3. Grammaires

Cours 2 : Les grammaires

3. Grammaires : Backus-Naur Form

Au départ :

Backus et Naur introduisent **Backus-Naur Form** : Métalangage introduit pour ALGOL60

Basée sur la définition inductive.

Moyen simple et élégant de décrire toutes les phrases permises d'un langage (de programmation)

3.1. Backus-Naur Form

Exemple (en langage naturel) :
<phrase > : := <sujet ><verbe ><objet >
<sujet > : := <déterminant ><nom >
<objet > : := <déterminant ><nom >
<verbe > : := mangent | voient
<déterminant > : := les | des
<nom > : := chats | lions | souris | jambons

Le symbole | signifie ou bien
Exemple de phrase permise : les jambons mangent des lions

3.1. Backus-Naur Form

On va utiliser les règles décrites par cette 'grammaire' pour construire la phrase.

On apppliquera à chaque étape une règle :
c'est une étape de **dérivation**.

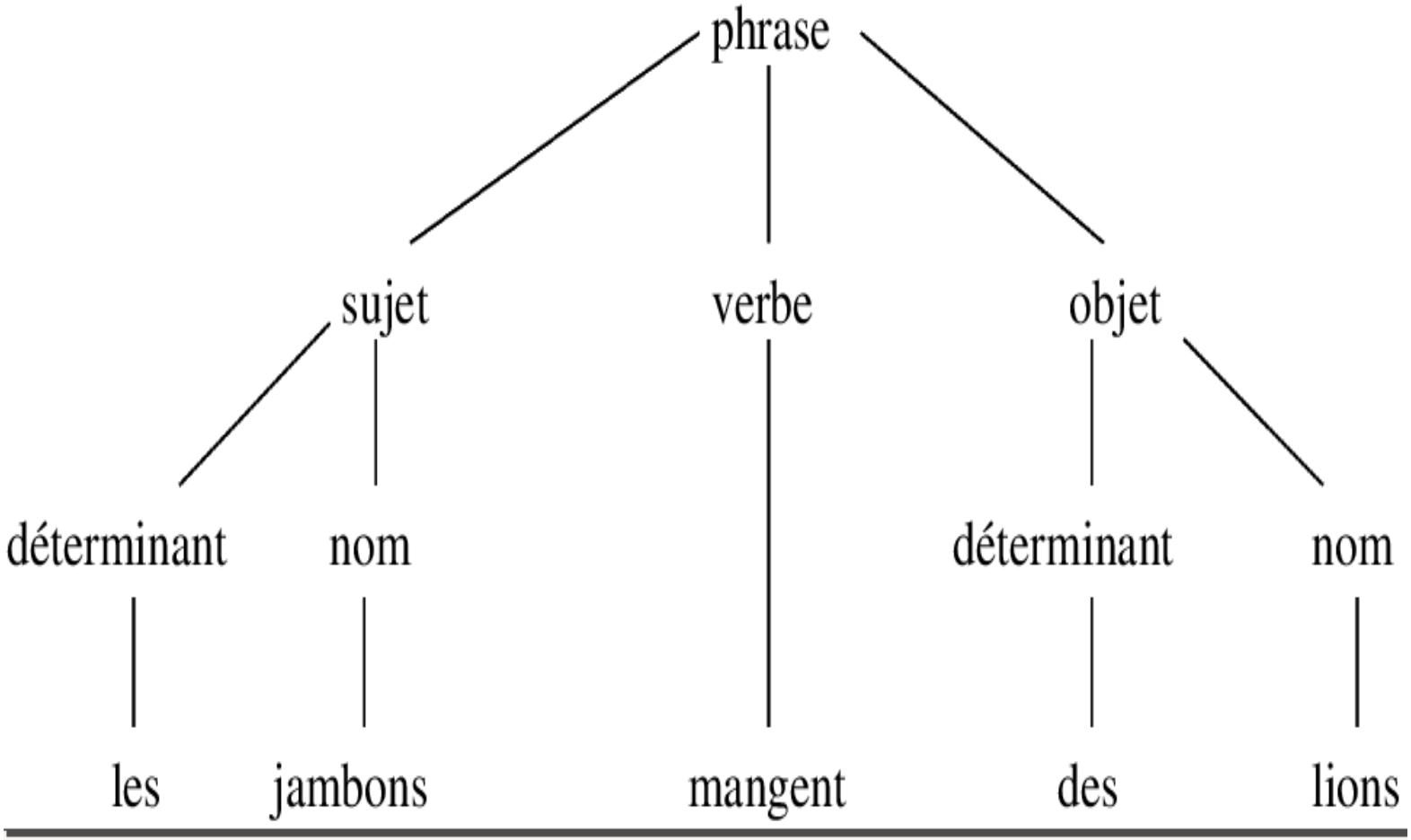
3.1. Backus-Naur Form

<phrase > : := <sujet ><verbe ><objet >
<sujet > : := <déterminant ><nom >
<objet > : := <déterminant ><nom >
<verbe > : := mangent | voient
<déterminant > : := les | des
<nom > : := chats | lions | souris | jambons

