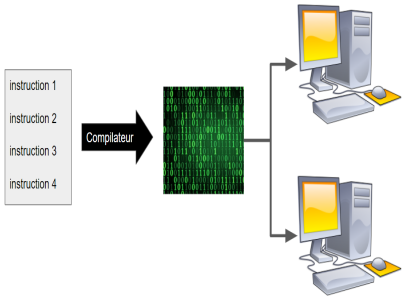


Analyse syntaxique descendante

Université Assane Seck
UFR Sciences et technologie
Département Informatique

Introduction

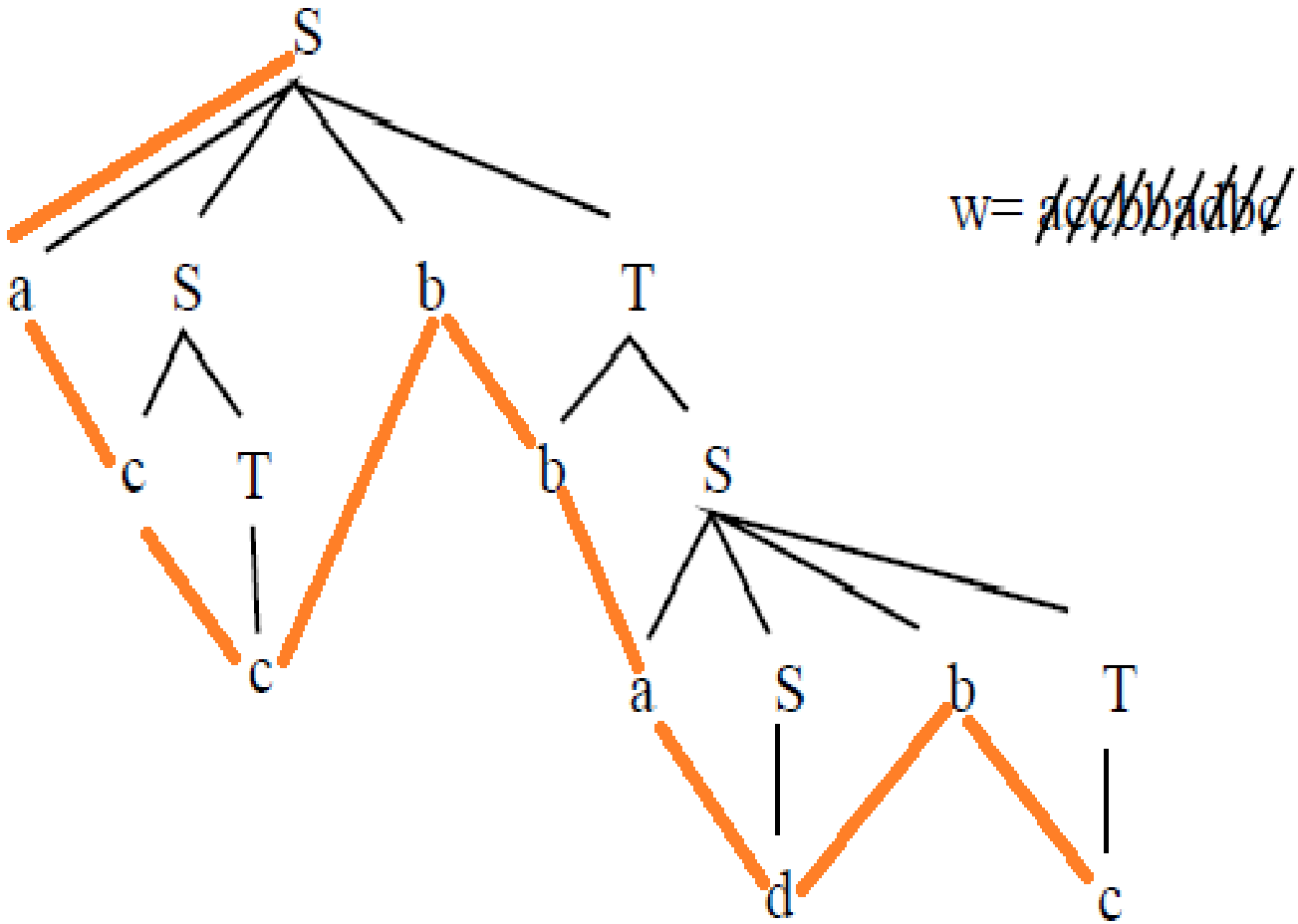
- L'analyseur syntaxique reçoit un mot sous forme de lexèmes
- Il crée un arbre de dérivation pour déterminer si le mot est syntaxiquement correct
- Il existe deux approches pour construire un arbre de dérivation
  - Une méthode descendante : du haut vers le bas
  - Une méthode ascendante: du bas vers le haut

