Université Mohammed Khider-BISKRA

Faculté des Sciences Exactes et la nature et de la vie

Département d'informatique

Cours du Module

Théorie des langages

Pr. CHERIF Foudil

Chapitre 3: Typologie des grammaires

1- Définition d'une grammaire

C'est à l'aide d'une **grammaire** que l'on décrit de façon générique et productive les expressions bien formées d'un langage.

Une grammaire est un système formel défini par un **axiome** et des **règles** dites règles de production.

Les phrases se dérivent à partir de l'axiome et par application successive des règles. Les règles de la grammaire sont construites avec des symboles effectifs dits symboles **terminaux** et des symboles outils dits **non terminaux**, qui dénotent des morceaux de chaînes correctes durant la construction du langage.

Chapitre 3: Typologie des grammaires

Exemple: soit la phrase suivante - 19.5 10^{-3}

L : ensemble des nombres de cette forme (nombres décimaux)

ND : Un nombre décimal

On peut former une grammaire qui génère l'ensemble L des nombres décimaux comme suit:

ND → S E P E F

E → C E

E → C

C → 0 / 1 / 2 / 3 / 4 / 5 / 6 / 7 / 8 / 9

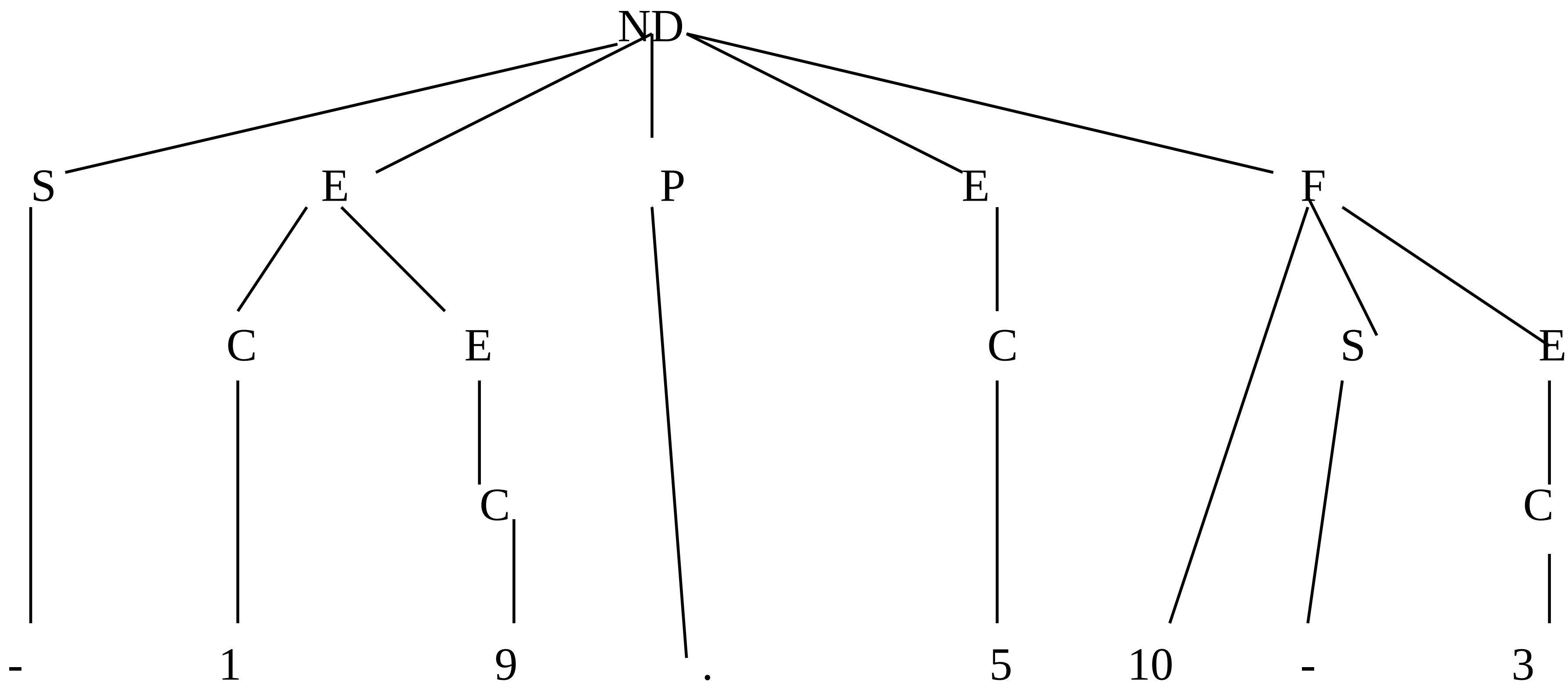
P → .

