Option Informatique en Spé MP et MP*
Sous-mots, mélange de mots, le théorème de Higman : le corrigé

1 Sous-mots : définition, quelques propriétés

Question 1 • On construit l'intervalle discret $[\![0, n-1]\!]$, où n est la longueur de la chaîne ; on transforme ensuite cette liste de positions en la liste des caractères de la chaîne.

```
let list_of_string u =
 map (fun i -> u.[i]) (intervalle 0 (string_length u - 1)) ;;
```

Question 2 • On transforme la liste de caractères en liste de chaînes, puis on concatène celles-ci.

```
let string_of_list l =
 concat (map (fun c -> make_string 1 c) l);;
```

Question 3 • Il existe quatre applications qui conviennent. Le tableau ci-dessous les énumère, en indiquant pour chacune d'elles en **gras** les lettres extraites du mot *ababb* pour obtenir le mot *abb*.

$1 \to 1,\ 2 \to 2,\ 3 \to 4$	**ab**a**b**b
$1 \to 1,\ 2 \to 2,\ 3 \to 5$	**ab**ab**b**
$1 \to 1,\ 2 \to 4,\ 3 \to 5$	**a**ba**bb**
$1 \to 3,\ 2 \to 4,\ 3 \to 5$	ab**abb**

Question 4 • $\mathrm{SM}(u)$ est un langage fini : c'est un ensemble de mots de longueur au plus $|u|$.

Question 5 • au est un sous-mot de av ssi u est sous-mot de v ; et, si $a \neq b$, au est sous-mot de bv ssi au est sous-mot de v. Les conventions choisies (ε est sous-mot de tout mot v, et le seul sous-mot de ε est ε) assurent la terminaison de cet algorithme. La programmation est à peu près immédiate :

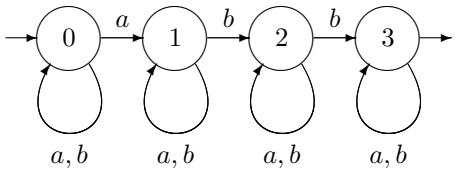
```
let est_sous_mot u v =
 let rec aux = function
  | ([],_) -> true
  | (_,[]) -> false
  | (t::q,t'::q') when t = t' -> aux (q,q')
  | (x,_::q') -> aux (x,q')
 in aux (list_of_string u,list_of_string v);;
```

Question 6 • La réflexivité se montre en prenant pour s l'identité de $[\![1, n]\!]$; la transitivité en notant que la composée de deux injections croissantes est elle-même une injection croissante. Enfin, pour l'antisymétrie, il suffit de noter que, si $u \prec v$ et $v \prec u$, alors $p \leqslant n$ (sinon il ne peut exister d'injection de $[\![1, p]\!]$ dans $[\![1, n]\!]$) et par raison de symétrie $n \leqslant p$; puis, nos deux injections croissantes ayant pour composée l'identité, chacune est l'identité, donc $u = v$.

• L'ordre est total ssi $|X| = 1$. En effet, si l'alphabet X se réduit à une lettre a, alors $u = a^p \prec v = a^q$ ssi $p \leqslant q$. Si X contient au moins deux lettres distinctes a et b, les mots a et b ne sont pas comparables.

Question 7 • On a clairement $L_u = X^* u_1 X^* u_2 \dots X^* u_n X^*$.

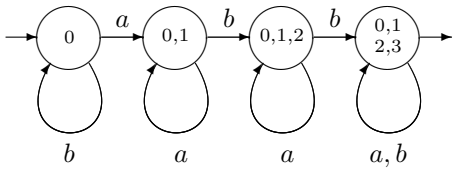
Question 8 • L'automate non-déterministe ci-dessous répond à la question.



On le déterminise avec le procédé usuel, résumé par la table ci-contre :

	$\{0\}$	$\{0,1\}$	$\{0,1,2\}$	$\{0,1,2,3\}$
a	$\{0,1\}$	$\{0,1\}$	$\{0,1,2\}$	$\{0,1,2,3\}$
b	$\{0\}$	$\{0,1,2\}$	$\{0,1,2,3\}$	$\{0,1,2,3\}$

Et voici l'automate espéré :



Question 9 • Soit $\mathcal{A}$ un automate fini reconnaissant L. Pour chaque état q de $\mathcal{A}$, et chaque lettre x de X, ajoutons (q, x, q) à la table de transitions de $\mathcal{A}$. L'automate obtenu reconnaît $\widehat{L}$.

