# Analyse LL(k)

Analyse descendante déterministe



# left to right scanning leftmost derivation

- Elle nécessite d'avoir k caractères pour savoir quelle règle appliquer.
- Il n'y a pas de retour arrière dans l'analyse du texte source ni dans la construction de l'arbre de dérivation.
- On dispose des symboles terminaux dans un tableau. Dans la pratique, on ne conserve que les éléments sur lesquels il peut être nécessaire de revenir.
- Un langage est généré par une grammaire LL(k) s'il existe une grammaire LL(k) qui le génère et qu'il n'existe pas de grammaire LL(k-1) qui peut le générer

# symbole directeur

Un symbole directeur est un symbole terminal qui permet le choix d'un alternant pour une notion donnée.

$A \rightarrow \alpha \ | \ \ldots$

SD(A, α ) :

C'est l'ensemble des symboles terminaux qui permettent le choix d'un alternant α pour la notion A.



# symbole directeur

$$SD(A,\alpha) = \begin{cases} PREMIER(\alpha) \cup SUIVANT(A) \ si \ \varepsilon \in PREMIER(\alpha) \\ PREMIER(\alpha) \ sinon \end{cases}$$

# symboles PREMIER

PREMIER ($\alpha$) : $\{a \,/\, \alpha \rightarrow^* a\,\beta$ , $\alpha, \beta \in V^*$, $a \in Vt\}$

Si $\alpha$ engendre la chaîne vide, il faut ajouter dans l'ensemble des symboles directeurs les symboles PREMIER de toutes les chaînes qui peuvent suivre A.



# Calcul des symboles PREMIER

**<u>Algorithme de calcul de premier($\alpha$)</u> :**

$\forall a \in V_T$
$F(a) = \{a \,/\, a \in V_T\}$ premier(a) =F(a)

$\forall A \in V_N$
$F_0(A) = \{a / a \in V_T \cup \{\varepsilon\}$ et $A \rightarrow a\alpha \in P\}$ $A \in V_N$, $\alpha \in V^*$ $Y_1 \in V_N \cup V_T$ (les non terminaux à droite)

i=1 ;
répéter
    $Fi(A) = \{a /\, A \rightarrow Y_1 Y_2 \ldots Y_n \in P$ et $a \in F_{i-1}(A) \cup F_{i-1}(Y_1) \oplus_1 F_{i-1}(Y_2) \oplus_1 \ldots . \oplus_1 F_{i-1}(Y_n)\}$
    $i \leftarrow i+1$
<u>jusqu'à</u> $F_i(A) = F_{i-1}(A)$

$\oplus_1$ : **premier de la concaténation**

# Calcul des symboles PREMIER

**$\oplus_1$ : premier de la concaténation**
**Soient $L_1$ et $L_2$**
**$L1 \oplus_1 L2$ = { premier(L1) si ε∉L1 sinon premier(L1)-{ε} ∪premier(L2)}**

**ex :**

**L1 = {a,aa,ab}**
**L2 = {x,xy,yy}**
**L3 = {a,aa,ab,ε}**

**Concaténation des mots de L1 et L2** = {ax,axy, ayy, aax, aaxy, aayy, aba, abxy, abyy}
**$L1 \oplus_1 L2$ = {a}**

**Concaténation des mots de L3 et L2** ={ax,axy, ayy, aax, aaxy, aayy, aba, abxy, abyy, x, xy, yy}
**$L3 \oplus_1 L2$ = {a,x,y} → toutes les 1ères lettres de chaque chaîne après concaténation**



# Calcul des symboles PREMIER

## calcul rapide Premier :

**A→aα : ajouter a à premier(A)**

**A→Bα : ajouter premier(B) à premier(A)**
**et si ε∈premier(B) alors ajouter premier(α) à premier(A)**

# symboles SUIVANT

SUIVANT(A) =

$\{ a \,/\, S \Rightarrow \beta A\, a\, \mu\ ,\quad A \in V_N,$

$\beta, \mu \in V^*,$

$a \in V_T\}$



## Algorithme de calcul du suivant

Initialisation :

