# I Alphabet et mot

**Définition : Alphabet**

Un alphabet est un ensemble $\Sigma$ fini, dont les éléments sont des lettres.

**Définition : Mot**

Un mot $m$ d'un alphabet $\Sigma$ est une suite finie $m_1$, ..., $m_n$ de lettres de $\Sigma$, et on note $m = m_1...m_n$.
$n$ est la taille de $m$, qu'on note $|m|$.
Le mot vide (contenant aucune lettre) est noté $\varepsilon$.

Attention : $\varepsilon$ est un mot, pas une lettre.

On définit aussi :

- $\Sigma^*$ : l'ensemble des mots sur $\Sigma$.
- $\Sigma^+ = \Sigma^* \setminus \{\varepsilon\}$.
- $\Sigma^n$ l'ensemble des mots de longueur $n$ sur $\Sigma$.

**Définition : Égalité de mots**

Deux mots $u = u_1...u_n$ et $v = v_1...v_p$ sur le même alphabet $\Sigma$ sont égaux s'ils ont la même longueur ($n = p$) et si pour tout $i \in \{1, ..., n\}$, $u_i = v_i$.

**Définition : Concaténation et puissance**

La concaténation de deux mots $u = u_1...u_n$ et $v = v_1...v_p$ est :
$$uv = u_1...u_n v_1...v_p$$
Elle est aussi parfois notée $u \cdot v$.
Si $u$ est un mot, on définit $u^0 = \varepsilon$ et $u^k = \underbrace{uu...u}_{k}$.

**Exercice 1.**
Soient $\Sigma$ un alphabet, $a$, $b \in \Sigma$ et $u \in \Sigma^*$. On suppose $au = ub$.
Montrer que $a = b$ et qu'il existe $k \in \mathbb{N}$ tel que $u = a^k$.

**Définition : Préfixe, suffixe, facteur, sous-mot**

- $u$ est un préfixe de $m$ s'il existe un mot $v$ tel que $m = uv$.
- $u$ est un suffixe de $m$ s'il existe un mot $v$ tel que $m = vu$.
- $u$ est un facteur (*substring* en anglais) de $m$ s'il existe des mots $v$, $w$ tels que $m = vuw$.
- $u$ est un sous-mot (*subsequence* en anglais) de $m$ si $u$ est une sous-suite (ou : suite extraite) de $m$.

Exemple : *abc* est un sous-mot de *aabacb*, mais pas un facteur.

## I.1 Implémentation d'un mot avec une chaîne de caractères

Rappels d'utilisation d'une chaîne de caractères (`string`) :

```
let s = "abc" (* défini une chaîne de caractères *)
s.[1] (* donne 'b' *)
String.length s (* donne 3 *)
"abc" ^ "def" (* concaténation *)
```

Remarque : Contrairement à `array`, le type `string` est immuable (on ne peut pas modifier un caractère d'une chaîne de caractères).

**Exercice 2.**
Écrire une fonction `sous_mot : string -> string -> bool` qui teste si un mot est un sous-mot d'un autre, en complexité linéaire.

# II Langage

**Définition : Langage**

Un langage $L$ sur un alphabet $\Sigma$ est un ensemble de mots de $\Sigma$.

De façon équivalente, $L$ est un langage si $L \subseteq \Sigma^*$ ou encore $L \in \mathcal{P}(\Sigma^*)$.

Exemples :

1. L'ensemble $L_0 = \{\varepsilon, a, bab\}$ sur $\Sigma = \{a, b\}$.
2. L'ensemble $L_1$ des mots du dictionnaire français sur $\Sigma = \{a, b, ..., z\}$.
3. L'ensemble $L_2$ des formules arithmétiques sur $\Sigma = \{0, ..., 9, +, -, /, *\}$.
4. L'ensemble $L_3$ des programmes OCaml sur $\Sigma = \{a, ..., z, , !, <, >, ...\}$.

**Définition : Concaténation**

La concaténation $L_1L_2$ de $L_1$ et $L_2$ est définie par :

$$L_1L_2 = \{m_1m_2 \mid m_1 \in L_1,\ m_2 \in L_2\}$$

$L_1L_2$ est donc l'ensemble des mots obtenus par concaténation d'un mot de $L_1$ et d'un mot de $L_2$.

**Exercice 3.**

1. Soit $L_1 = \{a, ab\}$ et $L_2 = \{\varepsilon, b, bba\}$. Déterminer $L_1L_2$.
2. Quel lien a t-on entre $|L_1L_2|$ et $|L_1||L_2|$, dans le cas général ?

**Exercice 4.**

1. La concaténation de deux langages est-elle commutative ? Associative ?
2. Quel est l'élément neutre de la concaténation ? L'élément absorbant ?