Question 10 • Soit $\mathcal{A}$ un automate fini reconnaissant L. Pour toute transition (q, x, q') de $\mathcal{A}$, ajoutons la transition instantanée (q, ε, q') à la table de $\mathcal{A}$. L'automate $\mathcal{A}'$ obtenu reconnaît $\mathrm{SM}(L)$.

Question 11 • $L = \{a^n b^n \mid n \in \mathbb{N}\}$ n'est pas rationnel. Or $\mathrm{SM}(L) = a^*b^*$ est rationnel.

2 Ordre de multiplicité d'un sous-mot

Question 12 • Notons $p = |u|$. Une injection s de $[\![1, 1]\!]$ dans $[\![1, p]\!]$ est parfaitement définie par le choix de $s(1)$. Ici, on impose $u_{s(1)} = a$; il existe donc $|u|_a$ choix possibles, et finalement $\binom{u}{a} = |u|_a$.

Question 13 • Supposons dans un premier temps $a \neq b$; au est sous-mot de bv ssi au est sous-mot de v, ce qui donne $\binom{v}{au}$ possibilités. Sinon, au est sous-mot de va ssi ou bien au est sous-mot de v, ce qui donne $\binom{v}{au}$ possibilités, ou bien u est sous-mot de v, ce qui donne $\binom{v}{u}$ possibilités. On peut résumer ceci par la formule unique : $\binom{bv}{au} = \binom{v}{au} + \delta_{a,b}\binom{v}{u}$.

Question 14 • Une remarque pour alléger les calculs : $\binom{u}{u} = 1$. $\binom{ababb}{abb} = \binom{babb}{abb} + \binom{babb}{bb}$. Mais $\binom{babb}{abb} = \binom{abb}{abb} = 1$. Par ailleurs $\binom{babb}{bb} = \binom{abb}{bb} + \binom{abb}{b}$, $\binom{abb}{bb} = \binom{bb}{bb} = 1$, et $\binom{abb}{b} = \binom{bb}{b} = \binom{bb}{\varepsilon} + \binom{b}{b} = 2$; donc $\binom{babb}{bb} = 3$ et finalement $\binom{ababb}{abb} = 4$, ce qui confirme le calcul effectué à la question 3.

Question 15 • Il suffit d'appliquer la formule établie à la question 13 :

```
let rec omul = function
 | ([],_) -> 1
 | (_,[]) -> 0
 | (a::u,b::v) when a = b -> omul(u,v) + omul(a::u,v)
 | (a::u,b::v) -> omul(a::u,v);;
```

Question 16 • Si $u = a^n$ et $v = a^{2n}$, le nombre d'appels de `omul` majore le coefficient binomial $\binom{2n}{n}$. Donc le coût peut être exponentiel par rapport à la somme des longueurs de u et v.

Question 17 • Notons $\tilde{u}$ le mot miroir du mot u ; il est clair que $\binom{\tilde{v}}{\tilde{u}} = \binom{v}{u}$. On peut alors déduire de la formule établie à la question 13 la nouvelle formule $\binom{bv}{ua} = \binom{v}{ua} + \delta_{a,b}\binom{v}{u}$. En remplaçant dans cette formule u par $u[1..j]$, v par $v[1..i]$, a par u_{j+1} et b par v_{i+1} il vient $m_{i+1,j+1} = m_{i+1,j} + \delta_{u_{i+1},v_{j+1}} m_{i,j}$. On procède alors au remplissage de la matrice par colonnes successives : on a déjà $m_{i,0} = \delta_{i,0}$ pour tout i. Supposons la colonne j remplie, alors $m_{0,j+1} = 0$ et la formule que l'on vient d'établir permettent de remplir la colonne $j+1$, de haut en bas. Comme le coût du remplissage de chaque cellule est un $\mathcal{O}(1)$, le coût total est un $\mathcal{O}(np)$. La technique utilisée ici est connue sous le nom de *programmation dynamique*.

L'énoncé ne demandait pas de mettre en œuvre cette technique, mais nous allons le faire. Attention à l'indexation dans les chaînes de caractères et dans la matrice.

```
let omul_dyn u v =
 let n = string_length u and p = string_length v in
 let m = make_matrix (n+1) (p+1) 1 in
 for i = 1 to n do
  m.(i).(0) <- 0;
  for j = 1 to p do
   m.(i).(j) <- m.(i).(j-1);
   if u.[i-1] = v.[j-1]
    then m.(i).(j) <- m.(i).(j) + m.(i-1).(j-1)
  done
 done;
 m;;
```

Question 18 • On suit l'algorithme exposé à la question précédente, et on trouve $\binom{ababb}{abb} = m_{3,5} = 4$.

