

I – Alphabet et mot

Définition 1 : Alphabet, lettre, mot

- Un *alphabet* est un ensemble Σ fini, dont les éléments sont des *lettres*.
- Un *mot* m de Σ est une suite finie $m_1, \dots, m_n$ de lettres de Σ , noté $m = m_1 \dots m_n$.
- La *taille* $|m|$ de m est son nombre de lettres.
- Le *mot vide*, ne contenant aucune lettre, est noté ε .
- Des mots $u = u_1 \dots u_n$ et $v = v_1 \dots v_p$ de Σ sont *égaux* si $n = p$ et : $\forall i \in \llbracket 1, n \rrbracket, u_i = v_i$.
- On note Σ^* l'ensemble des mots sur Σ et $\Sigma^+ = \Sigma^* \setminus \{\varepsilon\}$.

Attention : ε est un mot, pas une lettre.

Définition 2 : Concaténation, Puissance

- La *concaténation* de deux mots $u = u_1 \dots u_n$ et $v = v_1 \dots v_p$ est : $uv = u_1 \dots u_n v_1 \dots v_p$
- Si $m \in \Sigma^*$, on définit sa *puissance* par récurrence : $m^0 = \varepsilon$ et $m^k = mm^{k-1}$ pour $k \geq 1$.

Remarque : Σ^* muni de la concaténation est un *monoïde* : la concaténation $\cdot$ est associative, d'élément neutre ε .
 $u \mapsto |u|$ est un morphisme de $(\Sigma^*, \cdot)$ vers $(\mathbb{N}, +)$: $|uv| = |u| + |v|$.

Exercice 1

Soient $a, b \in \Sigma$ et $m \in \Sigma^*$ tels que $am = mb$. Montrer que $a = b$ et : $\exists k \in \mathbb{N}, m = a^k$.

Exercice 2 : Lemme de Lévy

Soient $u, v, x, y \in \Sigma^*$ tels que $uv = xy$. Montrer qu'il existe un unique $m \in \Sigma^*$ vérifiant :

$$(u = xm \text{ et } y = mv) \text{ ou } (x = um \text{ et } v = my)$$

Définition 3 : 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.

Exercice 3

Soit m un mot de taille n dont toutes les lettres sont différentes.

Quel est son nombre de préfixes, de suffixes, de facteurs et de sous-mots distincts ?

I.1 – Chaînes de caractères en C et en OCaml

```
#include <string.h> // tableau de char
char s[4] = "abc"; // le dernier caractère est '\0'
s[1]; // donne 'b'
s[1] = 'd'; // ok : s est mutable
strlen(s); // donne 3
```

```
"abc" ^ "def" (* concaténation *)
let s = "abc" (* type string *)
s.[1] (* donne 'b' *)
s.[1] <- 'd' (* erreur : string est immuable *)
String.length s (* donne 3 *)
```

Exercice 4

1. Écrire une fonction `int facteur(char *, char *)` testant si le premier mot est un facteur du deuxième.
2. Écrire une fonction `sous_mot : string -> string -> bool` testant si le premier mot est un sous-mot du deuxième.
3. Donner deux algorithmes de MP2I pour chercher un facteur dans un mot.

II – Langage

Définition 4 : Langage

Un *langage* L sur un alphabet Σ est un ensemble de mots sur Σ , c'est-à-dire $L \subset \Sigma^*$ ou encore $L \in \mathcal{P}(\Sigma^*)$

Langage	$\{\varepsilon, a, bab\}$	Mots du dictionnaire français	Formules arithmétiques	Programmes OCaml
Alphabet	$\{a, b\}$	$\{a, b, \dots, z\}$	$\{0, \dots, 9, +, -, /, *, (,)\}$	Symboles ASCII
Exemple de mot	bab	chat	$(2+3)*4$	<code>let x = 1</code>

Exemples de langages

II.1 – Concaténation

Définition 5 : Concaténation

La *concaténation* de deux langages L_1 et L_2 est $L_1L_2 = \{m_1m_2 \mid m_1 \in L_1, m_2 \in L_2\}$.

Remarque : La concaténation de langages est associative, d'élément neutre $\{\varepsilon\}$ et d'élément absorbant $\emptyset$.

Exercice 5

1. Comparer L_1L_2 et L_2L_1 où $L_1 = \{a, ab\}, L_2 = \{\varepsilon, b, bba\}$.
2. Quel lien a-t-on entre $|L_1L_2|$ et $|L_1||L_2|$, en général ?

Exercice 6

- Lequelles de ces propositions sont vraies ?
1. $\forall u, v, w \in \Sigma^* : uv = uw \implies v = w$.
2. $\forall L_1, L_2, L_3 \subset \Sigma^* : L_1L_2 = L_1L_3 \implies L_2 = L_3$.

II.2 – Puissance

Définition 6 : Puissance

Si $L \subset \Sigma^*$, on définit sa *puissance* par récurrence : $L^0 = \{\varepsilon\}$ et $L^k = LL^{k-1}$ pour $k \geq 1$.

