Chapi tre 2:Grammaires

Plan

- 1. Définitions
- 2. Dérivation et langage engendré
- 3. Arbre de dérivation
- 4. Hiérarchie de Chomsky

Définition

Une grammaire G est un quadruplet (V_N, V_T, S, R) où :

- V_N désigne un ensemble fini appelé vocabulaire non terminal.
- V_T désigne un ensemble fini appelé vocabulaire terminal (On note $V=V_N \cup V_T$).
- S: symbole initial ou axiome, est un élément de V_N.
- R est l'ensemble fini des règles ; $R \subset (V+\backslash V_T)\ x V^*$, une règle $r \in R$ est notée :

$$\alpha(r) \rightarrow \beta(r)$$

tel que :

- $\alpha(r)$: la gauche. - $\beta(r)$: la droite.

Définition

- Exemple :

• Soit $G = (\{S\}, \{0,1\}, S, \{ S \longrightarrow 0S1 \ , S \longrightarrow 01 \})$.

G est une grammaire de Chomsky.

• Il existe plusieurs façons de représenter les règles d'une grammaire de Chomsky :

Couples	Dérivation	BNF
(α, β)	$\alpha \rightarrow \beta$	$\alpha ::= \beta \mid \gamma$
(α, γ)	$\alpha \rightarrow \gamma$	

Notation

- Lorsque plusieurs règles de production d'une grammaire ont une même forme en partie gauche, on pourra les factoriser en séparant les parties droites par des traits verticaux.

$A \rightarrow a\,A$ et $A \rightarrow \varepsilon$ On écrira: $A \rightarrow a\,A \mid \varepsilon$

Dérivation

- Un mot y dérive immédiatement d'un mot x si et seulement s'il existe une règle r et deux mots g et d de V* tels que ;

$$x=g\alpha(r)d \text{ et } y=g\beta(r)d.$$

- On notera : $x \xrightarrow{r} y$

- Soit $\Rightarrow$ la fermeture réflexive et transitive de la relation $\rightarrow$. La relation $\Rightarrow$ est appelé dérivation.
- On notera r la suite de règles permettant de dériver y de x donc :

$$x \overset{\hat{r}}{\Rightarrow} y$$

Exemple

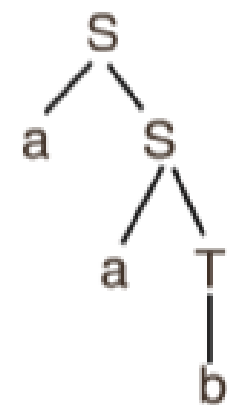
- Soit la grammaire

$G = (\{a,b\}, \{S,T\}, S, \{S \rightarrow aS \mid aT, T \rightarrow bT \mid b\})$.

Elle génère les mots abb et aab parce que

$S \rightarrow aT \rightarrow abT \rightarrow abb$ et $S \rightarrow aS \rightarrow aaT \rightarrow aab$.

Ce qui donne donc l'arbre syntaxique suivant



On peut facilement voir alors que le langage généré par cette grammaire est : tous les mots sur $\{a,b\}$ de la forme $a^m b^n$ avec $m,n > 0$.

Langage engendré par une grammaire

- Le langage engendré par une grammaire G, noté L(G), est l'ensemble des mots terminaux dérivant de S.
- Formellement :

$$\boldsymbol{L(G) = \left\{ x \in V_T^* / \exists \hat{r} \in R^+, S \overset{\hat{r}}{\Rightarrow} x \right\}}$$

Exemple

- 000111 (noté aussi $0^3 1^3$) est un mot du langage engendré par la grammaire G de l'exemple précédent.
- Soit les deux règles r1 et r2 telles que :

$$r_1 : S \rightarrow 0S1 \text{ et } r_2 : S \rightarrow 01.$$

- Donc : $0^3 1^3$ se dérive de S par l'application de deux fois la règle r1 puis d'une fois la règle r2.
- On aura doc : ***S***