$j \downarrow \ i \rightarrow$	0	1	2	3
0	1	0	0	0
1	1	1	0	0
2	1	1	1	0
3	1	2	1	0
4	1	2	3	1
5	1	2	5	4

3 Mélange de mots

Question 19 • Un élément u de $ab \circ aab$ est un mot de longueur 5 qui commence par a, finit par b, et vérifie $|u|_a = 3$ et $|u|_b = 2$. Ceci ne laisse que trois choix pour $u[2..4]$: aab, aba et baa. On vérifie que chacun de ces choix est effectivement obtenu. Ainsi $ab \circ aab = \{aaabb, aabab, abaab\}$.

Question 20 • Comme la réunion ensembliste est associative, il suffit d'établir $u \circ v = v \circ u$, ce que nous ferons par induction structurelle. Si $u = \varepsilon$ ou $v = \varepsilon$, alors $u \circ v = v \circ u$ résulte de (R1). Sinon, en utilisant successivement (R2), l'hypothèse d'induction structurelle, la commutativité de $\cup$ et à nouveau (R2), il vient :

$$\begin{aligned} ua \circ vb &= (ua \circ v)b \cup (u \circ vb)a = (v \circ ua)b \cup (vb \circ u)a \\ &= (vb \circ u)a \cup (v \circ ua)b = vb \circ ua \end{aligned}$$

Question 21 • Il est clair que, si $P \subset Q$, alors $L \circ P \subset L \circ Q$. Ainsi $L \circ M$ et $L \circ N$ sont tous deux contenus dans $L \circ (M \cup N)$; ainsi $(L \circ M) \cup (L \circ N) \subset L \circ (M \cup N)$.
Réciproquement, soit $x \in L \circ (M \cup N)$: il existe $y \in L$ et $z \in M \cup N$ tels que $x = y \circ z$; alors $x \in L \circ M$ si $z \in M$, et $x \in L \circ N$ si $z \in N$; dans les deux cas, $x \in (L \circ M) \cup (L \circ N)$; ainsi $L \circ (M \cup N) \subset (L \circ M) \cup (L \circ N)$.

Question 22 • Le résultat de la question précédente s'étend sans difficulté à une réunion quelconque :

$$\begin{aligned} L \circ (M \circ N) &= L \circ \Big(\bigcup_{\substack{v \in M \\ w \in N}} v \circ w \Big) = \bigcup_{\substack{v \in M \\ w \in N}} \big(L \circ (v \circ w) \big) \\ &= \bigcup_{\substack{v \in M \\ w \in N}} \Big(\bigcup_{u \in L} u \circ (v \circ w) \Big) = \bigcup_{\substack{u \in L \\ v \in M \\ w \in N}} u \circ (v \circ w) \end{aligned}$$

Il suffit donc d'établir $u \circ (v \circ w) = (u \circ v) \circ w$, ce que nous ferons par induction structurelle. Notons déjà que $\varepsilon \circ L = L$ pour tout langage L, si bien que $\varepsilon \circ (v \circ w) = v \circ w = (\varepsilon \circ v) \circ w$; les cas $v = \varepsilon$ et $w = \varepsilon$ s'établissent de façon analogue. Suit un calcul pénible :