F → 10 S E

S → + / -

Chapitre 3: Typologie des grammaires

Un nombre décimal est représenté par un arbre de dérivation



Chapitre 3: Typologie des grammaires

Concepts à définir :

- Nombre décimal : ND c'est la racine de l'arbre de dérivation
- Eléments syntaxiques : (ND, S, E, P, F, C) Vocabulaire non terminal
- Alphabet initial : (0,1,2,3,4,5,6,7,8,9,+,-,.,10) Vocabulaire terminal
- Ensemble des règles : C'est l'articulation des différents éléments entre eux.

Chapitre 3: Typologie des grammaires

2-Définition formelle d'une grammaire

Une grammaire est un quadruplet G = (Vt, Vn, S, R) où

- Vt : est le vocabulaire terminal, i.e, . Est un ensemble fini non vide.
- Vn : est le vocabulaire non terminal, i.e, l'ensemble des symboles qui n'apparaissent pas dans les mots générés, mais qui sont utilisés au cours de la génération. Est un ensemble fini non vide.
- S : est un élément de Vn, est le symbole de départ ou axiome. C'est à partir de ce symbole que l'on commencera la génération de mots au moyen des règles de la grammaire.
- R : est un ensemble de règles dites de réécriture ou de production de la forme: $u \rightarrow v$ tel que $u \in (Vt \cup Vn)^+$ et $v \in (Vt \cup Vn)^*$
- Et $Vt \cap Vn = \emptyset$

Chapitre 3: Typologie des grammaires

Terminologie :

- Une suite de symboles terminaux et non terminaux(un élément de $(Vt \cup Vn)^*$ est appelée forme.
- Une règle $u \rightarrow v$ telle que $v \in Vt^*$ est appelée une règle terminale.

<u>3- Dérivation dans une grammaire</u>

•Dérivation directe notée par "====> " :

soit une règle de R $u \rightarrow v$ et soient x, y des mots de $(Vt \cup Vn)^*$ on dit que y dérive directement de x dans G (x ===> y) si et seulement si

$x = \alpha u \beta$ et $y = \alpha v \beta$ $\alpha, \beta \in (Vt \cup Vn)^*$

- **Dérivation indirecte notée par "==*==> " :**

on dit que y dérive de x dans G (x ==*=> y) si et seulement si il existe une suite finie $W_0, W_1, \ldots, W_n$ telle que $W_0 = x$ $W_n = y$ et W_i ====> W_{i+1} $0 \leq i \leq n$

x ===> W_1 …. W_{n-1} ====> y

<u>4- Langage engendré par une grammaire</u>

Le langage défini, ou généré, par une grammaire est l'ensemble des mots qui peuvent être obtenus à partir du symbole de départ par application des règles de la grammaire. Plus formellement est l'ensemble des dérivations terminales de son axiome.

G = (Vt, Vn, S, R)

L(G) = { x / x Є Vt^* et S ==*==> x }

Chapitre 3: Typologie des grammaires

Remarque importante:

Une grammaire définit un seul langage. Par contre un langage peut être engendré par plusieurs grammaires différentes. On dit que ces deux **grammaires sont équivalentes**.