Introduction

- Construire l'arbre du haut vers le bas
- La racine est l'axiome de départ
- Les feuilles sont les unités lexicales
- Exemple 1:

$$\begin{cases} S & \longrightarrow & aSbT|cT|d \\ T & \longrightarrow & aT|bS|c \end{cases}$$

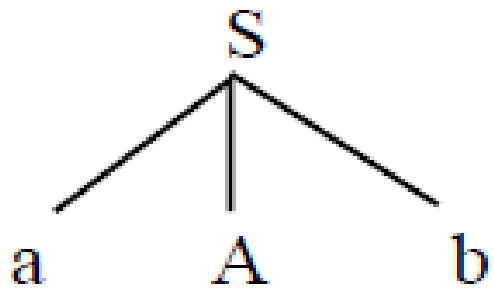
Considérons le mot *accbbadbc*


S
a
S
b
T
c
T
b
S
c
a
S
b
T
d
c
w= acbbadbc

Problématique de la méthode descendante

On considère la grammaire suivante: $\begin{cases} S & \longrightarrow & aAb \\ S & \longrightarrow & cd|c \end{cases}$

Supposons le mot $w = acb$



w=~~a~~cb

Après la lecture de a on a deux choix pour c:

- $A \longrightarrow cd$
- $A \longrightarrow c$

Pour le savoir, il faut aussi lire le caractère suivant b

$$\begin{cases} E \rightarrow TE' \\ E' \rightarrow +TE' | \varepsilon \\ T \rightarrow FT' \\ T' \rightarrow *FT' | \varepsilon \\ F \rightarrow (E) | \ \mathbf{nb} \end{cases}$$

avec le mot w=3*4+10*(5+11)

Pfff ... Alors là on ne voit plus rien du tout !!

Problématique de la méthode descendante

Conclusion

Ce qui serait pratique ça serait d'avoir une table qui nous dit: quand je lis un tel caractère et que je suis à dériver tel symbole non-terminal, alors j'applique telle règle et je ne me pose pas de question. Ça existe, et ça s'appelle une **table d'analyse**.

Pour construire une table d'analyse, on a besoin des ensembles *PREMIER* et *SUIVANT*

Outline

- On appelle $PREMIER(\alpha) \longrightarrow$ l'ensemble de tous les **terminaux** qui peuvent commencer une chaine qui se dérive de α
- On cherche toutes les lettres a tel $\alpha \xrightarrow{*} a\beta$ avec $\beta \in (V_N \cup VT)^*$
- $\epsilon \in PREMIER(\alpha)$ si et seulement il existe une dérivation $\alpha \xrightarrow{*} \epsilon$
- Exemple

$$\begin{cases} S & \longrightarrow & Ba \\ B & \longrightarrow & cP|bP|P|\epsilon \\ P & \longrightarrow & dS \end{cases}$$

$S \xrightarrow{*} a$ donc $a \in PREMIER(S)$, $S \xrightarrow{*} cPa$ donc $c \in PREMIER(S)$,
$S \xrightarrow{*} bPa$ donc $b \in PREMIER(S)$, $S \xrightarrow{*} dSa$ donc $d \in PREMIER(S)$
Donc $PREMIER(\alpha) = \{a, b, c, d\}$

