# TELECOM Nancy 1A — Mathématiques Appliquées pour l'Informatique

Théorie des langages : généralités, opérations sur les langages,
langages réguliers, expressions régulières, codes

**Exercice 1**

Soient $A$ un alphabet et $L$, $L_1$, $L_2$, $L_3$ et $L_4$ cinq langages sur $A$. Montrer que :

1. si $L_1 \subset L_2$ et $L_3 \subset L_4$ alors $L_1.L_3 \subset L_2.L_4$
2. si $L_1 \subset L_2$ alors $L_1^* \subset L_2^*$
3. $(L^*)^* = L^*$
4. $L.(L_1 \cup L_2) = L.L_1 \cup L.L_2$
5. $L.(\bigcup_{i \geq 0} L_i) = \bigcup_{i \geq 0} L.L_i$ où les $L_i$ sont des langages ($i \in \mathbb{N}$).
6. $L^+ = L.L^* = L^*.L$
7. $L.L^* \cup \{\varepsilon\} = L^*$

**Exercice 2**

Soient $A$ un alphabet et $L$, $L_1$, $L_2$, $L_3$ quatre langages sur $A$. Soient les propriétés suivantes dire si elles sont vraies ou fausses en justifiant vos réponses.

1. $L \cup \emptyset = \emptyset \cup L = L$
2. $L_1 \cup L_2 = L_2 \cup L_1$
3. $(L_1 \cup L_2) \cup L_3 = L_1 \cup (L_2 \cup L_3)$
4. $L_1.L_2 = L_2.L_1$
5. $L \cup A^* = A^* \cup L = A^*$
6. $(L_1.L_2).L_3 = L_1.(L_2.L_3)$
7. $\{\varepsilon\}.L = L.\{\varepsilon\} = L$
8. $\emptyset.L = L.\emptyset = \emptyset$

**Exercice 3**

Soit $A = \{a, b\}$ un alphabet. Ecrire des expressions régulières (rationnelles) qui dénotent les langages suivants :

1. $L_1 =$ l'ensemble des mots de $A^*$ se terminant par $a$
2. $L_2 =$ l'ensemble des mots de $A^*$ contenant au moins un $b$
3. $L_3 =$ l'ensemble des mots de $A^*$ contenant au plus un $b$
4. $L_4 =$ l'ensemble des mots de $A^*$ contenant un nombre pair de $a$
5. $L_4 =$ l'ensemble des mots de $A^*$ contenant autant de $a$ que de $b$

Avez-vous rencontré des difficultés pour décrire un de ces langages ? Qu'en conclure ?

**Exercice 4**

Soit $A = \{a, b\}$ un alphabet. Décrire les langages dénotés par les expressions régulières (rationnelles) suivantes :

1. $e_1 = (a + b)^*$
2. $e_2 = (a^*b^*)^*$
3. $e_3 = a^*ba^*ba^*ba^*$
4. $e_4 = (a + ba + bbaa)^*(b + \epsilon)$

**Exercice 5**

On rappelle qu'un langage non vide $C \subset A^+$ est un code si tout mot de $C^*$ se décompose de manière unique comme produit (concaténation) de mots de $C$. On a les propriétés suivantes :

1. Tout sous-ensemble d'un code est un code.
2. Si $C$ est un code, $\varepsilon \notin C$.
3. Si $C$ est un code, $C$ et $\bigcup_{n \geq 2} C^n$ sont deux ensembles disjoints.
4. Si tous les mots de $C$ sont de même longueur non nulle, alors $C$ est un code. On parle de code uniforme.
5. Si aucun mot de $C$ n'est préfixe (resp. suffixe) d'autre mot de $C$ alors $C$ est un code. Dans ce cas $C$ est qualifié de code préfixe (resp. suffixe).

Lorsque ces propriétés ne sont pas suffisantes pour montrer qu'un langage $L$ est ou n'est pas un code, on a recours à l'algorithme de Sardinas Patterson suivant :

1. Initialisation : $U_0 = L^{-1}.L \setminus \{\varepsilon\}$
2. Itération : $U_{n+1} = U_n^{-1}.L \cup L^{-1}.U_n$
3. Condition d'arrêt : $\left\{ \begin{array}{lll} \varepsilon \in U_n & \Rightarrow & L\ n'est\ pas\ un\ code \\ \exists (i,j) \in \mathbb{N}^2\ i \neq j\ et\ U_i = U_j & \Rightarrow & L\ est\ un\ code \end{array} \right.$

Parmi les langages suivants déterminer ceux qui sont des codes [1].

1. $L_1 = \{a, b\}$
2. $L_2 = ab^*$
3. $L_3 = \{a, ab, ba\}$
4. $L_4 = \{a, ba, bb\}$
5. $L_5 = \{aa, baa, ba\}$
6. $L_6 = \{a, abbba, babab, bb\}$
7. $L_7 = a^+b^+$
8. $L_8 = a^+b^*$
9. $L_9 = \{a^n b^n, n \in \mathbb{N}^*\}$
10. $L_{10} = (ab)^*$

**Exercice 6**

Soit $A = \{0, 1\}$ un alphabet et $C$ un sous-ensemble fini de $A^n$.

1. Soient $\alpha, \beta \in A^n$, on pose $d_H(\alpha, \beta) =$ le nombre de rangs entre 1 et $n$ pour lesquels les deux mots $\alpha$ et $\beta$ diffèrent. Donner une définition plus formelle de cette quantité appelée distance de Hamming de $\alpha$ à $\beta$. Vérifier que $d_H$ a bien les propriétés d'une distance.
2. On pose $H_C = inf\{d_H(\alpha, \beta),\ (\alpha, \beta) \in C \times C\ et\ \alpha \neq \beta\}$. Cette quantité s'appelle constante de Hamming de $C$. Quelle utilité peut avoir cette constante ?

1. On demande une démonstration pour chaque langage. Il est recommandé d'utiliser prioritairement les propriétés.