

# Théorie des langages

Lionel Rieg (TL1)  
Marie-Laure Potet (TL1)

Xavier Nicollin (TL2)

Grenoble INP - Ensimag, 1<sup>re</sup> année

Année 2023-2024

# Motivations et objectifs

- **Théorie des langages** : étude des langages formels
- **Langage formel** : ensemble de *mots, phrases, textes, énoncés* défini « formellement », sans considération sémantique a priori
- Objectif : trouver des moyens de *définir* des langages formels, et des moyens de les *reconnaître* (savoir si un mot appartient au langage)

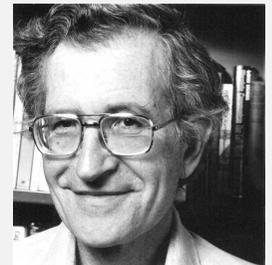
# À quoi ça sert ?

- Définition de langages de programmation  
  ↳ Analyse lexicale, syntaxique d'un programme (cf. TL2)
- Calculabilité, complexité (cf. TL2)
- Description de la structure de documents XML
- Recherche de texte dans un document
- Génération automatique d'images
- Bioinformatique
- Traitement automatique des langues naturelles
- Cryptographie
- Contrôle de systèmes
- ...

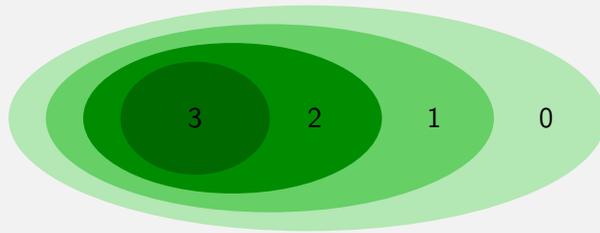
# La référence

Noam Chomsky (1928-), États-Unis

- Linguiste, philosophe, logicien, activiste
- 1956 : définition des grammaires formelles
  - ▶ Ensemble de règles permettant d'engendrer des langages formels
  - ▶ Classification des grammaires (et des langages engendrés) en fonction de la forme de leurs règles
  - ▶ **Hiérarchie de Chomsky**



## Hierarchie de Chomsky



Type	Langage	Engendré par	Reconnu par
0	Calculatoirement énumérable	Grammaire générale	<b>Machine de Turing</b>
1	Sous-contexte	Grammaire sous-contexte	Machine de Turing linéairement bornée
2	Hors-contexte	<b>Grammaire hors-contexte</b>	Automate à pile
3	Régulier	Grammaire linéaire à droite	<b>Automate fini</b>

## Description du cours de TL1

- 11 CM en amphis, 10 séances de TD, 1 séance de TP/projet
  - ▶ 7 CM sur les langages réguliers et automates
  - ▶ 4 CM sur les grammaires
  - ▶ Horaires : on commence **à l'heure**, on finit 10 minutes en avance
- Matériel disponible en ligne sur Chamilo
  - ▶ Quizz (à faire entre CM et TD, ouverts 15 jours après chaque cours)
  - ▶ Diapos à trous sans animation (disponible à l'avance)
  - ▶ Diapos (en ligne chaque semaine après le 2<sup>e</sup> CM)
  - ▶ Sujets d'exercices de TD (également en papier)
  - ▶ Annales d'examen (2 dernières années)
  - ▶ Polycopié
- Permanences (*office hours*)
  - ▶ 1h30 par groupe
  - ▶ Chaque intervenant de TD place sa permanence

## Théorie des langages 1

### Cours 1 : Vocabulaires, mots, langages, induction

L. Rieg

Grenoble INP - Ensimag, 1<sup>re</sup> année

Année 2023-2024

## Définitions

### Définitions (Vocabulaire, mot)

- Un **vocabulaire** (ou alphabet) est un ensemble **fini** quelconque. Ses éléments sont appelés des **symboles** (ou lettres).
- Un **mot** sur un vocabulaire  $V$  est une **séquence finie** de symboles de  $V$ .

### Exemples

$V$	mot sur $V$	notations abrégées
$\{a, b, \dots, z\}$	$[b, r, o, c, c, c, o, l, i]$	<i>broccoli</i> , $broc^3oli$
$\{0, \dots, 9\}$	$[2, 0, 2, 0]$	2020, $(20)^2$
$\{a, b, ab\}$	$[ab]$ , $[a, b]$	<i>ab</i> ?

### Définition

- On note  $\varepsilon$  le mot correspondant à la séquence vide (« *mot vide* »).

## Définitions (suite)

### Définition (longueur d'un mot)

Soit  $V$  un vocabulaire et soit  $u = u_1 \cdots u_n$  un mot sur  $V$ .  
La **longueur** de  $u$  est alors  $n$ , et on note  $|u| = n$ .

### Remarque

En particulier, on a  $|\varepsilon| = 0$ .

