Langages et expressions régulières

1 Mots

1.1 Alphabets et mots

Définition 1.1. Un alphabet Σ est un ensemble fini de symboles appelés aussi lettres ou caractères.

Définition 1.2. Un mot m sur un alphabet Σ est une suite finie de lettres notée $m_0m_1\dots m_{k-1}$. La longueur d'un mot m noté $|m|$ est son nombre de lettres. Le mot vide noté ε est le mot associé à la suite vide. Il est de longueur 0.

Définition 1.3. Soit m un mot et x une lettre, la longueur en x de m notée $|m|_x$ est le nombre d'occurences de x dans m.

Définition 1.4. Pour $n \in \mathbb{N}$, Σ^n est l'ensemble des mots de longueur n sur l'alphabet Σ.
Σ^* est l'ensemble de tous les mots sur l'alphabet Σ. On a $\Sigma^* = \bigcup_{n\in\mathbb{N}} \Sigma^n$. On note $\Sigma^+ = \bigcup_{n\in\mathbb{N}^*} \Sigma^n$.

1.2 Concaténation

Définition 1.5. La concaténation de deux mots u et v où $u = u_0u_1\dots u_{p-1}$ et $v = v_0v_1\dots v_{q-1}$ est le mot $uv = u_0u_1\dots u_{p-1}v_0v_1\dots v_{q-1}$.

Proposition 1.1. *La concaténation est une loi de composition interne dans Σ^* associative et d'élément neutre ε. On note, pour un mot u, $u^0 = \varepsilon$ et $\forall n \in \mathbb{N},\quad u^{n+1} = uu^n$.*

Proposition 1.2. *Soit x, y, z des mots sur un alphabet Σ et $c \in \Sigma$. Alors,*
(i) $|xy| = |x| + |y|$ (ii) $|xy|_c = |x|_c + |y|_c$ (iii) *si $xy = xz$ ou $yx = zx$, alors $y = z$.*

Proposition 1.3 (Lemme de Levi)**.** *Soit x, y, z et $t \in \Sigma^*$ tels que $xy = zt$. Alors, il existe un unique mot u tel que l'une des deux conditions suivantes est réalisée* :

$$x = zu \text{ et } t = uy \qquad \text{ou} \qquad z = xu \text{ et } y = ut.$$

1.3 Préfixe, suffixe, facteur, sous-mot

Définition 1.6. Soit $x, m \in \Sigma^*$. On dit que :

- x est préfixe de m s'il existe $y \in \Sigma^*$ tel que $m = xy$.
- x est suffixe de m s'il existe $y \in \Sigma^*$ tel que $m = yx$.
- x est facteur de m s'il existe $y, z \in \Sigma^*$ tel que $m = yxz$.

Exercice 1.1.
1. Montrer que la relation $u \leq_p v$: "u préfixe de v" est une relation d'ordre.
2. Soit u, v, w tel que $u \leq_p w$ et $v \leq_p w$, montrer que $u \leq_p v$ ou $v \leq_p u$
3. Montrer que si $u \leq_p v$ alors $wu \leq_p wv$.

Définition 1.7. Le mot $u = u_0u_1\dots u_{k-1}$ est un sous mot du mot v s'il existe des mots $w_0, w_1, w_2, \dots, w_k$ tels que $v = w_0u_0w_1u_1w_2\dots w_{k-1}u_{k-1}w_k$.

Exercice 1.2. Étant donné 2 mots u et v proposer un algorithme efficace pour savoir si :
1. u est un préfixe (respectivement suffixe) de v
2. u est un facteur de v
3. u est un sous-mot de v

2 Langages

Définition 2.1. Un langage sur l'alphabet Σ est une partie de Σ^*.

2.1 Opérations sur les langages

Définition 2.2. Soit L et L' deux langages sur l'alphabet Σ. La concaténation de L et L' notée $L.L'$ est le langage $L.L' = \{uv \mid u \in L \text{ et } v \in L'\}$.

Remarque 2.1. Les opérations ensemblistes usuelles s'appliquent aux langages : union, intersection, complémentaire et différence.

Définition 2.3. On définit $L^0 = \{\varepsilon\}$ et pour $n \in \mathbb{N}$, $L^{n+1} = L.L^n$.