La concaténation est distributive par rapport à l'union : $L_1(L_2 \cup L_3) = L_1L_2 \cup L_1L_3$.

**Définition : Puissance**

On définit la puissance $L^n$ du langage $L$ par récurrence :

$$L^0 = \{\varepsilon\}$$
$$L^n = L^{n-1}L, \text{ pour } n \geq 1$$

Dit autrement : $L^n = \underbrace{L \cdot \ldots \cdot L}_{n} = \{m_1 \cdots m_n \mid m_1 \in L, \ldots, m_n \in L\}$.

Exemple : $\Sigma^n$ est l'ensemble des mots de longueur $n$ sur l'alphabet $\Sigma$.

**Exercice 5.**
Soit $L$ un langage.

1. À quelle condition a t-on $L \subseteq L^2$ ?
2. Quel lien a t-on entre $L^2$ et $\{u^2 \mid u \in L\}$ ?

**Définition : Étoile de Kleene**

On définit l'étoile (de Kleene) $L^*$ d'un langage $L$ par :

$$L^* = \bigcup_{k \in \mathbb{N}} L^k$$

$L^*$ est donc l'ensemble des mots obtenus par concaténation d'un nombre quelconque de mots de $L$.
Remarque : $L^*$ contient toujours $\varepsilon$, car $L^0 = \{\varepsilon\}$.

**Exercice 6.**
Montrer que $(L^*)^* = L^*$.

# III Langages réguliers

**Définition : Langage régulier (ou : langage rationnel)**

L'ensemble $\mathrm{Reg}(\Sigma)$ des langages réguliers sur $\Sigma$ est défini par :
- $\mathrm{Reg}(\Sigma)$ contient tous les langages finis.
- $\mathrm{Reg}(\Sigma)$ est stable par union, concaténation et étoile de Kleene.
- $\mathrm{Reg}(\Sigma)$ est le plus petit ensemble de langages ayant ces propriétés (si $P$ est un ensemble de langages contenant les langages finis et stable par union, concaténation et étoile de Kleene, alors $\mathrm{Reg}(\Sigma) \subseteq P$).

Définition inductive équivalente :

**Propriété**

- Tout langage fini est régulier.
- Si $L_1$ et $L_2$ réguliers alors $L_1 \cup L_2$ est régulier.
- Si $L_1$ et $L_2$ réguliers alors $L_1 L_2$ est régulier.
- Si $L$ est régulier alors $L^*$ est régulier.

Par récurrence immédiate, si $L_1$, ..., $L_n$ sont réguliers alors $L_1 \cup ... \cup L_n$ et $L_1 L_2 ... L_n$ sont réguliers.

Attention :

- Une union infinie de langages réguliers n'est pas forcément régulière.
- Un langage particulier n'est pas forcément stable par concaténation.

Exemples :

1. Soit $m$ un mot. Alors $\{m\}$ est fini donc est un langage régulier, qu'on note aussi $m$ par abus de langage.
2. $\Sigma$ est fini donc est régulier. $\Sigma^*$ est l'étoile d'un langage régulier donc est régulier.
3. Soit $m$ un mot. L'ensemble des mots ayant comme facteur $m$ est égal à $\Sigma^* m \Sigma^*$ donc est un langage régulier.
4. Soit $m = m_1 \cdots m_n$ un mot. L'ensemble des mots ayant comme sous-mot $m$ est égal à $\Sigma^* m_1 \Sigma^* m_2 \cdots \Sigma^* m_n \Sigma^*$ donc est un langage régulier.

**Exercice 7.**
Montrer que les langages suivants sont réguliers sur $\Sigma = \{a, b\}$ :

1. Mots commençants par $a$ : _____
2. Mots commençants par $a$ et finissant par $b$ : _____
3. Mots de taille paire : _____
4. Mots de taille impaire : _____

# IV Expressions régulières

Les expressions régulières sont une notation plus concise pour représenter un langage régulier :

**Définition : Expression régulière (ou : expression rationnelle)**

L'ensemble des expressions régulières sur un alphabet $\Sigma$ est le plus petit langage $\mathcal{R}$ sur $\Sigma \cup \{\emptyset, \varepsilon, |, ^*, (, )\}$ vérifiant :
- $\forall a \in \Sigma$, $a \in \mathcal{R}$
- $\emptyset \in \mathcal{R}$, $\varepsilon \in \mathcal{R}$
- $\forall e_1, e_2 \in \mathcal{R}$, $(e_1|e_2) \in \mathcal{R}$ et $(e_1 e_2) \in \mathcal{R}$
- $\forall e \in \mathcal{R}$, $e^* \in \mathcal{R}$

Remarque : On utilise parfois + à la place de |.