Autrement dit : $L^n = \underbrace{L \dots L}_n = \{m_1 \dots m_n \mid m_1 \in L, \dots, m_n \in L\}$.

Exemple : Σ^n est l'ensemble des mots de taille n sur l'alphabet Σ . On note aussi $\Sigma^{<n} = \bigcup_{k < n} \Sigma^k$.

Exercice 7

1. Montrer que $L \subset L^2$ si et seulement si $L = \emptyset$ ou $\varepsilon \in L$.
2. Comparer L^2 et $\{m^2 \mid m \in L\}$.

II.3 – Étoile de Kleene

Définition 7 : Étoile de Kleene

L'*étoile de Kleene* d'un langage L est $L^* = \bigcup_{k \in \mathbb{N}} L^k$.

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

Exercice 8

1. Montrer que $(L^*)^* = L^*$.
2. Montrer que $L^* = L$ si et seulement si L est un *sous-monoïde* de Σ^* , c'est-à-dire L stable par concaténation et $\varepsilon \in L$.

III – Langages réguliers

Définition 8 : Langages réguliers

L'ensemble $\mathcal{L}_{\text{reg}}$ des *langages réguliers* sur Σ est le plus petit ensemble de langages, pour l'inclusion, contenant les langages finis sur Σ et stable par union, concaténation et étoile de Kleene.

Remarque : Cette définition est similaire à celle de tribu borélienne en maths (plus petite tribu contenant tous les ouverts).

En particulier :

- Si L est un langage fini alors $L \in \mathcal{L}_{\text{reg}}$.
- Si $L_1 \in \mathcal{L}_{\text{reg}}$ et $L_2 \in \mathcal{L}_{\text{reg}}$ alors $L_1 \cup L_2 \in \mathcal{L}_{\text{reg}}$ et $L_1L_2 \in \mathcal{L}_{\text{reg}}$.
- Si $L \in \mathcal{L}_{\text{reg}}$ alors $L^* \in \mathcal{L}_{\text{reg}}$.

Exercice 9

Montrer qu'on peut remplacer « contenant les langages finis sur Σ » par « $\emptyset \in \mathcal{L}_{\text{reg}}, \{\varepsilon\} \in \mathcal{L}_{\text{reg}}$ et $\{a\} \in \mathcal{L}_{\text{reg}}, \forall a \in \Sigma$ ».

Propriété 1

Si $L_1, \dots, L_n$ sont réguliers alors $L_1 \cup \dots \cup L_n$ et $L_1 \dots L_n$ 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 un langage fini donc régulier. On le note aussi m par abus de notation.
2. Σ est fini donc régulier. Donc Σ^* est régulier.
3. L'ensemble des mots ayant m comme facteur est égal à $\Sigma^* m \Sigma^*$, donc régulier.
4. Si $m = m_1 \dots m_n$, l'ensemble des mots ayant m comme sous-mot est égal à $\Sigma^* m_1 \Sigma^* m_2 \dots \Sigma^* m_n \Sigma^*$, donc régulier.

Exercice 10

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

1. Mots commençant par a .
2. Mots commençant par a et finissant par b .
3. Mots de taille paire.
4. Mots de taille impaire.

IV – Expressions régulières

Une expression régulière est une notation concise pour décrire un langage régulier, utilisée dans les éditeurs de texte.

Définition 9

L'ensemble $\mathcal{R}(\Sigma)$ des *expressions régulières* sur Σ est le plus petit ensemble vérifiant :

- $\forall a \in \Sigma, a \in \mathcal{R}(\Sigma)$
- $\emptyset \in \mathcal{R}(\Sigma)$ et $\varepsilon \in \mathcal{R}(\Sigma)$
- Si $e_1, e_2 \in \mathcal{R}(\Sigma)$ alors $(e_1 | e_2) \in \mathcal{R}(\Sigma)$ et $(e_1 e_2) \in \mathcal{R}(\Sigma)$.
- Si $e \in \mathcal{R}(\Sigma)$ alors $e^* \in \mathcal{R}(\Sigma)$.

Exemples : a^* , $(a | aba)^*$ et $a(a | b)^*b$ sont des expressions régulières.

- On utilise seulement les parenthèses nécessaires : $((ab)c | d)$ sera noté $abc | d$. De plus, on note parfois $+$ à la place de $|$.

Définition 10 : Langage dénoté par une expression régulière

Le langage $\mathcal{L}(e)$ dénoté par une expression régulière e est défini récursivement :

- $\mathcal{L}(a) = \{a\}$ si $a \in \Sigma$.
- $\mathcal{L}(\emptyset) = \emptyset$ et $\mathcal{L}(\varepsilon) = \{\varepsilon\}$
- $\mathcal{L}(e | e') = \mathcal{L}(e) \cup \mathcal{L}(e')$
- $\mathcal{L}(ee') = \mathcal{L}(e)\mathcal{L}(e')$
- $\mathcal{L}(e^*) = \mathcal{L}(e)^*$

On confond parfois e et $\mathcal{L}(e)$. Suivant le contexte, a peut désigner une lettre, un mot de taille 1 ou un langage singleton.