On dit que G_1 et G_2 sont équivalentes si et seulement si $L(G_1) = L(G_2)$

<u>Exemples: construction des langages</u>

Trouver les langages engendrés par ces grammaires:

1. G_1=({a,b}, {S},S,R)

   R = (S → a S b , S → ab)

2. G_2=({_/ , _}, {S, A, U, V}, S, R)

   R = (S → U A V   S → U V   A → V S U

   A → V U   U → _/   V → _)

3. G_3=({a,b}, {S, A, B}, S, R)

R = (S → AS   S → Ab   A → A B   B → a A   )

4. G_4=({a}, {S},S,R)

R= (S → A S A   S → ϵ   A → S A   A → A S a   )

Exemples de langages :

- $L1 = \{ab, a, ba, bb\}$;
- $L2 = \{\omega \in \{a, b\}* / |\omega| > 3\}$;
- $L3 = \{\omega \in \{a, b\}* / |\omega| \equiv 0\ [5]\}$;

Chapitre 3: Typologie des grammaires

5- <u>Types de grammaires</u>

En introduisant des critères plus au moins restrictifs sur les règles de production, on obtient des classes de grammaires hiérarchisées, ordonnées par inclusion. La classification des grammaires, définies en 1957 par Noam CHOMSKY, distingue les quatre classes suivantes:

5.1- Grammaires de type 0

Les grammaires sans restriction sur les règles, donc toutes les grammaires sont de type 0.

$u \rightarrow v \qquad u \in (Vn \cup Vt)^{+} \quad \text{et} \quad v \in (Vn \cup Vt)^{*}$

Chapitre 3: Typologie des grammaires

5.2 Grammaires de type 1

Les grammaires de type 1 sont appelées aussi les grammaires sensibles au contexte ou contextuelles. Les règles de la grammaire sont de la forme:

$u\ A\ v \rightarrow u\ W\ v$

$A \in V_n$, $W \in (V_n \cup V_t)^+$ et $u,v \in (V_n \cup V_t)^*$

Autrement dit, le symbole non terminal A est remplacé par la forme W mais si on les contextes u à gauche et v à droite. On restreint les règles en obligeant le membre droit à être au moins aussi long que le membre de gauche. Ceci oblige à exclure le mot vide de la grammaire.

5.3- Grammaires de type 2

Les grammaires de type 2 sont appelées aussi les grammaires hors contexte, contexte libre, algébrique ou de Chomsky. C'est la grammaire la plus utilisée en théorie des langages et en compilation.

Les règles de la grammaire sont de la forme:

$A \rightarrow W$

$A \in Vn$, $W \in (Vn \cup Vt)^*$

Autrement dit, le membre de gauche est constitué d'un seul symbole non terminal.

Chapitre 3: Typologie des grammaires

5.4- Grammaires de type 3

Les grammaires de type 3 sont appelées aussi les grammaires régulières à droite (respectivement à gauche)

Les règles de la grammaire sont de la forme:

$A \rightarrow a\,B$ (respectivement $A \rightarrow B\,a$)

ou $A \rightarrow a$

$A, B \in V_n$ et $a \in V_t$

Autrement dit, le membre de gauche de chaque règle est constitué d'un seul symbole non terminal, et le membre de droite est constitué d'un symbole terminal éventuellement suivi (respectivement précédé) d'un seul non terminal.

Chapitre 3: Typologie des grammaires

6- Type de langage

A chaque type de grammaire est associé un type de langage: Les grammaires de type 3 génèrent les langages réguliers, les grammaires de type 2 génèrent les langages hors contexte et les grammaires de type 1 génèrent les langages contextuels et les grammaires de type 0 permet de générer tous les langages " décidables ", autrement dit, tous les langages qui peuvent être reconnus en un temps fini par une machine.

Les langages qui ne peuvent pas être générés par une grammaire de type 0 sont dits " indécidables ".

Chapitre 3: Typologie des grammaires

Type de langage

Ces langages sont ordonnés par inclusion: l'ensemble des langages générés par les grammaires de type n est strictement inclus dans celui des grammaires de type n-1 (pour n = 1,2,3).