On peut les représenter informatiquement par le type OCaml :

```
type 'a regexp =
    | Vide | Epsilon | L of 'a (* L a désigne la lettre a *)
    | Union of 'a regexp * 'a regexp
    | Concat of 'a regexp * 'a regexp
    | Etoile of 'a regexp
```

**Définition : Langage d'une expression régulière**

Si $e$ est une expression régulière, on définit son langage associé $L(e)$ récursivement :
- $L(a) = \{a\}$ si $a \in \Sigma$
- $L(\emptyset) = \emptyset$, $L(\varepsilon) = \{\varepsilon\}$
- $L(e|e') = L(e) \cup L(e')$
- $L(ee') = L(e)L(e')$
- $L(e^*) = L(e)^*$

Remarques :

- Par abus de langage, on confond souvent $e$ et $L(e)$.
- Une expression régulière n'est donc qu'une façon plus pratique de décrire un langage régulier, en enlevant les accolades et en remplaçant $\cup$ par $|$. C'est cette notation qui est utilisée dans les éditeurs de texte.

**Théorème**

Soit $L$ un langage. $L$ est régulier si et seulement si il existe une expression régulière $e$ telle que $L = L(e)$.

Exemples :

- $(a|b)^*$ : ensemble de tous les mots ($= \Sigma^*$).
- $(a|b)^*bb$ : mots finissant par $bb$.

**Exercice 8.**
Donner une expression régulière pour les langages suivants, sur $\Sigma = \{a, b\}$ :

1. Mots contenant au plus un $a$ : _____
2. Mots de taille $n \equiv 1 \mod 3$ : _____
3. Mots contenant un nombre pair de $a$ : _____
4. Mots contenant un nombre impair de $a$ : _____
5. Écritures en base 2 des entiers divisibles par 4 : _____

# V Équivalence d'expressions régulières

**Définition : Équivalence d'expressions régulières**

Deux expressions régulières $e_1$ et $e_2$ sont dites équivalentes, noté $e_1 \equiv e_2$, si elles définissent le même langage, c'est-à-dire si $L(e_1) = L(e_2)$.

Exemple : $(ab)^*a \equiv a(ba)^*$.

**Théorème**

Pour une expression régulière $e$ :
1. $e\emptyset \equiv$ _____
2. $e\{\varepsilon\} \equiv$ _____
3. $e \cup \emptyset \equiv$ _____
4. $\emptyset^* \equiv$ _____
5. $\varepsilon^* \equiv$ _____
6. $(e_1|e_2)e_3 \equiv$ _____
7. $e_1(e_2e_3) \equiv$ _____

# VI Induction structurelle

**Théorème : Induction structurelle sur les langages réguliers**

Soit $\mathcal{P}(L)$ une propriété sur les langages réguliers $L$ telle que :
- $\mathcal{P}(L)$ est vraie pour les langages $L$ finis (cas de base)
- $\mathcal{P}(L_1) \wedge \mathcal{P}(L_2) \implies \mathcal{P}(L_1 L_2)$
- $\mathcal{P}(L_1) \wedge \mathcal{P}(L_2) \implies \mathcal{P}(L_1 \cup L_2)$
- $\mathcal{P}(L) \implies \mathcal{P}(L^*)$

Alors $\mathcal{P}(L)$ est vraie pour tout langage régulier $L$.

Preuve :

De même pour les expressions régulières :

**Théorème : Induction structurelle sur les expressions régulières**

Soit $\mathcal{P}(e)$ une propriété sur les expressions régulières telle que :
- $\mathcal{P}(\emptyset)$, $\mathcal{P}(\varepsilon)$ sont vraies (cas de base)
- $\mathcal{P}(a)$ est vraie pour $a \in \Sigma$ (cas de base)
- $\mathcal{P}(e_1) \wedge \mathcal{P}(e_2) \implies \mathcal{P}(e_1 e_2)$
- $\mathcal{P}(e_1) \wedge \mathcal{P}(e_2) \implies \mathcal{P}(e_1 \cup e_2)$
- $\mathcal{P}(e) \implies \mathcal{P}(e^*)$

Alors $\mathcal{P}(e)$ est vraie pour toute expression régulière $e$.

**Exercice 9.**
Si $m = m_1 ... m_n$ est un mot, on définit son miroir $\widetilde{m} = m_n ... m_1$.
Si $L$ est un langage, on définit son miroir $\widetilde{L} = \{\widetilde{m} \mid m \in L\}$.

1. Donner une expression régulière du miroir de $a(a|b)^*b$.
2. Soit $e$ une expression régulière de langage $L$. Montrer que $\widetilde{L}$ est régulier.
3. Écrire une fonction OCaml `miroir : 'a regexp -> 'a regexp` renvoyant le miroir d'une expression régulière.