$$\begin{aligned} au \ &\circ \ (bv \circ cw) \\ &= au \circ (b(v \circ cw) \cup c(bv \circ w)) = (au \circ b(v \circ cw)) \cup (au \circ c(bv \circ w)) \\ &= a(u \circ b(v \circ cw)) \cup b(au \circ (v \circ cw)) \cup a(u \circ c(bv \circ w)) \cup c(au \circ (bv \circ w)) \\ &= a((u \circ b(v \circ cw)) \cup (u \circ c(bv \circ w))) \cup b(au \circ (v \circ cw)) \cup c(au \circ (bv \circ w)) \\ &= a(u \circ (b(v \circ cw) \cup c(bv \circ w))) \cup b(au \circ (v \circ cw)) \cup c(au \circ (bv \circ w)) \\ &= a(u \circ (bv \circ cw)) \cup b(au \circ (v \circ cw)) \cup c(au \circ (bv \circ w)) \\ (au \ &\circ \ bv) \circ cw \\ &= (a(u \circ bv) \cup b(au \circ v)) \circ cw = ((a(u \circ bv) \circ cw) \cup (b(au \circ v)) \circ cw) \\ &= a((u \circ bv) \circ cw) \cup c(a(u \circ bv) \circ w) \cup b((au \circ v) \circ cw) \cup c(b(au \circ v)) \circ w) \\ &= a((u \circ bv) \circ cw) \cup b((au \circ v) \circ cw) \cup c((a(u \circ bv) \circ w) \cup (b(au \circ v)) \circ w)) \\ &= a((u \circ bv) \circ cw) \cup b((au \circ v) \circ cw) \cup c((a(u \circ bv) \cup b(au \circ v)) \circ w) \\ &= a((u \circ bv) \circ cw) \cup b((au \circ v) \circ cw) \cup c((au \circ bv) \circ w) \end{aligned}$$

L'hypothèse d'induction structurelle permet alors de conclure.

Question 23 • Il suffit de suivre la définition par induction structurelle ; ce n'est peut-être pas la méthode la plus efficace, mais c'est certainement la plus rapide à traduire en Caml. `map_cons x y` place x en tête de chacune des listes membres de la liste y ; son type est `'a -> 'a list list -> 'a list list`.

```
let map_cons x y = map (fun z -> x::z) y;;
let s_o_c = string_of_list
and c_o_s = list_of_string;;

let melange_mots x y =
 let rec mm = function
  | ([],v) -> [v]
  | (u,[]) -> [u]
  | (a::u,b::v) -> let m1 = mm(a::u,v) and m2 = mm(u,b::v) in
    union (map_cons b m1) (map_cons a m2)
 in map s_o_c (mm (c_o_s x,c_o_s y));;
```

Question 24 • Définissons d'abord une fonction

```
flat_union : 'a list list -> 'a list
```

qui construit la liste réunion des membres de ses membres (*sic*) :

```
let flat_union y = it_list union [] y;;
```

Tout repose ensuite sur une utilisation adroite de `map` et de `flat_union`. On écrit d'abord une fonction `melange_mot_langage` qui réalise le mélange d'un langage réduit à un mot et d'un langage quelconque, au moyen de la formule $u \circ M = \bigcup_{v \in M} u \circ v$:

```
let melange_mot_langage u mang =
 flat_union (map (melange_mots u) mang);;
```

Maintenant, il suffit d'appliquer la formule $L \circ M = \bigcup_{u \in L} u \circ M$:

```
let melange_langages lang mang =
 flat_union (map (fun u -> melange_mot_langage u mang) lang);;
```

4 Le théorème de Higman

Question 25 • Sur un alphabet réduit à une seule lettre, une antichaîne ne peut contenir plus d'un mot !

Question 26 • Soit $n \in \mathbb{N}$; X^n est une antichaîne de cardinal 2^n.

Question 27 • La réponse a déjà été donnée à la question 25 : il ne peut exister d'antichaîne infinie sur un alphabet à une seule lettre !

Question 28 • S'il existait un mot $u \in L$ de longueur 1, on obtiendrait (en enlevant le mot u de L et la lettre u de X) une antichaîne infinie sur un alphabet de cardinal $q-1$, contredisant la minimalité de q.

Question 29 • $M = (L \setminus L') \cup \{u[1..n-1]\}$ est également une antichaîne, sur le même alphabet de cardinal q ; comme $\min_{v \in M} |v| = n-1 < n$, l'hypothèse faite sur n implique que M n'est pas une antichaîne infinie.

Question 30 • Notons ℓ la longueur de z_i. Notons $p_{i,1}$ la position de la première occurrence de u_1 dans z_i. Puis, pour $j \in [\![2, n-1]\!]$, notons $p_{i,j}$ la position de la première occurrence de u_j dans $z_i[p_{i,j-1}+1..\ell]$. On définit ensuite $z_{i,1} = z_i[1..p_{i,1}-1]$; puis, pour $j \in [\![2, n-1]\!]$, $z_{i,j} = z_i[p_{i,j-1}+1..p_{i,j}-1]$; enfin, $z_{i,n} = z_i[p_{i,n-1}+1..\ell]$. Ainsi, pour $j \in [\![1, n-1]\!]$ il n'y aura aucune occurrence de u_j dans le mot $z_{i,j}$.