### Définitions

- Pour  $n \in \mathbb{N}$ ,  $V^n$  est l'ensemble des mots sur  $V$  de longueur  $n$ .  
Par abus de notation, on identifie  $V$  et  $V^1$ .
- $V^+$  est l'ensemble des mots sur  $V$  de longueur au moins 1.
- $V^*$  est l'ensemble des mots sur  $V$ .
- Pour  $w \in V^*$  et  $a \in V$ ,  $|w|_a$  est le nombre d'occurrences de  $a$  dans  $w$ .

## Exemples

### Exemples

- Soient  $V = \{a, b\}$  et  $w = ababb$ .
- Soient  $V = \{cd, dc\}$  et  $w = cdcdcd$ .

### Proposition

On a les égalités suivantes :

$$V^* = \bigcup_{n \geq 0} V^n$$
$$V^+ = \bigcup_{n > 0} V^n$$

## Concaténation

### Définition

Soit  $V$  un vocabulaire,  $u = u_1 \cdots u_n$  et  $v = v_1 \cdots v_m$  deux mots de  $V^*$ .  
La **concaténation de  $u$  et  $v$** , notée  $u.v$ , est le mot de  $V^*$  défini par

$$u.v = u_1 \cdots u_n v_1 \cdots v_m$$

### Exemple

Soient  $u = bac$  et  $v = aacb$ .

### Théorème

$(V^*, \cdot, \varepsilon)$  est un monoïde (« groupe sans inverse »)

- $\cdot$  associative :  $u.(v.w) = (u.v).w$
- $\varepsilon$  élément neutre :  $\varepsilon.u = u.\varepsilon = u$

### Notation

On pourra noter  $uv$  au lieu de  $u.v$ .

## Concaténation (suite)

### Proposition

Si  $|u| = i$  et  $|v| = j$ , alors  $|uv| = i + j$ .

### Définitions

Soient  $v, z \in V^*$ . On dit que  $v$  est un :

- **sous-mot** de  $z$  ssi  $\exists u, w \in V^*$  tels que  $z = u.v.w$
- **préfixe** de  $z$  ssi  $\exists w \in V^*$  tel que  $z = v.w$
- **suffixe** de  $z$  ssi  $\exists u \in V^*$  tel que  $z = u.v$

## Langages

### Définition

On appelle **langage** sur  $V$  tout **sous-ensemble de  $V^*$** .

### Exemples

- $\emptyset \subseteq \{a, b\}^*$
- $\{a, b\}^* \subseteq \{a, b\}^*$
- $\{abab, ab, abba\} \subseteq \{a, b\}^*$
- $\{a^n b^n \mid n \geq 0\} \subseteq \{a, b\}^*$  ( $\{\varepsilon, ab, aabb, aaabbb, \dots\}$ )
- « Ensemble des programmes Python »  $\subseteq$  Unicode\*

### Remarque

On s'intéressera en TL à **définir** et **reconnaître** des sous-ensembles de  $V^*$ .

## Comment définir un langage / un ensemble ?

- Par extension
  - ▶ On énumère les éléments de l'ensemble.
  - ▶  $\{a, b\}^3 = \{aaa, aab, aba, abb, baa, bab, bba, bbb\}$
  - ▶  $P = \{0, 2, 4, 6, 8, 10, \dots\}$
- Par compréhension
  - ▶ On décrit les caractéristiques des éléments de l'ensemble.
  - ▶  $P = \{n \in \mathbb{N} \mid \exists k \in \mathbb{N}, n = 2k\}$
- Par **induction structurelle**
  - ▶ On explique comment **construire** les éléments de l'ensemble.
  - ▶ Fréquemment utilisé en informatique
  - ▶  $P$  est le **plus petit** ensemble (pour l'inclusion) tel que :
    - ★  $0 \in P$ , et
    - ★ si  $n \in P$  alors  $n + 2 \in P$ .

## Définition par induction structurelle

**Principe général** : on définit un ensemble en spécifiant :

- Des **cas de base** : quels sont les éléments les « plus simples » de l'ensemble ?
- Des **règles de construction** : comment peut-on, en partant d'éléments de l'ensemble, en construire de nouveaux ?

### Exemples

Définitions inductives de  $V^*$  et  $L \stackrel{\text{def}}{=} \{a^n b^n \mid n \geq 1\}$  :

## Définition générale

### Définition (Ensemble inductif)

Soit  $U$  un ensemble ; définir un ensemble  $E \subseteq U$  par **induction structurelle** consiste à donner :

- un ensemble **non vide d'atomes**  $B = \{b_1, \dots, b_n\} \subseteq U$
- un ensemble  $K = \{\kappa_1, \dots, \kappa_m\}$  de **constructeurs** inductifs, où  $\kappa_i : U^{a_i} \rightarrow U$  et  $a_i > 0$  pour tout  $i$  ( $a_i$  : **arité** de  $\kappa_i$ )

$E$  est alors le **plus petit ensemble** tel que :

- $B \subseteq E$
- $\forall i \in [1, m]$ , si  $(e_1, \dots, e_{a_i}) \in E^{a_i}$ , alors  $\kappa_i(e_1, \dots, e_{a_i}) \in E$

### Exemples

$V^*$ , listes, arbres, formules logiques, ...

