Théorie des langages : Généralités

1 Alphabets

Définition : (Alphabet/Vocabulaire)
On appelle alphabet ou vocabulaire tout ensemble fini.

Définition : (Lettres)
Les éléments d'un alphabet sont appelés des lettres

2 Mots

Définition : (Mot)
Soit A un alphabet,
Un mot sur A est une suite finie à valeurs dans A.
L'ensemble des mots de A est noté A^*

Remarques :
- Soit α un mot sur A, $\alpha : [1,n] \to A$
- n s'appelle la longueur du mot, on note $|\alpha| = n$
- Pour $i \in [1,n], \alpha(i)$ ou α_i désigne la ième lettre de α
- Si $n = 0$, α est le mot vide, noté ε (ou $\wedge$)

Définition : (Egalité de 2 mots)
Deux mots α et β sont égaux si et seulement si :

$$\begin{cases} |\alpha| = |\beta| = n \\ \forall i \in [1,n], \alpha_i = \beta_i \end{cases}$$

Notation :
Soit A un alphabet, $a \in A$ et $\alpha \in A^*$, on note $|\alpha|_a$ le nombre d'occurences de la lettre a dans le mot α.

Définition : (Concaténation de 2 mots)
Soit A un alphabet,
La concaténation est une opération définie sur A^* de la façon suivante :

$$\begin{array}{cccc} . & : & A^* \text{ x } A^* & \to & A^* \\ & & (\alpha,\beta) & \mapsto & \alpha.\beta \end{array}$$

Où $\alpha = \alpha_1\alpha_2...\alpha_p, \beta = \beta_1\beta_2...\beta_q$ et $\alpha.\beta = \alpha_1\alpha_2...\alpha_p\beta_1\beta_2...\beta_q$

Proposition :
L'opération de concaténation possède les propriétés suivantes :
- $\forall \alpha, \beta \in A^*, |\alpha + \beta| = |\alpha| + |\beta|$
- $\forall \alpha, \beta, \gamma \in A^*, (\alpha.\beta).(\gamma) = \alpha.(\beta.\gamma)$ (Associativité)
- $\forall \alpha \in A^*, \alpha.\varepsilon = \varepsilon.\alpha = \alpha$ (ε est l'élément neutre de la concaténation)
- $\forall \alpha \in A^*, \alpha.\alpha = \alpha \Leftrightarrow \alpha = \varepsilon$ (ε est le seul mot idempotent)

Définition (Facteur, facteur propre)
Soient A un alphabet, $\alpha, \beta \in A^*$,
On dit que α est facteur de β si :

$$\exists \gamma, \delta \in A^*, \beta = \gamma\alpha\delta$$

Si de plus, $\alpha \neq \beta$ et $\alpha \neq \varepsilon$, alors α est dit facteur propre de β

Définition : (Préfixe, préfixe propre)
Soient A un alphabet, $\alpha,\beta \in A^*$.
On dit que α est préfixe de β et l'on note $\alpha \sqsubseteq \beta$ si :

$$\exists \delta \in A^*, \beta = \alpha.\delta$$

Si $\alpha \neq \beta$ et $\alpha \neq \varepsilon$, alors α est dit préfixe propre de β et l'on note alors $\alpha \sqsubset \beta$

Remarque : La relation $\sqsubseteq$ est une relation d'ordre

Définition : (Suffixe, suffixe propre)
Soient A un alphabet, $\alpha, \beta \in A^*$.
On dit que α est suffixe de β si :

$$\exists \gamma \in A^*, \beta = \gamma.\alpha$$

Si $\alpha \neq \beta$ et $\alpha \neq \varepsilon$, alors α est dit suffixe propre de β

3 Langages

Définition : (Langage)
Soit A un alphabet,
On appelle langage sur A toute partie L de A^*. L'ensemble des langages sur A est donc $\mathcal{P}(A) = \{L \mid L \subset A^*\}$ Un langage sur un alphabet est donc un ensemble de mots sur cet alphabet.

3.1 Opéations ensemblistes sur les langages

Définition : (Union de 2 langages)
Soient L_1 et L_2 deux langages sur A
On appelle union de L_1 et L_2 et l'on note $L_1 \cup L_2$ (ou $L_1 + L_2$) le langage défini par :

$$L_1 \cup L_2 = L_1 + L_2 = \{\alpha \in A^* \mid \alpha \in L_1 \text{ OU } \alpha \in L_2\}$$

Définition : (Intersection de 2 langages)
Soient L_1 et L_2 deux langages sur A.
On appelle intersection de L_1 et L_2 et l'on note $L_1 \cap L_2$ le langage défini par :

$$L_1 \cap L_2 = \{\alpha \in A^* \mid \alpha \in L_1 \text{ ET } \alpha \in L_2\}$$