Question 31 • S'il existait une occurrence de u_n dans $z_{i,n}$, on pourrait écrire $z_{i,n} = wu_nw'$ puis $z_i = z_{i,1}u_1 \dots z_{i,n-1}u_{n-1}wu_nw'$: u serait donc sous-mot de z_i, ce qui contredirait le fait qu'ils sont tous deux membres d'une même antichaîne.

Question 32 • Les éléments de Z_j sont des mots sur l'alphabet $X \setminus \{u_j\}$, lequel est de cardinal $q-1$. Par ailleurs, les éléments maximaux de Z_j (s'il en existe) forment une antichaîne : compte tenu du choix de q, elle ne peut pas être infinie.

Question 33 • Supposons Z_j infini. Soit $s(1)$ un indice à partir duquel il n'y plus aucun élément maximal (un tel indice existe car il n'y a qu'un nombre fini d'éléments maximaux). Il existe un indice $s(2) > s(1)$ tel que $z_{s(1),j} \prec z_{s(2),j}$ puisque $z_{s(1),j}$ n'est pas maximal ; puis, comme $z_{s(2),j}$ n'est pas maximal, il existe un indice $s(3) > s(2)$ tel que $z_{s(2),j} \prec z_{s(3),j}$. On peut poursuivre indéfiniment l'extraction, pour exhiber une suite $(z_{s(i),j})_{i\geqslant 1}$ vérifiant $z_{s(i),j} \prec z_{s(i+1),j}$ pour tout $i \geqslant 1$.

Question 34 • Appliquons successivement ceci pour chaque $j \in [\![1, n]\!]$; comme L' est infini, il est impossible que les Z_j soient tous finis. Après au plus n extractions, on obtient une partie de L' infinie (donc contenant au moins deux éléments) composée de mots deux à deux comparables : or L' est censée être une antichaîne.

Question 35 • Notons $L_\downarrow$ l'ensemble des éléments minimaux de L. Cet ensemble n'est certainement pas vide, il contient par exemple les éléments de longueur minimale de L. Si v admet $u \in L$ comme sous-mot, alors il admet aussi comme sous-mot un élément de $L_\downarrow$. Donc $\widehat{L} = \bigcup_{w\in L_\downarrow} L_w$. Mais les éléments de $L_\downarrow$ sont deux à deux incomparables pour $\prec$, donc forment une antichaîne. Le théorème de HIGMAN permet d'affirmer que $L_\downarrow$ est un ensemble fini. Par ailleurs, chaque L_w est rationnel (cf. question 7), donc $\widehat{L}$ est rationnel en tant que réunion d'une famille *finie* de rationnels.

Question 36 • Notons $K = X^* \setminus \mathrm{SM}(L)$, nous allons montrer que $K = \widehat{K}$. D'après la question précédente, K sera rationnel, et donc son complémentaire $\mathrm{SM}(L)$ le sera aussi.

• L'inclusion $K \subset \widehat{K}$ est banale. Réciproquement, soit $v \in \widehat{K}$. v possède un sous-mot $u \in K$; $u \notin \mathrm{SM}(L)$, donc $v \notin \mathrm{SM}(L)$, soit $v \in K$. *Remarque* : les résultats des questions 10 et 11 apparaissent désormais comme des banalités.

Références bibliographiques

▶ Ce sujet a bénéficié de nombreuses corrections et améliorations suggérées par Nicolas PUECH.

▶ L'article *Ordering by divisibility in abstract algebras* de G. H. HIGMAN a été publié dans *Proceedings of the London Mathematical Society*, 3, (1952) 326-336.

▶ Pour la preuve du théorème de HIGMAN, j'ai repris les idées exposées par Alexandru MATEESCU et Arto SALOMAA dans *Formal Languages : an Introduction and a Synopsis*, chapitre 1, volume 1 du *Handbook of Formal Languages*, (éd. Springer, 1997).

▶ On trouvera un chapitre complet sur les sous-mots dans le livre de M. LOTHAIRE : *Combinatorics on Words* (éd. Cambridge University Press, 1983 et 1997). Les parties 2 et 3 s'en inspirent largement.

▶ Ce sujet a été soumis à la sagacité des étudiants le mardi 6 février 1999.

Le devoir donné en novembre 1997 sur le *plus long sous-mot commun* comporte une première partie dans laquelle on étudie les sous-mots, avec un point de vue notablement différent de celui adopté ici.

FIN