<**phrase** >
⇒ <**sujet** ><verbe ><objet >
⇒ <**déterminant** ><nom ><verbe ><objet >
⇒ les <**nom** ><verbe ><objet >
⇒ les jambons <**verbe** ><objet >
⇒ les jambons mangent <**objet** >
⇒ les jambons mangent <**déterminant** ><nom >
⇒ les jambons mangent des <**nom** >
⇒ les jambons mangent des lions

3.1. Backus-Naur Form

On peut aussi représenter la phrase par un **arbre syntaxique**.



Cet arbre constitue une **preuve que la phrase appartient bien au langage**.

3.1. Backus-Naur Form

Notation moderne : la forme est diffférente :
$\rightarrow$ remplace $:=$

3.1. Backus-Naur Form

Phrase $\rightarrow$ Sujet Verbe Objet
Sujet $\rightarrow$ Déterminant Nom
Objet $\rightarrow$ Déterminant Nom
Verbe $\rightarrow$ mangent | voient
Déterminant $\rightarrow$ les | des
Nom $\rightarrow$ chats | lions | souris| jambons

Ou, plus simplement :
P$\rightarrow$ SVO
S $\rightarrow$ DN
O $\rightarrow$ DN
V $\rightarrow$ mangent | voient
D $\rightarrow$ les | des
N $\rightarrow$ chats | lions | souris | jambons

3.1. Backus-Naur Form

Règles de grammaire :
P$\rightarrow$ SVO
S $\rightarrow$ DN
O $\rightarrow$ DN
V $\rightarrow$ mangent | voient
D $\rightarrow$ les | des
N $\rightarrow$ chats | lions | souris | jambons

Dérivation d'une phrase ('mot' du langage) :
les jambons mangent des lions
P$\Rightarrow$ SVO $\Rightarrow$ DNVO $\Rightarrow$ les NVO $\Rightarrow$ les jambons VO
$\Rightarrow$ les jambons mangent O $\Rightarrow$ les jambons mangent DN
$\Rightarrow$ les jambons mangent des N $\Rightarrow$ les jambons mangent des lions

3.1. Backus-Naur Form

Exemple 2 : le langage des expressions arithmétiques binaires simplissimes

Start $\rightarrow$ Exp

Exp $\rightarrow$ Exp + Exp

Exp $\rightarrow$ (Exp)

Exp $\rightarrow$ 0Exp

Exp $\rightarrow$ 1Exp

Exp $\rightarrow$ 0

Exp $\rightarrow$ 1

Ou, plus simplement :

S $\rightarrow$ E

E $\rightarrow$ E + E

E $\rightarrow$ (E)

E $\rightarrow$ 0E

E $\rightarrow$ 1E

E $\rightarrow$ 0

E $\rightarrow$ 1

3.1. Backus-Naur Form

On peut **factoriser** une grammaire en utilisant | :

$S \rightarrow E$

$E \rightarrow E + E$

$E \rightarrow (E)$

$E \rightarrow 0E$

$E \rightarrow 1E$

$E \rightarrow 0$

$E \rightarrow 1$

devient :

$S \rightarrow E$

$E \rightarrow E + E$

$E \rightarrow (E)$

$E \rightarrow 0E \mid 1E \mid 0 \mid 1$

ou même :

$S \rightarrow E$

$E \rightarrow E + E \mid (E) \mid 0E \mid 1E \mid 0 \mid 1$

3.1. Backus-Naur Form

$S \rightarrow E$
$E \rightarrow E + E \mid (E) \mid 0E \mid 1E \mid 0 \mid 1$
Exemple de dérivation possible :
$S \Rightarrow E \Rightarrow (E) \Rightarrow (E + E) \Rightarrow (0E + E) \Rightarrow (01 + E) \Rightarrow (01 + 1)$