Définition : (Complémentaire d'un langage)
Soit L un langage sur A.
On appelle complémentaire de L et l'on note $\overline{L}$ le langage défini par :

$$\overline{L} = A^* \backslash L = \{\alpha \in A^* \mid \alpha \notin L\}$$

Définition : (Différence symétrique)
Soient L_1 et L_2 deux langages sur A.
On appelle différence symétrique (ou réunion disjointe) entre L_1 et L_2 et l'on note $L_1 \Delta L_2$ le langage défini par :

$$L_1 \Delta L_2 = \{\alpha \in A^* \mid \alpha \in L_1 \oplus \alpha \in L_2\} = L_1 \cup L_2 \backslash L_1 \cap L_2$$

On s'autorise également l'utilisation des autres opérations ensemblistes (produit cartésien, différence, ...) même si l'on n'en rappelle pas les définitions.

3.2 Produit de langages

Définition : (Produit de 2 langages)
Soient L_1 et L_2 deux langages sur A.
On appelle produit (ou concaténation) de L_1 et L_2 et l'on note $L_1.L_2$ le langage défini par :

$$L_1.L_2 = \{\gamma \in A^* \mid \exists (\alpha, \beta) \in L_1 \text{x} L_2, \gamma = \alpha.\beta\}$$

Définition : Puissance nième
Soit L un langage sur A.
On appelle puissance nième de L et l'on note L^n le langage défini par :

$$\begin{cases} L^0 = \{\varepsilon\} \\ L^1 = L \\ L^n = L.L^{n-1} \quad \text{si} \quad n > 1 \end{cases}$$

3.3 Itéré d'un langage

Définition : Itéré d'un langage

Soit L un langage sur A.
On appelle fermeture itérative (ou étoile ou fermeture de Kleene) de L et l'on note L^* le langage défini par :

$$L^* = \bigcup_{n\geq 0} L^n = \sum_{n\geq 0} L^n$$

Il s'agit du plus petit langage sur A contenant L, le mot vide et stable par concaténation.

Définition : (Itéré strict)

Soit L un langage sur A.
On appelle étoile stricte (ou itéré strict) de L et l'on note L^+ le langage défini par :

$$L^+ = \bigcup_{n>0} L^n = \sum_{n>0} L^n$$

Il s'agit du plus petit langage sur A contenant L, et stable par concaténation.

Remarques :
— $L^* = L^+ \cup \{\varepsilon\}$
— $L^+ = L.L^*$
— $L^* = L^+ \Leftrightarrow \varepsilon \in L$

3.4 Langage mirroir

Définition : (Langage mirroir)

Soit L un langage sur A.
On appelle mirroir (ou réfléchi) de L et l'on note $\tilde{L}$ le langage défini par :

$$\tilde{L} = \{\phi(\alpha) \mid \alpha \in L\}$$

où $\phi : A^* \to A^*$ est l'application définie par :

$$\begin{cases} \phi(\varepsilon) = \varepsilon \\ \forall \alpha \in A^*, a \in A, \phi(a.\alpha) = \phi(\alpha).a \end{cases}$$

Remarque :
ϕ est la fonction qui renverse un mot :

$$\phi(\alpha_1\alpha_2...\alpha_n) = \alpha_n...\alpha_2\alpha_1$$

3.5 Quotients

Définition : (Quotient gauche)

Soient L_1 et L_2 deux langages sur A.
On appelle quotient gauche de L_2 par L_1 et l'on note $L_1^{-1}.L_2$ le langage défini par :

$$L_1^{-1}.L_2 = \{\gamma \in A^*, \exists \alpha \in L_1, \alpha.\gamma \in L_2\}$$

Définition : (Quotient droit)

Soient L_1 et L_2 deux langages sur A.
On appelle quotient droit de L_1 par L_2 et l'on note $L_1.L_2^{-1}$ le langage défini par :

$$L_1.L_2^{-1} = \{\gamma \in A^*, \exists \alpha \in L_2, \gamma.\alpha \in L_1\}$$

4 Proprietés des opérations

Propriétés de l'union et du produit

L'union est associative, commutative, d'élement neutre $\emptyset$ et d'élement absorbant A^*. Autrement dit, $\forall L, L_1, L_2, L_3 \in \mathcal{P}(A^*)$:
— $(L_1 \cup L_2) \cup L_3 = L_1 \cup (L_2 \cup L_3)$

— $L_1 \cup L_2 = L_2 \cup L_1$
— $\emptyset \cup L = L \cup \emptyset = L$
— $L \cup A^* = A^* \cup L = A^*$

Le produit est associatif, d'élement neutre $\{\varepsilon\}$, d'élement absorbant $\emptyset$ et distributif par rapport à l'union. Autrement dit, $\forall L, L_1, L_2, L_3 \in \mathcal{P}(A^*)$:
— $(L_1.L_2).L_3 = L_1.(L_2.L_3)$
— $\{\varepsilon\}.L = L.\{\varepsilon\} = L$
— $\emptyset.L = L.\emptyset = \emptyset$
— $L_1.(L_2 \cup L_3) = L_1.L_2 \cup L_1.L_3$