Calcul des $PREMIER(\alpha)$ pour $\alpha \in (V_T \cup V_N)^*$

On a $\alpha \rightarrow Y_1 Y_2 ... Y_n$ $avec Y_i \in (V_N \cup VT)^*$
Si Y_1 est un symbole terminal alors ajouter Y_1 dans $PREMIER(\alpha)$
Sinon
//Y_1 est un non-terminal
ajouter les éléments de $PREMIER(Y_1)$ **sauf** ϵ dans $PREMIER(\alpha)$
Si $\epsilon \in PREMIER(Y_1)$ alors
Faire la meme chose avec Y_2
Si Y_2 est un symbole terminal alors ajouter Y_2 dans $PREMIER(\alpha)$
Sinon
//Y_2 est un non-terminal
ajouter les éléments de $PREMIER(Y_2)$ **sauf** ϵ dans $PREMIER(\alpha)$
Si $\epsilon \in PREMIER(Y_2)$ alors
Faire la même chose avec Y_3
Ainsi de suite jusqu'à Y_n

Exemples

$$\begin{cases} E & \longrightarrow & TE' \\ E' & \longrightarrow & +TE' \mid -TE' \mid \epsilon \\ T & \longrightarrow & FT' \\ T' & \longrightarrow & *FT' \mid /FT' \mid \epsilon \\ F & \longrightarrow & (E) \mid nb \end{cases}$$

$PREMIER(E) = PREMIER(T) = \{(, nb\}$

$PREMIER(E') = \{+, -, \epsilon\}$

$PREMIER(T) = PREMIER(F) = \{(, nb\}$

$PREMIER(T') = \{*, /, \epsilon\}$

Exemples

$$\begin{cases} E & \longrightarrow & TE' \\ E' & \longrightarrow & +TE' \mid -TE' \mid \epsilon \\ T & \longrightarrow & FT' \\ T' & \longrightarrow & *FT' \mid /FT' \mid \epsilon \\ F & \longrightarrow & (E) \mid nb \end{cases}$$

$PREMIER(E) = PREMIER(T) = \{(, nb\}$

$PREMIER(E') = \{+, -, \epsilon\}$

$PREMIER(T) = PREMIER(F) = \{(, nb\}$

$PREMIER(T') = \{*, /, \epsilon\}$

Exemple

$$\begin{cases} S & \longrightarrow & ABCe \\ A & \longrightarrow & aA|\epsilon \\ B & \longrightarrow & bB|cB|e \\ C & \longrightarrow & de|da|dA \end{cases}$$

$PREMIER(S) = \{a, b, c, d\}$

$PREMIER(A) = \{e, \epsilon\}$

$PREMIER(B) = \{b, c, \epsilon\}$

$PREMIER(C) = \{d\}$

Exemple

$$\begin{cases} S & \longrightarrow & ABCe \\ A & \longrightarrow & aA|\epsilon \\ B & \longrightarrow & bB|cB|e \\ C & \longrightarrow & de|da|dA \end{cases}$$

$PREMIER(S) = \{a, b, c, d\}$

$PREMIER(A) = \{e, \epsilon\}$

$PREMIER(B) = \{b, c, \epsilon\}$

$PREMIER(C) = \{d\}$

- on appelle $SUIVANT(A) \longrightarrow$ l'ensemble de tous les **terminaux** a qui peuvent apparaitre immédiatement à droite de A dans une dérivation: $S \xrightarrow{*} \alpha A a \beta$
- Exemple

$$\begin{cases} S & \longrightarrow & Sc|Ba \\ B & \longrightarrow & BPa|bPb|P|\epsilon \\ P & \longrightarrow & dS \end{cases}$$