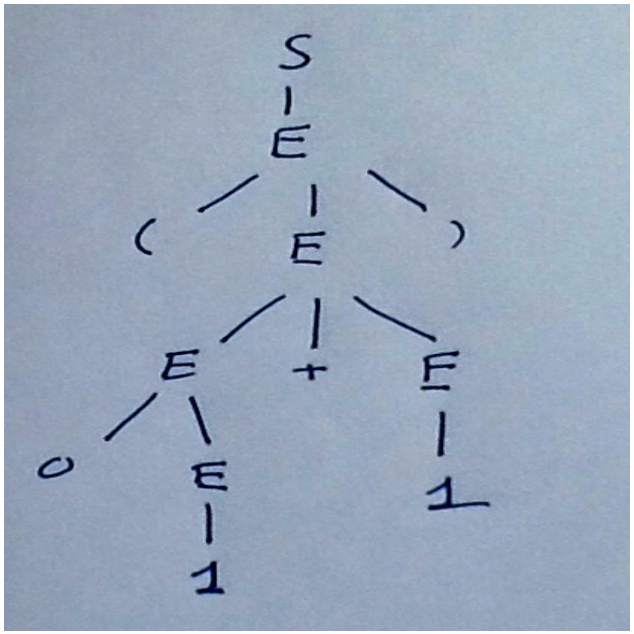
On a montré que l'expression $(01 + 1)$ appartient bien au langage défini par la 'grammaire'

3.1. Backus-Naur Form

$S \rightarrow E$

$E \rightarrow E + E \mid (E) \mid 0E \mid 1E \mid 0 \mid 1$

Arbre syntaxique correspondant au mot $(01 + 1)$:



3.2. Définition formelle des grammaires

Plus formellement :

3.2. Définition formelle des grammaires

Définition

Une grammaire est un **quadruplet** $G = (N, T, P, S)$
où :

- N *est l'ensemble des symboles* **non terminaux**
- T *est l'ensemble des symboles* **terminaux** *:*
  caractères de l'alphabet
- P *est un ensemble de* **règles de production**,
  de la forme $\alpha \to \beta$, *avec* $\alpha \in (N \cup T)^+, \beta \in (N \cup T)^*$
- $S =$ *symbole de départ appelé* **l'axiome**

3.2. Définition formelle des grammaires

Notations et remarques

- Pour caractères de N : on utilisera (habituellement) des majuscules.
- Pour caractères de T : on utilisera (habituellement) des minuscules.
- Pour les règles de P, nos règles seront de la forme $X \rightarrow \beta$, avec $X \in N$ et $\beta \in (N \cup T)^*$.
- L'axiome, noté S (habituellement), est la base de la définition inductive, et c'est la racine de tout arbre de dérivation valide

Souvent, par abus de langage, on décrit une grammaire seulement par ses règles !

3.2. Définition formelle des grammaires

Exemple :
S $\rightarrow$ E
E $\rightarrow$ E + E |(E)|0E| 1E | 0 | 1
N={ S,E }
T= $\{0, 1, (,), +\}$
P= $\{S \rightarrow$ E, E $\rightarrow$ E + E |(E)|0E| 1E | 0 | 1 }

3.2. Définition formelle des grammaires

On peut avoir des règles de production dont la partie droite est réduite à ϵ ; on appelera ces règles des ϵ-**productions**.

3.2. Définition formelle des grammaires

Exemple :
$S \rightarrow E$
$E \rightarrow E + E \mid (E) \mid 0E \mid 1E \mid \epsilon$
Une autre grammaire pour le langage des expressions arithmétiques binaires simplissimes.

3.2. Langage engendré par une grammaire

Définition

Pour une grammaire G, on note $L(G)$ le langage engendré par G : c'est l'ensemble des mots que l'on peut définir à partir de l'axiome de G en appliquant un nombre fini de fois des règles de G.

3.2. Définition formelle des grammaires

Exemple :
Grammaire G_1 :
$S \rightarrow aS$
$S \rightarrow bS$
$S \rightarrow a$
$S \rightarrow b$
$S \rightarrow \epsilon$
Quel langage décrit cette grammaire ?
Comment pourrait-on simplifier cette grammaire ?
Que se passe-t-il si on retire l'$\epsilon - production$?