$$S \xrightarrow{r_1} 0S1 \xrightarrow{r_1} 00S11 \xrightarrow{r_2} 000111$$

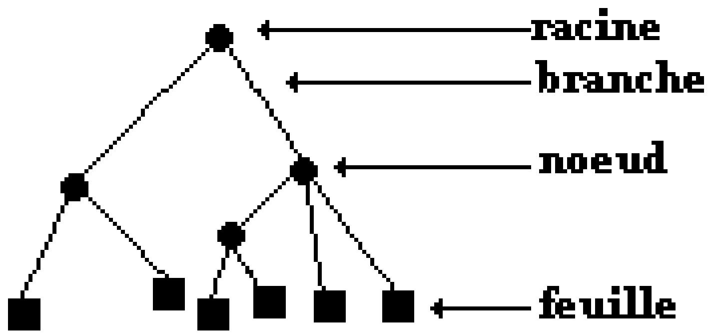
Arbre de dérivation d’un mot

- Soit la grammaire G, G = (V_N, V_T, S, R).
  Un arbre étiqueté est un " **arbre de dérivation** " dans G **<u>ssi</u>** :
- L’alphabet des étiquettes est inclus dans V_N È V_T.
- Les noeuds sont étiquetés par des éléments de V_N.
- Les feuilles sont étiquetées par des éléments de V_T.
- L’étiquette de tout noeud est un élément de V_N.
- Pour tout noeud $< \mathbf{A}, f_1, f_2, \ldots, f_n >$ on associe une règle R
- de la forme : $\mathbf{A} \rightarrow f_1 f_2 \ldots f_n$
  (règle de dérivation dans G).

Arbre de dérivation d’un mot

- Un arbre de dérivation est traditionnellement dessiné la racine en haut. Sur un tel arbre dessiné, le mot dérivé s’obtient en concaténant les étiquettes des feuilles de gauche à droite.

- Représentation graphique d’un arbre :



Exemples

- La grammaire G est la suivante:

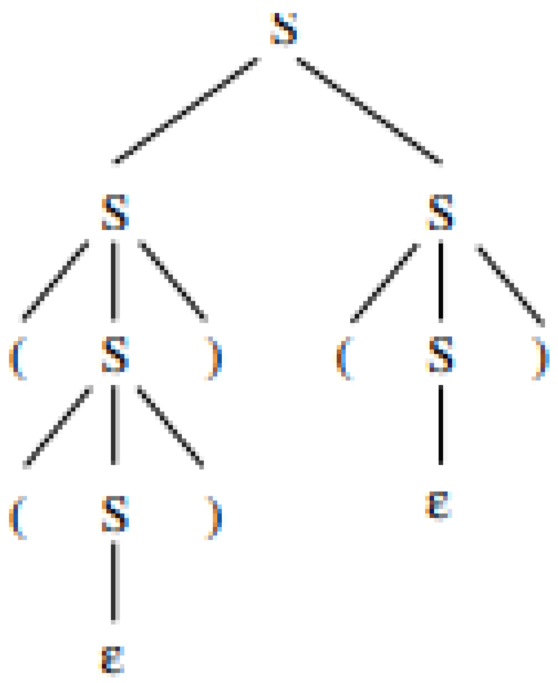
  $S \to \varepsilon$, $S \to SS$, $S \to (S)$

- Les lettres terminales sont (et) et S est une lettre non terminale, qui est également l'axiome de la grammaire.
- Par exemple, le mot (())() s'obtient par :

$$
\begin{aligned}
S \to SS &\to (S)S \\
&\to ((S))S \\
&\to ((\varepsilon))S = (())S \\
&\to (())(S) \\
&\to (())(\varepsilon) = (())().
\end{aligned}
$$

Exemples

- Arbre de dérivation du mot (())() :



Différentes types de grammaires

- Une grammaire est dite de type 3 (linéaire, régulière) est une grammaire dans laquelle toute production $\alpha \rightarrow \beta$ est soit de la forme $A \longrightarrow aB$, avec a dans X et A,B dans N, soit de la forme $A \longrightarrow a$.
- Formellement:
  - $\forall \boldsymbol{r} \in \boldsymbol{R}\ \boldsymbol{\alpha}(\boldsymbol{r}) \in \boldsymbol{V_N} \boldsymbol{et}\ \boldsymbol{\beta}(\boldsymbol{r}) \in \boldsymbol{V_T V_N} \cup \boldsymbol{V_T}$
- Une grammaire est dite de type 2 (non contextuelle, algébrique) si et seulement si :
  - $\forall \boldsymbol{r} \in \boldsymbol{R}\ \boldsymbol{\alpha}(\boldsymbol{r}) \in \boldsymbol{V_N}$