$a, b, c, d \in SUIVANT(S)$ car il y'a les dérivations
$S \xrightarrow{*} Sc, S \xrightarrow{*} dSa, S \xrightarrow{*} bdSba, S \xrightarrow{*} dSdSaa$

- Ajouter un marqueur de fin de chaine (symbole \$ par exemple) à $SUIVANT(S)$
- Pour chaque forme $A \longrightarrow \alpha B \beta$ où β est un non-terminal, alors ajouter $PREMIER(\beta)$ à $SUIVANT(B)$ sauf ϵ
- Pour chaque production de la forme $A \longrightarrow \alpha B$ ajouter $SUIVANT(A)$ à $SUIVANT(B)$. En effet, toute régle $S \longrightarrow SA\beta$ se dérive en $S \longrightarrow S\alpha B\beta$. Donc, $SUIVANT(A) \in SUIVANT(B)$
- Pour chaque production de la forme $A \longrightarrow \alpha B \beta$ avec $\epsilon \in Premier(\beta)$ ajouter $SUIVANT(A)$ à $SUIVANT(B)$

Exemples

$$\begin{cases} E & \longrightarrow & TE' \\ E' & \longrightarrow & +TE' \mid -TE' \mid \epsilon \\ T & \longrightarrow & FT' \\ T' & \longrightarrow & *FT' \mid /FT' \mid \epsilon \\ F & \longrightarrow & (E) \mid nb \end{cases}$$

$SUIVANT(E) = \{\$,)\}$
$SUIVANT(E') = \{\$,)\}$
$SUIVANT(T) = \{+, -, \$,)\}$
$SUIVANT(T') = \{+, -, \$,)\}$
$SUIVANT(F) = \{*, /,), +, -\$,)\}$

Exemples

$$\begin{cases} E & \longrightarrow & TE' \\ E' & \longrightarrow & +TE' \mid -TE' \mid \epsilon \\ T & \longrightarrow & FT' \\ T' & \longrightarrow & *FT' \mid /FT' \mid \epsilon \\ F & \longrightarrow & (E) \mid nb \end{cases}$$

$SUIVANT(E) = \{\$,)\}$
$SUIVANT(E') = \{\$,)\}$
$SUIVANT(T) = \{+, -, \$,)\}$
$SUIVANT(T') = \{+, -, \$,)\}$
$SUIVANT(F) = \{*, /,), +, -\$,)\}$

Table d'analyse LL(1): Construction de la table

Une table d'analyse est un tableau à deux dimensions où les cases sont de la forme $[A, a]$ avec $a \in V_T$ et $A \in V_N \cup \$$

- Pour tout $a \in PREMIER(\alpha)$ (et $a \neq \epsilon$), rajouter la production $A \longrightarrow \alpha$ dans la case $M[A, a]$
- Si $\epsilon \in PREMIER(\alpha)$, alors chaque $b \in SUIVANT(A)$ ajouter $A \longrightarrow \alpha$ dans la case $M[A, b]$
- Chaque case $M[A, b]$ vide correspond à une erreur syntaxique

Table d'analyse

	nb	+	−	∗	/	(	)	$
E	$E \to TE'$					$E \to TE'$		
E'		$E' \to +TE'$	$E' \to -TE'$				$E' \to \varepsilon$	$E' \to \varepsilon$
T	$T \to FT'$					$T \to FT'$		
T'		$T' \to \varepsilon$	$T' \to \varepsilon$	$T' \to *FT'$	$T' \to /FT'$		$T' \to \varepsilon$	$T' \to \varepsilon$
F	$F \to$ nb					$F \to (E)$		

Outline

Analyseur syntaxique

On utilise une pile initialisée à Z_0, S, et un symbole de "look-ahead" p qui pointe sur le caractère courant du mot à analyser.
Etape d'analyse : On compare le symbole de haut de pile X avec p.