3.2. Définition formelle des grammaires

Exemple 2 :
On veut traduire le langage sur $\Sigma = \{a,b\}$ où tous les mots sont de la forme $\omega = \alpha$ aa β

3.2. Définition formelle des grammaires

On va utiliser la grammaire G_2 :

$S \rightarrow AaaB$
$A \rightarrow aA|bA|\epsilon$
$B \rightarrow aB|bB|\epsilon$

3.2. Définition formelle des grammaires

Pour le mot : $\omega = abbaabb$

$S \rightarrow AaaB$

$A \rightarrow aA$

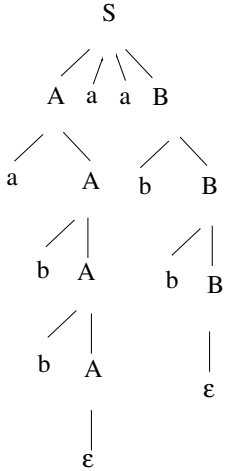
$A \rightarrow bA$

$A \rightarrow \epsilon$

$B \rightarrow aB$

$B \rightarrow bB$

$B \rightarrow \epsilon$



Grammaire sous forme factorisée

$S \rightarrow AaaB$
$A \rightarrow aA$
$A \rightarrow bA$
$A \rightarrow \epsilon$
$B \rightarrow aB$
$B \rightarrow bB$
$B \rightarrow \epsilon$

devient :
$S \rightarrow AaaB$
$A \rightarrow aA|bA|\epsilon$
$B \rightarrow aB|bB|\epsilon$

3.2. Définition formelle des grammaires

Les règles de production ont une numérotation implicite (de 1 à 7 ici).

$S \rightarrow AaaB$
$A \rightarrow aA|bA|\epsilon$
$B \rightarrow aB|bB|\epsilon$

$S \rightarrow AaaB$ (1)
$A \rightarrow aA$ (2) | bA(3) | ϵ(4)
B$\rightarrow$aB(5) | bB(6) | ϵ(7)

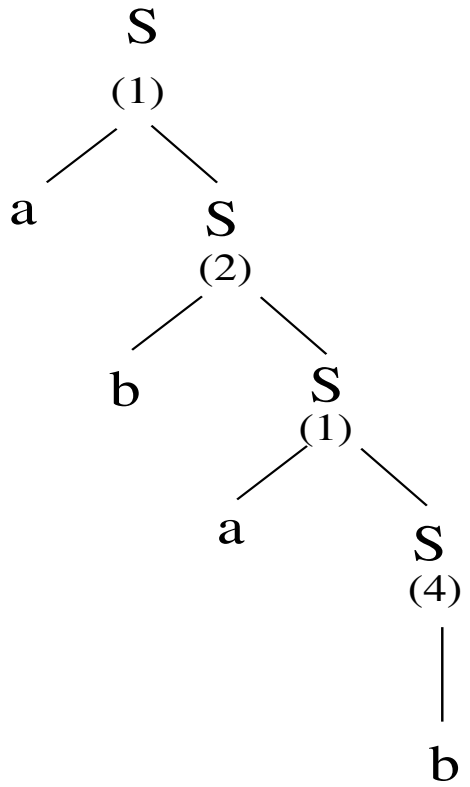
Arbre syntaxique

Définition

Un **arbre syntaxique** *est un arbre dont la racine est l'axiome (S), dont les noeuds internes sont étiquetés par des symboles de N, et dont les feuilles sont étiquetées par des symboles de T ou par le mot vide ϵ. Chaque noeud interne correspond à une règle de production.*

Exemple avec G_1 :
$S \rightarrow aS|bS|a|b|\epsilon$
mot $\omega = abab$

arbre syntaxique :



3.2. Définition formelle des grammaires

Pour un langage donné, il n'y a pas de grammaire unique ! ! ! !

Exemple : les deux grammaires suivantes décrivent le même langage :

$G_1 : S \rightarrow aS|bS|a|b|\epsilon$

$G_2 : S \rightarrow aS|bS|\epsilon$

3.2. Définition formelle des grammaires

Définition

On dit que deux grammaires G_1 et G_2 sont **équivalentes**, *noté $G_1 \sim G_2$, si elles engendrent le même langage, i.e. si $L(G_1) = L(G_2)$.*

3.3. Dérivations

Si $\alpha \rightarrow \beta$ est une production de P, on note $\gamma_1\alpha\gamma_2 \Rightarrow \gamma_1\beta\gamma_2$.
On dit qu'on a procédé à **une dérivation**. On dit que $\gamma_1\beta\gamma_2$ **se dérive de** $\gamma_1\alpha\gamma_2$.

Exemple :
$S \Rightarrow AaaA$
et
$AaaA \Rightarrow aAaaA$
sont des dérivations pour la grammaire G_2 ($S \rightarrow AaaA$; $A \rightarrow aA|bA|\epsilon$).

On peut étiqueter la dérivation par le numéro de la règle de production utilisée.