Théorème 1

Un langage L est régulier si et seulement s'il existe une expression régulière e telle que $L = \mathcal{L}(e)$.

Exercice 11

Donner une expression régulière pour les langages suivants sur $\Sigma = \{0, 1\}$:

1. Mots contenant au plus un 1.
2. Mots dont la taille est congrue à 1 modulo 3.
3. Écritures en base 2 des entiers divisibles par 4.
4. Mots ne contenant pas le facteur 11.

Exercice 12

Écrire les fonctions suivantes sur une expression régulière e :

1. vide : 'a regexp -> bool qui teste $\mathcal{L}(e) = \emptyset$.
2. epsilon : 'a regexp -> bool qui teste $\mathcal{L}(e) = \{\varepsilon\}$.
3. fini : 'a regexp -> bool qui teste si $\mathcal{L}(e)$ est fini.

```
type 'a regexp =
| Vide | Epsilon | L of 'a
| Union of 'a regexp * 'a regexp
| Concat of 'a regexp * 'a regexp
| Etoile of 'a regexp
```

Description	Notation	Exemple	Ensemble d'appartenance
Lettre	a	a	
Alphabet	Σ	$\{a, b\}$	
Mot vide	ε	ε	
Mot	m	aba	
Langage	L	$\{\varepsilon, a, ab\}$	
Expression régulière	e	ab^*	
Langage d'une expression régulière	$\mathcal{L}(e)$	$\{ab^n \mid n \in \mathbb{N}\}$	
Ensemble des langages réguliers	$\mathcal{L}_{\text{reg}}$	$\mathcal{L}_{\text{reg}}$	

Définition 11 : Équivalence d'expressions régulières

Deux expressions régulières e_1 et e_2 sont dites *équivalentes*, noté $e_1 \equiv e_2$, si $\mathcal{L}(e_1) = \mathcal{L}(e_2)$.

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

Exercice 13

Pour des expressions régulières e, e_1, e_2 et e_3 :

1. $e\emptyset \equiv$

2. $e\varepsilon \equiv$

3. $e \mid \emptyset \equiv$
4. $\emptyset^* \equiv$

5. $\varepsilon^* \equiv$

6. $(e_1 \mid e_2)e_3 \equiv$

V – Induction structurale

Théorème 2

Soit $\mathcal{P}(L)$ une propriété sur les langages telle que :

- $\mathcal{P}(L)$ est vraie pour les langages L finis
- $\mathcal{P}(L_1) \wedge \mathcal{P}(L_2) \implies \mathcal{P}(L_1 L_2)$
- $\mathcal{P}(L_1) \wedge \mathcal{P}(L_2) \implies \mathcal{P}(L_1 \cup L_2)$
- $\mathcal{P}(L) \implies \mathcal{P}(L^*)$

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

Théorème 3

Soit $\mathcal{P}(e)$ une propriété sur $e \in \mathcal{R}(\Sigma)$ telle que :

- $\mathcal{P}(\emptyset), \mathcal{P}(\varepsilon)$ et $\mathcal{P}(a)$ sont vraies pour $a \in \Sigma$
- $\mathcal{P}(e_1) \wedge \mathcal{P}(e_2) \implies \mathcal{P}(e_1 e_2)$
- $\mathcal{P}(e_1) \wedge \mathcal{P}(e_2) \implies \mathcal{P}(e_1 \mid e_2)$
- $\mathcal{P}(e) \implies \mathcal{P}(e^*)$

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

Exercice 14 : Miroir

On note $\widetilde{m} = m_n \dots m_1$ le *miroir* d'un mot $m = m_1 \dots m_n$ et $\widetilde{L} = \{\widetilde{m} \mid m \in L\}$ celui d'un langage L .

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

HP VI – Complément : dénombrabilité et langages non réguliers

Soit Σ un alphabet non vide.

Théorème 4

Σ^* est infini dénombrable.

Corollaire 1

Un langage $L \subset \Sigma^*$ est au plus dénombrable.

Théorème 5 : Cantor

Un ensemble E ne peut pas être en bijection avec $\mathcal{P}(E)$.

Corollaire 2

L'ensemble $\mathcal{P}(\Sigma^*)$ des langages sur Σ est indénombrable.

$\mathcal{P}(\Sigma^*)$ est indénombrable alors que l'ensemble des programmes OCaml est dénombrable.

Donc il existe des langages L pour lesquels **Problème 1** est indécidable (aucun programme OCaml ne le résout) :

Problème 1 : Appartenance à un langage

Étant donné un mot m , est-ce que $m \in L$?

Théorème 6

$\mathcal{L}_{\text{reg}}(\Sigma)$ est infini dénombrable.

Corollaire 3

Il existe des langages non réguliers sur Σ .