## Exemple

Posons  $V = \{a, b\}$ , et soit  $L_0$  le langage défini par induction structurale de la façon suivante :

- Base :  $\varepsilon \in L_0$  ( $b_1 = \varepsilon$ )
- Induction (constructeurs) : si  $u, v \in L_0$  alors
  - ▶  $aubv \in L_0$  ( $\kappa_1(u, v) = aubv$ )
  - ▶  $bua v \in L_0$  ( $\kappa_2(u, v) = buav$ )

Quelques mots dans  $L_0$  :

$\varepsilon, ab, ba, aabb, abab, bbaaaabb$

**Exercice : construire les trois derniers mots**

## Énumération d'un ensemble inductif

### Théorème (Admis)

Soit  $E$  un ensemble défini par induction sur l'ensemble d'atomes  $B$  et l'ensemble de constructeurs  $K$ .

Alors  $E = \bigcup_{n \geq 0} E_n$ , où la suite  $(E_n)$  est définie par :

$$E_0 \stackrel{\text{def}}{=} B,$$
$$E_{n+1} \stackrel{\text{def}}{=} E_n \cup \{\kappa_i(e_1, \dots, e_{a_i}) \mid \kappa_i \in K, e_1, \dots, e_{a_i} \in E_n\}$$

**algorithme**  $E =$

$n \leftarrow 0, E_0 \leftarrow B$

**répéter**

$E_{n+1} \leftarrow E_n \cup \{\kappa_i(e_1, \dots, e_{a_i}) \mid \kappa_i \in K, e_1, \dots, e_{a_i} \in E_n\}$

$n \leftarrow n + 1$

**jusqu'à**  $E_{n+1} = E_n$

**renvoyer**  $E_n$

**Question : Est-ce que ça termine toujours ?**

## Fonction définie inductivement

### Définition

Soit  $E$  un ensemble défini inductivement par l'ensemble d'atomes  $B$  et l'ensemble de constructeurs  $K$ , et soit  $U'$  un ensemble quelconque.

Pour **définir une fonction**  $f : E \rightarrow U'$ , il suffit d'expliciter :

- les images des atomes  $f(b_1), \dots, f(b_n)$  ;
- la façon dont la fonction « interagit » avec les constructeurs : comment **exprimer**  $f(\kappa_i(e_1, \dots, e_{a_i}))$  **en fonction de**  $f(e_1), \dots, f(e_{a_i})$ .

Remarque :  $f(E)$  est un ensemble inductif !

### Exemple (Longueur d'un mot)

$|\_|\_| : V^* \rightarrow \mathbb{N}$

- Cas de base :  $|\varepsilon| = 0$
- Constructeurs inductifs :  $|xw| = 1 + |w|$

## Lien avec la programmation récursive

**L'induction structurale formalise la programmation récursive !**

### Exemple (Somme des éléments d'une liste)

- en OCaml :

```
let rec somme_list l =  
  match l with  
  | []      -> 0  
  | x :: xs -> x + somme_list xs
```

- En Haskell :

```
sommeList :: [Int] -> Int  
sommeList [] = 0  
sommeList (x:xs) = x + sommeList xs
```

## Exemples

Soient les fonctions  $dl$  et  $pa : L_0 \rightarrow \mathbb{N}$  telles que  $\forall w \in L_0$ ,

$$dl(w) = |w|_a$$

$$pa(w) = \max \{|x| \mid x \text{ préfixe de } w \wedge x \in \{a\}^*\}$$

**Exercice** : définir les fonctions  $dl$  et  $pa$  par induction structurelle  
 $L_0$  est défini par **1** cas de base et **2** constructeurs donc **3 cas** à considérer.

## Preuve par induction structurelle

### Définition

Soit  $E$  un ensemble défini inductivement par l'ensemble d'atomes  $B$  et l'ensemble de constructeurs  $K$ , et soit  $P$  une propriété sur  $E$ .

Pour montrer que  $P(e)$  est vraie pour tout  $e \in E$ , on peut :

- montrer que  $P(b_1), \dots, P(b_n)$  sont vrais;
- pour  $i \in [1, m]$ , montrer que si  $P(e_1), \dots, P(e_{a_i})$  sont tous vrais, alors  $P(\kappa_i(e_1, \dots, e_{a_i}))$  l'est également.

### Remarque

La preuve par induction structurelle est une *généralisation de la preuve par récurrence* :

- Base : 0
- Constructeur : la fonction successeur  $s : n \mapsto n + 1$

## Application

Soit  $M_0 = \{w \in \{a, b\}^* \mid |w|_a = |w|_b\}$ . Montrer que  $L_0 \subseteq M_0$ .

**Exercice(★)** Montrer que  $M_0 \subseteq L_0$ .