Exemple :

- une grammaire de type 3 est aussi de type 2, 1, 0
- une grammaire de type 2 est aussi de type 1, 0 mais elle n'est pas de type de 3
- une grammaire de type 1 est aussi de type 0

Chapitre 3: Typologie des grammaires

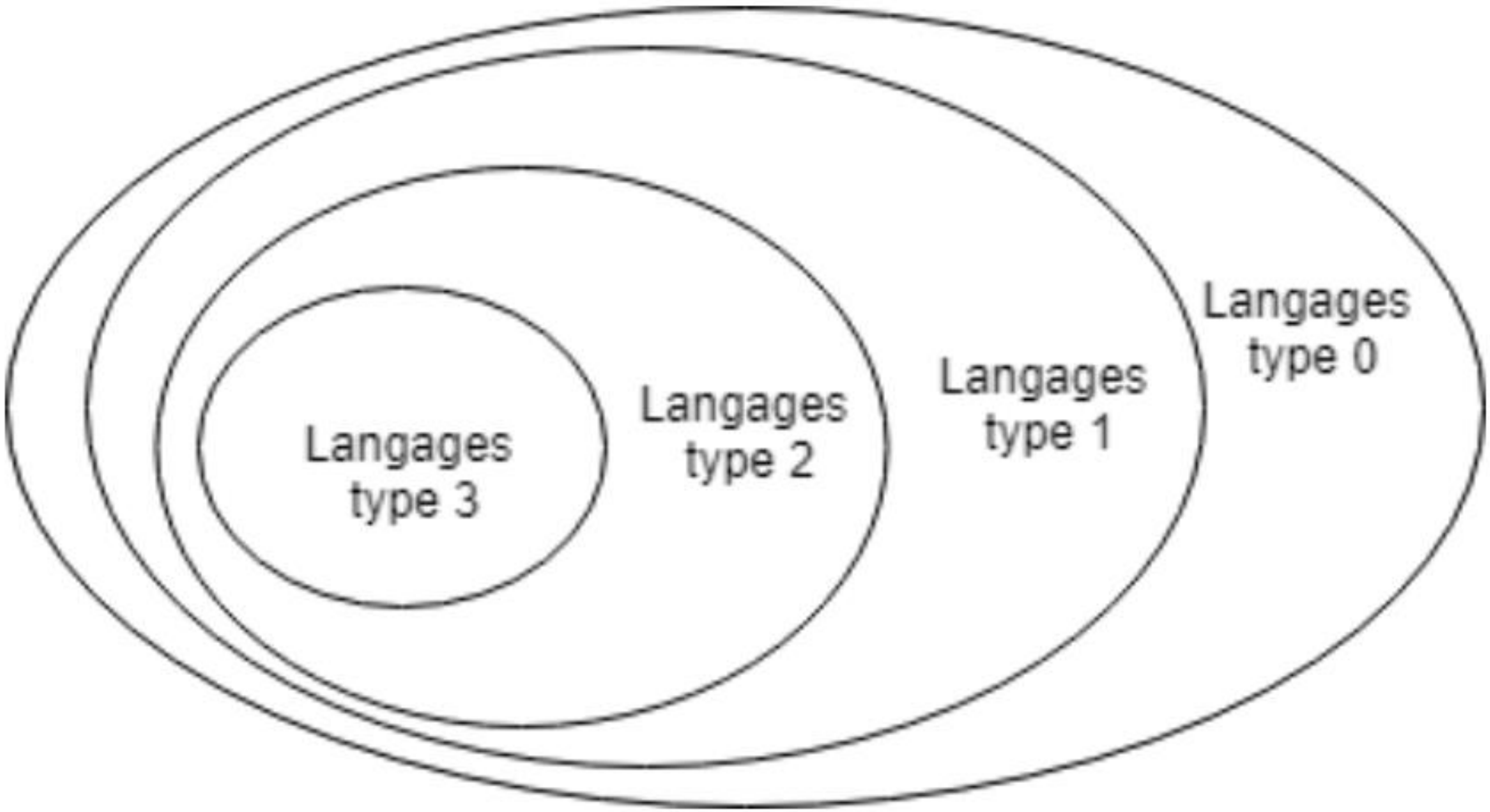
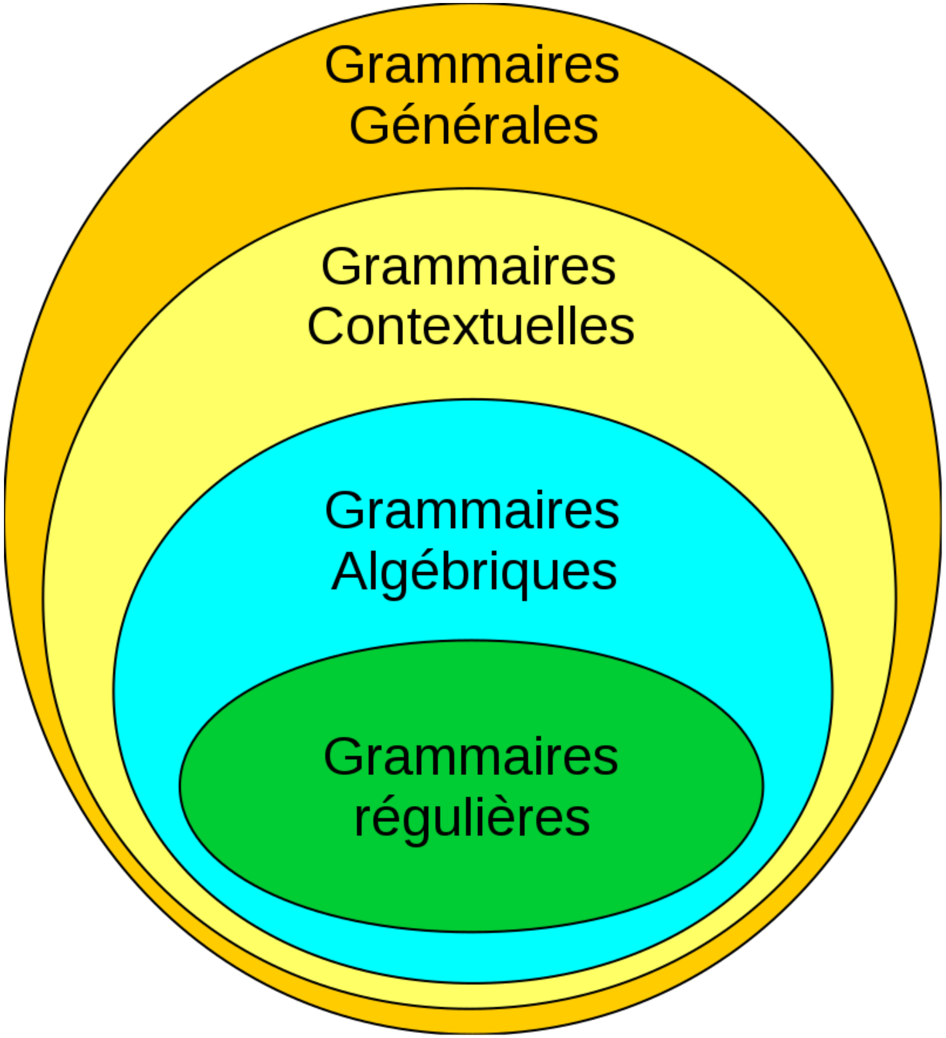


FIGURE 1.1 – Relation d'inclusion entre les types de langages.

Chapitre 3: Typologie des grammaires



Conclusion

Enfin, à chaque type de grammaire est associé un type d'automate qui permet de reconnaître les langages de sa classe:

les langages réguliers sont reconnus par des automates finis,

les langages hors-contexte sont reconnus par des automates à pile;

les langages à contexte liés sont reconnus par des machines à bornes linéaires

et les langages de type 0 sont reconnus par les machines de Turing.

La machine de Turing peut être considérée comme le modèle de machine le plus puissant qu'il soit, dans la mesure où tout langage qui ne peut pas être traité par une machine de Turing, ne pourra pas être traité par une autre machine.