Remarque :
Le produit n'est pas commutatif en général (sauf si $\text{card}(A) = 1$)

Proprietés de l'étoile et de l'étoile stricte
L'opération de fermeture itérative (ainsi que l'étoile stricte) est idempotente, autrement dit $\forall L \in \mathcal{P}(A^*)$:
— $(L^*)^* = L^*$
— $(L^+)^+ = L^+$

De plus on a $\forall L_1, L_2 \in \mathcal{P}(A), (L_1^* + L_2^*)^* = (L_1 + L_2)^* = (L_1^*.L_2^*)^*$

5 Langages réguliers et expressions régulières

Définition : Ensemble des langages réguliers
L'ensemble $\mathcal{R}at(A^*)$ des langages rationnels (ou réguliers) sur un alphabet A est défini inductivement par :
— La base $B = \{\emptyset, \{\varepsilon\}\} \cup \{\{a\} \mid a \in A\}$
— L'ensemble des opérations $\mathcal{O}p = \{\text{union, produit, itéré}\}$

Remarque :
On définit parfois la base comme étant l'ensemble de tous les langages finis sur A

Définition : Ensemble des expressions régulières
L'ensemble $\mathcal{R}_A$ des expressions régulières (ou rationnelles) sur un alphabet A est défini inductivement de la façon suivante :
— La base $B = \{\emptyset, \varepsilon\} \cup \{\{a\} \mid a \in A\}$
— L'ensemble des opérations $\mathcal{O}p$: $\{\forall e_1 \in \mathcal{R}_A, \forall e_2 \in \mathcal{R}_A, (e_1 + e_2) \in \mathcal{R}_A; (e_1 e_2) \in \mathcal{R}_A; (e_1)^* \in \mathcal{R}_A; (e_1)^+ \in \mathcal{R}_A$

Remarque :
Les expressions régulières forment un langage sur l'alphabet $A \cup \{\emptyset, \varepsilon, +, ^*, (,)\}$

Définition :
On définit une application $\mathcal{L} : \mathcal{R}_A \to \mathcal{R}at(A^*)$ de la manière suivante (Avec $a \in A$ et $e_1, e_2, e \in \mathcal{R}_A$) :
— $\mathcal{L}(\emptyset) = \emptyset$
— $\mathcal{L}(\varepsilon) = \{\varepsilon\}$
— $\mathcal{L}(a) = \{a\}$
— $\mathcal{L}((e_1 + e_2)) = \mathcal{L}(e_1) \cup \mathcal{L}(e_2)$
— $\mathcal{L}((e_1 e_2)) = \mathcal{L}(e_1).\mathcal{L}(e_2)$
— $\mathcal{L}((e)^*) = \mathcal{L}(e)^*$
— $\mathcal{L}((e)^+) = \mathcal{L}(e)^+$

Soient α un langage et e une expression régulière tels que $\mathcal{L}(e) = \alpha$, on dit que e dénote α

Théorème :
Un langage est régulier si et seulement si il est dénoté par une expression régulière.

6 Code

6.1 Définion et propriétés

Définition : Code
Soit C un langage.
On dit que C est un code s'il n'existe pas de mot de C^* ayant 2 factorisation différentes avec des mots de C.

Proposition :
Soit C un sous ensemble fini de A^+.
Si C est un code, alors tout sous ensemble de C est un code.

Propositions / Définitions :
Soit C un sous-ensemble fini de A^* :
— Si C est un code, $\varepsilon \in C$
— Si C est un code, C et $\bigcup_{n \geq 2} C^n$ sont 2 ensembles disjoints

- Si tous les mots de C sont de même longueur non nulle alors C est un code. On parle alors de code **uniforme**
- Si aucun mot de C n'est préfixe (resp. suffixe) d'un autre mot de C, C est un code. Dans ce cas, le code C est qualifié de code **préfixe** (resp. **suffixe**

6.2 Algorithme de Sardinas Patterson

Algorithme

Soit $C \in \mathcal{P}(A^*)$.
On cherche à déterminer si C est un code. L'algorithme de Sardinas-Patterson consiste en la construction d'une suite d'ensembles :

- *Initialisation* : $U_0 = C^{-1}.C \backslash \{\varepsilon\}$
- *Itération* : $U_{n+1} = U_n^{-1}.C \cup C^{-1}.U_n$
- *Condition d'arret* : $\begin{cases} \varepsilon \in U_n \Rightarrow C \text{ n'est pas un code} \\ \exists i \exists j, i > j \ et \ U_i = U_j \Rightarrow C \text{ est un code} \end{cases}$