# Interrogation du Lundi 9 Décembre

## Exercice 1 (Opérations sur les langages)

Soient $L_1$, $L_2$ et $L_3$ trois langages. Montrer les égalités suivantes :

1. $(L_1 \cup L_2).L_3 = (L_1.L_3) \cup (L_2.L_3)$
2. $(L_2.L_1)^*.L_2 = L_2.(L_1.L_2)^*$
3. $(L_1 \cup L_2)^* = (L_1^*.L_2^*)^*$

---

## Exercice 2 (Expressions et langages rationnels)

Soit $\Sigma = \{a, b\}$ un alphabet.

1. Donner une description en français des langages dénotés par les expressions rationnelles suivantes :
   (a) $(\varepsilon + \Sigma)(\varepsilon + \Sigma)$ ;
   (b) $(\Sigma^2)^*$ ;
   (c) $(b + ab)^*(a + \varepsilon)$ ;
   (d) $(ab^*a + b)^*$.
2. Donner des expressions rationnelles qui dénotent les langages suivants :
   (a) Les mots sur $\Sigma$ qui contiennent au moins un $a$ ;
   (b) Les mots sur $\Sigma$ qui contiennent au plus un $a$ ;
   (c) Les mots sur $\Sigma$ tels que toute série de $a$ soit de longueur paire ;
   (d) Les mots sur $\Sigma$ dont la longueur n'est pas divisible par 3 ;
   (e) Les mots sur $\Sigma$ tels que deux lettres consécutives soient toujours distinctes.
3. (a) Montrer que l'intersection des deux langages dénotés respectivement par $(b^*a^2b^*)^*$ et $(a^*b^2a^*)^*$ est rationnel.
   (b) Montrer que le complémentaire du langage dénoté par $(a + b)^*b$ est rationnel.

---

## Exercice 3 (Langages locaux)

1. Rappeler la définition d'un langage local.
2. Déterminer si les langages suivants sont des langages locaux. Justifier.
   (a) Le langage $L_1$ dénoté par $(ab)^* + c^*$ sur l'alphabet $\Sigma = \{a, b, c\}$.
   (b) Le langage $L_2$ dénoté par $(ab)^* + (ba)^*$ sur l'alphabet $\Sigma = \{a, b\}$.
3. Soit $L$ un langage sur un alphabet $\Sigma$.
   (a) Montrer que si $L$ est local, alors :

$$\forall u, v, u', v' \in \Sigma^*, \quad \forall a \in \Sigma, \quad (uav \in L \text{ et } u'av' \in L) \Rightarrow uav' \in L.$$

   (b) Démontrer la réciproque de cette propriété.

---

## Exercice 4 (Mots bien parenthésés ou mots de Dyck)

$\Sigma$ désigne l'alphabet $\{\,(\,,\,)\,\}$.

Définissons récursivement $\mathcal{M}$ l'ensemble des mots sur l'alphabet $\Sigma$ de la manière suivante :

— $\epsilon \in \mathcal{M}$ ($\epsilon$ désigne le mot vide) et $(\forall x \in \Sigma)(x \in \mathcal{M})$,

— $(\forall x \in \Sigma)(\forall m \in \mathcal{M})(xm \in \mathcal{M})$.

Définissons récursivement $\mathcal{M}^*$ l'ensemble des **mots bien parenthésés** sur l'alphabet $\Sigma$ de la manière suivante :

— $\epsilon \in \mathcal{M}^*$ ($\epsilon$ désigne le mot vide)

— $(\forall m \in \mathcal{M}^*)((m) \in \mathcal{M}^*)$,

— $(\forall m_1 \in \mathcal{M}^*)(\forall m_2 \in \mathcal{M}^*)(m_1 m_2 \in \mathcal{M}^*)$.

Un mot $m$ sera défini en OCaml comme un tableau de caractères :

$$let\ m = [|`(`;`(`;`)`;`(`;`)`;`)`;`)`;`(`;`(`;`)`|];;$$

1. Condition nécessaire.
   Ecrire en OCaml, une fonction *parentheses* de signature : *char array* $\rightarrow$ *bool* qui permet de savoir si un mot $m$ contient autant de parenthèses ouvrantes que de parenthèses fermantes.
   Cette information, sur le nombre de parenthèses ouvrantes et le nombre de parenthèses fermantes d'un mot $m$, suffit-elle pour savoir si le mot $m$ est bien parenthésé ? Justifier votre réponse.
2. Condition nécessaire et suffisante : profil d'un mot.
   Etant donné un mot $m$, notons $l(m)$ sa longueur, définissons alors le **profil** de $m$ comme l'application $p_m$ définie sur $[\![0\,,\,l(m)]\!]$ à valeurs dans $\mathbb{Z}$ par :
   $$\left\{\begin{array}{l} p_m(0) = 0 \\ \forall i \in [\![1\,,\,l(m)]\!] \mapsto p_m(i) = p_m(i-1) + \left\{\begin{array}{l} +1 \text{ si } m.(i-1) = `(` \\ -1 \text{ si } m.(i-1) = `)` \end{array}\right. \end{array}\right. .$$
   (a) Calculer le profil du mot $(\,(\,)\,(\,)\,)\,)\,(\,(\,)$. Ce mot est-il bien parenthésé ?
   (b) Ecrire en OCaml, une fonction *profil* de signature : *char array* $\rightarrow$ *int array* qui calcule le profil d'un mot $m$.
   (c) Soit $m$ un mot différent du mot vide tel que : $\left\{\begin{array}{l} p_m(l(m)) = 0 \\ p_m \text{ est à valeurs dans } \mathbb{N} \end{array}\right.$ .
   — Justifier l'existence d'un entier $k$ tel que :
   $$k = \min\{i \in [\![1\,,\,l(m)]\!] / p_m(i) = 0\}.$$
   — On suppose que $k = l(m)$.
   Montrer qu'il existe un mot $m'$ de longueur $l(m) - 2$ tel que : $m = (m')$.
   Définir le profil de $m'$ en fonction du profil de $m$ et montrer qu'il vérifie :
   $$\left\{\begin{array}{l} p_{m'}(l(m')) = 0 \\ p_{m'} \text{ est à valeurs dans } \mathbb{N} \end{array}\right. .$$
   — On suppose que : $k < l(m)$ .
   Montrer qu'il existe deux mots $m_1$ et $m_2$ de longueur strictement inférieure à $l(m)$ tels que : $m = m_1 m_2$ .
   Définir le profil de $m_1$ et le profil de $m_2$ en fonction du profil de $m$ et montrer qu'ils vérifient :
   $$\left\{\begin{array}{l} p_{m_1}(l(m_1)) = 0 \\ p_{m_1} \text{ est à valeurs dans } \mathbb{N} \end{array}\right. \text{ et } \left\{\begin{array}{l} p_{m_2}(l(m_2)) = 0 \\ p_{m_2} \text{ est à valeurs dans } \mathbb{N} \end{array}\right. .$$

— Montrer que $m$ est bien parenthésé.

(d) Soit $m$ un mot bien parenthésé.

Montrer que le profil de $m$ vérifie : $\begin{cases} p_m(l(m)) = 0 \\ p_m \text{ est à valeurs dans } \mathbb{N} \end{cases}$.

(e) En déduire une condition nécessaire et suffisante pour qu'un mot soit bien parenthésé.

(f) Ecrire en OCaml, une fonction *bien_parenthese* de signature : *char array* $\rightarrow$ *bool* qui permet de savoir si un mot $m$ est bien parenthésé.