines. Il s'agit de colorier les sommets, en utilisant les couleurs 1 à 9, étant entendu que les couleurs de certaines cases sont imposées. Ce graphe possède 810 arêtes.

FIN