3.3. Dérivations

Pour un nombre fini de dérivations successives
$\gamma_1\alpha\gamma_2 \Rightarrow \gamma_1\beta\gamma_2 \Rightarrow \omega$,
on écrit :
$\gamma_1\alpha\gamma_2 \stackrel{*}{\Rightarrow} \omega$.

Exemples :
$S \Rightarrow AaaA \Rightarrow aAaaA$ ou $S \stackrel{*}{\Rightarrow} aAaaA$
$S \stackrel{*}{\Rightarrow} abbaabb$

Définition

On appelle **dérivation gauche** *une suite de dérivations obtenues en choisissant à chaque étape le symbole non terminal le plus à gauche.*
On définit de façon similaire la **dérivation droite**.

Exemple de dérivation gauche avec $G_2 : S \to AaaB$;
$A \to aA|bA|\epsilon$; $B \to aB|bB|\epsilon$:
$S \overset{(1)}{\Rightarrow} \underline{A}aaB \overset{(2)}{\Rightarrow} a\underline{A}aaB \overset{(4)}{\Rightarrow} aaa\underline{B} \overset{(6)}{\Rightarrow} aaab\underline{B} \overset{(7)}{\Rightarrow} aaab$

3.3. Dérivations

Définition

On appelle **langage engendré par une grammaire** $G = (N, T, P, S)$ *l'ensemble des mots* ω *de* T^* *tels que* $S \stackrel{*}{\Rightarrow} \omega$.
On le note $L(G)$.
On dit qu'un mot ω *est engendré par une grammaire* G *si* $\omega \in L(G)$.

3.4. Grammaires ambigües

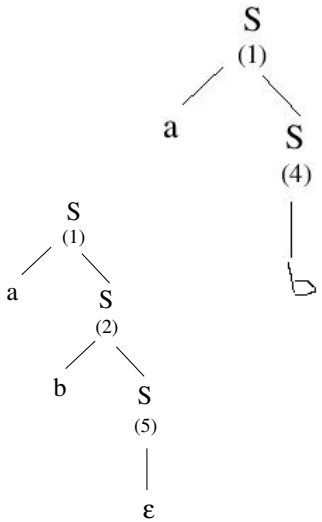
Définition

Une grammaire G est dite **ambiguë** *s'il existe un mot ω de $L(G)$ qui admet au moins deux arbres syntaxiques différents.*

Exemple : G_1 :

$G_1 : S \rightarrow aS|bS|a|b|\epsilon$

Le mot $\omega = ab$ admet deux arbres syntaxiques différents :



3.4. Grammaires ambigües

Définition

Un **langage** *est dit* **ambigü** *si* **toutes** *les grammaires qui l'engendrent sont ambiguës.*

Exemple : $G_1 : S \rightarrow aS|bS|a|b|\epsilon$
$G_2 : S \rightarrow aS|bS|\epsilon$
$L(G_1) = L(G_2)$; G_1 est ambiguë mais G_2 n'est pas ambiguë, donc $L(G_1) = L(G_2)$ n'est **pas** un langage ambigü.

3.5. Grammaires régulières

Définition

Une grammaire G est dite **régulière** *si toutes ses règles de production sont de la forme :*
$A \rightarrow \alpha B$, *avec :* $A \in N, \alpha \in T^*, B \in N$ *ou* $B = \epsilon$

Exemple : $G_1 : S \rightarrow aS|bS|a|b|\epsilon$

3.6. Décidabilité

Définition

Un problème est dit **indécidable** *si il n'existe pas (et il ne peut pas exister) d'algorithme générique pour le résoudre.*

3.6. Décidabilité

Les problèmes suivants sur les grammaires sont indécidables :

- Deux grammaires G_1 et G_2 sont-elles équivalentes ?
- Deux grammaires engendrent-elles des langages ayant un mot en commun ?
- Y a-t-il des mots qu'une grammaire n'engendre pas ?

3.7. Hiérarchie de Chomsky sur les grammaires

Type	Nom	Type de production
0	Langages récursivement énumérables	$X \rightarrow Y$ $X \in \mathbb{N}^+$, $Y \in (N \cup T)*$
1	Langages contextuels	$X \rightarrow Y$ $X \in \mathbb{N}^+$, $Y \in (N \cup T)^*$, $\lvert Y \rvert \geq \lvert X \rvert$
2	Langages context-free ou algébriques	$X \rightarrow Y$ $X \in \mathbb{N}$, $Y \in (N \cup T)^*$
3	Langages rationnels (réguliers)	$X \rightarrow Y$ $A \rightarrow \alpha B$, avec : $A \in N, \alpha \in T^*, B \in N$ ou $B = \epsilon$