Différentes types de grammaires

- Une grammaire est dite de type 1 (contextuelle, ou monotone) introduise une première restriction sur la forme des règles, en imposant que la partie droite de chaque production soit nécessairement plus longue que la partie gauche.
- Formellement :

$$- \forall \boldsymbol{r} \in \boldsymbol{R}\ |\boldsymbol{\alpha}(\boldsymbol{r})| \leq |\boldsymbol{\beta}(\boldsymbol{r})|$$

- Une grammaire sans restriction sur les règles est dite de type 0.
- Note :

– les langages de programmation usuels ont généralement une grammaire de type 2.

Type ?

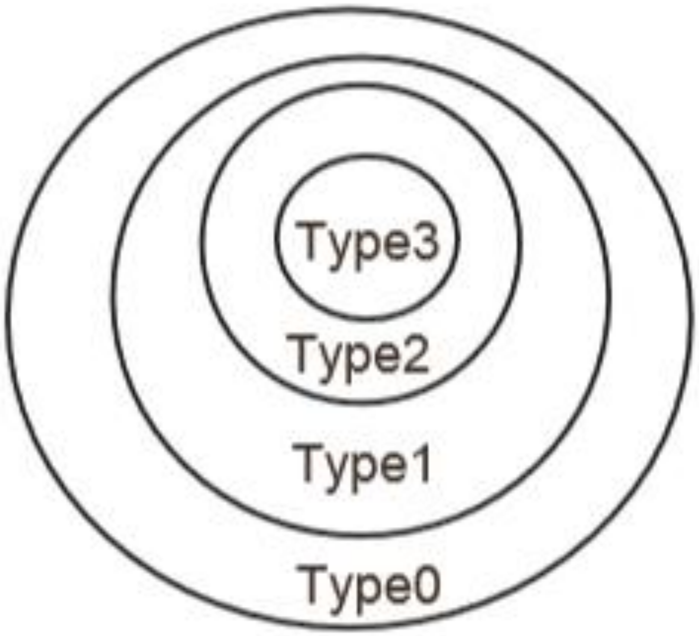
$$
\begin{aligned}
S &\rightarrow AB \mid CA \\
A &\rightarrow a \mid b \mid \varepsilon \\
B &\rightarrow BC \mid DB \\
C &\rightarrow E \mid \varepsilon \\
D &\rightarrow a \mid d \\
E &\rightarrow aB \mid c \mid d \mid \varepsilon
\end{aligned}
$$

1. G1<{a,b},{S,A,B},P,S>
où P : { S → AS / bB
A → a / ε
B → aB / a / ε }

2.G2<{a,b,c},{S,A,B,C, },P,S>
où P :
S → aSSB | ASC | a,
A → AAB | B | C,
B → a | ε,
C → AC | CB }

Différentes types de grammaires

- Il existe une relation d'inclusion entre les types de grammaires selon la figure suivante :



Type d’un langage

- Le type retenu pour une grammaire est le plus petit qui satisfait les conditions.
- Pour trouver la classe d’un langage on procède cependant comme suit :

– Chercher une grammaire de type 3 qui le génère, si elle existe, le langage est de type 3 (ou régulier)

– Sinon, chercher une grammaire de type 2 qui le génère, si elle existe, le langage est de type 2 (ou algébrique)

– Sinon, chercher une grammaire de type 1 qui le génère, si elle existe, le langage est de type 1 (ou contextuel)

– Sinon, le langage est de type 0.

Exercices

- Donnez, sans démonstration, les langages générés par les grammaires suivantes. Dites, a chaque fois de quelle type est la grammaire? :

– $G = (\{a\},\{S\},S,\{S \longrightarrow abS \mid b\});$

– $G = (\{a\},\{S\},S,\{S \longrightarrow aSa \mid \varepsilon\});$

– $G = (\{a\},\{S\},S,\{S \longrightarrow aSb \mid \varepsilon\});$

– $G = (\{a,b\},\{S\},S,\{S \longrightarrow aSa \mid bSb \mid \varepsilon\});$

- Donnez les grammaires qui génèrent les langages suivants :

– Les nombres binaires;

– Les mots sur $\{a,b\}$ qui contiennent le facteur aa