Définition 2.4. Soit L un langage sur l'alphabet Σ. L'étoile de Kleene de L est le langage L^* défini par $L^* = \bigcup_{k\in\mathbb{N}} L^k$.
On notera aussi $L^+ = \bigcup_{k\in\mathbb{N}^*} L^k$.

2.2 Langages réguliers

Définition 2.5. Soit Σ un alphabet. Les langages réguliers sont les langages définis par induction :

- Les langages $\emptyset, \{\varepsilon\}, \{a\}$ pour tout $a \in \Sigma$, sont réguliers,
- La concaténation, l'union et l'étoile de Kleene de langages réguliers sont des langages réguliers.

Définition 2.6. Soit Σ un alphabet. Les expressions régulières sont définies par induction :

- les symboles $\emptyset, \varepsilon, a$ pour $a \in \Sigma$ sont des expressions régulières,
- si e et f sont des expressions régulières, alors ef, $e|f$ et e^* sont des expressions régulières.

Remarque 2.2. On a l'ordre de priorité suivant : l'étoile est prioritaire sur la concaténation qui est prioritaire sur l'union.

Définition 2.7. Le langage dénoté par une expression régulière e notée $\mathcal{L}(e)$ est défini par induction :

- $\mathcal{L}(\emptyset) = \emptyset$; $\mathcal{L}(\varepsilon) = \{\varepsilon\}$; $\mathcal{L}(a) = \{a\}$ pour tout $a \in \Sigma$,
- Soit e, f deux expressions régulières :
  - $\mathcal{L}(e|f) = \mathcal{L}(e)|\mathcal{L}(f)$
  - $\mathcal{L}(ef) = \mathcal{L}(e).\mathcal{L}(f)$
  - $\mathcal{L}(e^*) = \mathcal{L}(e)^*$

Proposition 2.1. *Les langages dénotés par une expression régulière sont exactement les langages réguliers.*

Exercice 2.1. Avec $\Sigma = \{a, b\}$, trouver le langage dénoté par :

- $b^*ab^*a(b|a)^*$
- $(ab|b)^*(a|\epsilon)$
- $(a^*|b^*)^*$

On pourra commencer par tester quels mots de Σ^2 ils contiennent.

2.3 Morphismes

Définition 2.8. Soit Σ_1, Σ_2 deux alphabets et $\varphi : \Sigma_1^* \longrightarrow \Sigma_2^*$ une fonction.
On dit que φ est un morphisme si pour tous mots $u, v \in \Sigma_1^*$, on a $\varphi(uv) = \varphi(u)\varphi(v)$.

Définition 2.9. Soit φ une fonction de Σ_1 vers Σ_2. On peut l'étendre en un morphisme de Σ_1^* vers Σ_2^* noté aussi φ défini par :

$$\varphi(\varepsilon) = \varepsilon \quad \text{et} \quad \forall m \in \Sigma_1^+ \text{ avec } m = m_1 m_2 \dots m_k, \quad \varphi(m) = \varphi(m_1)\varphi(m_2)\dots\varphi(m_k).$$

On a bien : $\forall u, v \in \Sigma_1^+, \quad \varphi(uv) = \varphi(u)\varphi(v)$.

Définition 2.10. Soit $\varphi : \Sigma_1 \longrightarrow \Sigma_2$ et e une expression régulière sur Σ_1. Alors $\varphi(e)$ est l'expression régulière où chaque lettre est remplacée par son image par φ.

Proposition 2.2. *Soit e une expression régulière sur l'alphabet Σ et $\varphi : \Sigma \longrightarrow \Sigma$ alors $\mathcal{L}(\varphi(e)) = \varphi(\mathcal{L}(e))$.*

Corollaire 2.3. *Les langages réguliers sont stables par morphisme.*

3 Expressions régulières étendues

De nombreuses applications de recherche ou d'édition permettent d'utiliser des exepressions régulières comme par exemple la commande grep sous linux.
La norme POSIX propose 2 syntaxes pour les expressions régulières :

- BRE (Basic Regular Expressions)
- ERE (Extended Regular Expressions) qu'on verra en TP.

On y retrouve les opérations usuelles ainsi que de nouveaux constructeurs.