- 1er cas : X est un symbole terminal
  - si X = p, on dépile X et on décale p
  - sinon, REJECT

- 2e cas : X est un symbole non-terminal
  - si $[X, p]$ est une règle $X \longrightarrow \epsilon$, on dépile X (et on n'empile rien).
  - si $[X, p]$ est une règle $X \longrightarrow Y_1 Y_2 ... Y_n$, (où chaque Y_i est un caractère différent), on dépile X et on empile $Y_n Y_{n-1} ... Y_1$(sauf ϵ).
  - si $[X, p]$ est une case vide : REJECT.
- 3e cas : $X = Z_0$
  - si p=$, ACCEPT
  - sinon, REJECT

Exemple : on va analyser le mot $w = 2 + 3 * 4$

Grammaire LL(1)

Définition

On appelle **grammaire** $LL(1)$ une grammaire pour laquelle chaque case de la table d'analyse décrite contient au plus une règle de production.

- Le terme $LL(1)$ signifie:
  - L pour Left to right scanning: parcours de l'entrée de gauche à droite
  - L pour leftmost derivation: on utilise les dérivations gauches
  - (1): un seul élément de prévision est nécessaire à chaque étape
- Montrez que les grammaires suivantes ne sont pas $LL(1)$

$$\begin{cases} S & \longrightarrow & aAb \\ A & \longrightarrow & cd|c \end{cases} \qquad \begin{cases} S & \longrightarrow & aTbbb \\ T & \longrightarrow & Tb|\epsilon \end{cases}$$

Récursivité à gauche

Récursivité immédiate à gauche

Une grammaire est **immédiatement récursive à gauche** si elle contient un non-terminal A tel qu'il existe une production $A \longrightarrow A\alpha$ où αest une chaine quelconque

Elimination de la récursivité à gauche immédiate

- Remplacer toute règle de la forme $A \longrightarrow A\alpha|\beta$ par

$$\begin{cases} A & \longrightarrow & \beta A' \\ A' & \longrightarrow & \alpha A'|\epsilon \end{cases}$$

Récursivité à gauche

Récursivité immédiate à gauche

Une grammaire est **immédiatement récursive à gauche** si elle contient un non-terminal A tel qu'il existe une production $A \longrightarrow A\alpha$ où αest une chaine quelconque

Elimination de la récursivité à gauche immédiate

- Remplacer toute règle de la forme $A \longrightarrow A\alpha|\beta$ par

$$\begin{cases} A & \longrightarrow & \beta A' \\ A' & \longrightarrow & \alpha A'|\epsilon \end{cases}$$

Récursivité à gauche

$$\begin{cases} S & \longrightarrow & ScA|B \\ A & \longrightarrow & Aa|\epsilon \\ B & \longrightarrow & Bb|d|\epsilon \end{cases}$$

- Cette grammaire contient plusieurs récursivités à gauche immédiates
- Une grammaire non récursive équivalente est:

$$\begin{cases} S & \longrightarrow & BS' \\ S' & \longrightarrow & cAS'|\epsilon \\ A & \longrightarrow & A' \\ A' & \longrightarrow & aA'|\epsilon \end{cases}$$

Récursivité à gauche

Récursivité à gauche

Une grammaire est **récursive à gauche** si elle contient un non-terminal A tel qu'il existe une dérivation $A \xrightarrow{+} A\alpha$ où α est une chaine quelconque.

Exemple:

$$\begin{cases} S & \longrightarrow & Aa|b \\ A & \longrightarrow & Ac|Sd|c \end{cases}$$

Récursivité à gauche

Élimination de la récursivité gauche pour toute grammaire sans $\epsilon - production$

Ordonner tous les non-terminaux $A_1, A_2, ..., A_n$
Pour chaque i=1 à n faire
  pour j=1 à i-1 faire
    remplacer chaque production de la forme $A_i \longrightarrow A_j\alpha|\gamma$ où $A_j \longrightarrow \beta_1...\beta p$ par $A_i \longrightarrow \beta_1\alpha|...|\beta_p\alpha|\gamma$
    fin pour
éliminer les récursivités à gauche immédiates des productions A_i
fin pour

Récursivité à gauche

Exemple:

$$\begin{cases} S & \longrightarrow & Aa|b \\ A & \longrightarrow & Ac|Sd|BA|c \\ B & \longrightarrow & SSc|a \end{cases}$$

On ordonne S, A, B

$i = 1$ pas de récursivité immédiate dans $S \longrightarrow Aa|b$

$i = 2$ et $j = 1$ on obtient $A \longrightarrow Ac|Aad|bd|BA|c$

On élimine la récursivité immédiate:

$A \longrightarrow bdA^{'}|BAA^{'}|cA^{'}$

$A^{'} \longrightarrow cA^{'}|adA^{'}|\epsilon$

Récursivité à gauche

$i = 3$ on obtient $B \longrightarrow bdA^{'}aSc|BAA^{'}aSc|cA^{'}Asc|bSc|a$
On élimine la récursivité immédiate :
$B \longrightarrow bdA^{'}aScB^{'}|BAA^{'}aSc|cA^{'}AscB^{'}|bScA^{'}|aB^{'}$
$B \longrightarrow AA^{'}aScB^{'}|\epsilon$

- $S \longrightarrow Aa|b$
- $A \longrightarrow bdA^{'}|BAA^{'}|cA^{'}$
- $A^{'} \longrightarrow SSc|cA^{'}|\epsilon$
- $B \longrightarrow bdA^{'}aScB^{'}|cA^{'}AScB^{'}|bScA^{'}|aB^{'}$
- $B \longrightarrow AA^{'}aScB^{'}|\epsilon$

Factorisation à gauche

Factorisation à gauche

On dit qu'une grammaire algébrique G est non factorisée à gauche si elle contient un non terminal A dont l'ensemble des parties droites de productions est de la forme $A \longrightarrow \alpha\beta_1|\alpha\beta_2....|\gamma_1|....|\gamma_p$

Procédé de factorisation gauches

- Pour chaque non-terminal A, trouver un plus long préfixe α commun à plusieurs parties droites de production dont A est partie gauche
- $A \longrightarrow \alpha\beta_1|\alpha\beta_2....|\gamma_1|....|\gamma_p$ remplacé par
  - $A \longrightarrow \alpha A^{'}|\gamma_1|....|\gamma_p$
  - $A^{'} \longrightarrow \beta_1|\beta_2|...|\beta_n$
  - Recommencer jusqu'à ce qu'il n'y ait plus de préfixe propre commun.
  - On obtient une grammaire équivalente

Exemple de factorisation

On considère les règle de production suivantes:

$$\begin{cases} E & \longrightarrow & T + E | T \\ T & \longrightarrow & F * T | F \\ F & \longrightarrow & (E) | n \end{cases}$$

Réécriture de la règle :$T \longrightarrow T + E|T$. On obtient:

- $E \longrightarrow TE^{'}$
- $E^{'} \longrightarrow +E|\epsilon$

Exemple de factorisation

On considère les règle de production suivantes:

$$\begin{cases} E & \longrightarrow & T + E | T \\ T & \longrightarrow & F * T | F \\ F & \longrightarrow & (E) | n \end{cases}$$

Réécriture de la règle :$T \longrightarrow T + E|T$. On obtient:

- $E \longrightarrow TE^{'}$
- $E^{'} \longrightarrow +E|\epsilon$

Exemple de factorisation

Réécriture de la règle : $T \longrightarrow F * T|F$ on obtient :

- $T \longrightarrow FT^{'}$
- $T^{'} \longrightarrow *T|\epsilon$

On obtient globalement la forme :

- $E \longrightarrow TE^{'}$
- $E^{'} \longrightarrow +E|\epsilon$
- $T \longrightarrow FT^{'}$
- $T^{'} \longrightarrow *T|\epsilon$
- $F \longrightarrow (E)|n$