$F0(A)$= \$ $si\ A$=$S$ $(axiome)$

= $\emptyset\ sinon$

F1 : $B \rightarrow \cdots A\alpha\ avec\ \alpha \neq \varepsilon$

$F1(\ A)$ = $F0\ (A)$ ∪($PREMIER(\alpha)$∩$VT$)

Fi : $B \rightarrow \cdots A\alpha$ ($avec\ \alpha \rightarrow \varepsilon$ ou $\varepsilon \in PREMIER(\alpha)$

$i$=2

Répéter

$Fi\ (A)$ = $Fi{-}1\ A$ ∪$Fi{-}1\ B$

$i \leftarrow i$+1

Jusque $Fi\ (A)$ = $Fi{-}1(A)$

$SUIVANT(\ A)$ = $Fi(A)$

# Conditions LL(1)

Après avoir déterminé les symboles directeurs, on peut démontrer si une grammaire est LL(k).

Condition nécessaire et suffisante pour qu'une grammaire soit LL(1) :

Pour chaque $A \in V_n$, $A \rightarrow \alpha_1 | \alpha_2 | \dots | \alpha_n$

Il faut que les symboles directeurs associés aux différents alternants $\alpha_i$ soient disjoints



# Conditions LL(1)

- Pour une règle de grammaire du type

  $A \rightarrow \alpha_1 | \alpha_2 | \dots | \alpha_n$

  2 conditions :

  - Pour tout i et j tels que i≠j

    $Premier(\alpha i) \cap Premier(\alpha j) = \emptyset$

  - si $\alpha i \rightarrow \varepsilon$, $\forall j \neq i$, $Premier(\alpha j) \cap Suivant(A) = \emptyset$

    (condition $\alpha i \rightarrow \varepsilon$ = ε appartient Premier($\alpha$i))

# Table d'Analyse

$$T: (V_N \cup V_T) \times V_T \rightarrow \{n° \text{ règle de } G, \text{dépilé}, \text{erreur}, \text{succès}\}$$

$$T(A,a) = \begin{cases} n° \text{ règle } A \rightarrow \alpha \text{ si } a \in SD(A,\alpha) \\ \text{erreur sinon} \end{cases}$$

$$T(a,b) = \begin{cases} \text{dépiler si } a = b \\ \text{succès si } a = b = \$ \\ \text{erreur sinon} \end{cases}$$

$A \in V_N, a, b \in V_T, \alpha \in (V_N \cup V_T)^*$



# Analyse

Les analyseurs associés à ce type de grammaire sont caractérisés par des automates à pile.
Il utilisent :
- un tampon contenant la chaîne d'entré terminée par $
- une pile pour les symboles de la grammaire
- une table d'analyse qui permet de choisir la règle à utiliser en fonction du symbole en entrée et de la règle au sommet de la pile.

# algorithme d'Analyse

2. Analyseur LL(1) : $< \alpha, A\beta\$, \delta >$

   $\alpha$ : *chaîne d'entrée*

   $A$ : *prochaine notion à développer*

   $A\beta$ : *développement des règles*

   $\delta$ : *chaîne formée des n° de règles de G utilisées*

   a. Initialisation$< w\$, S\$, \varepsilon >$
   b. $w = a_1 a_2 \dots a_n$
   c. $< a_i\alpha, A\beta\$, \delta > := < a_i\alpha, a_i\gamma\beta\$, \delta k >$
      si $\exists\ A \rightarrow a_i\gamma\ (k) \in P$ alors

      $$T(A, a_i) = k$$
   d. $< a_i\alpha, a_i\beta\$, \delta > := < \alpha, \beta\$, \delta >$
   e. $T(a_i, a_i) = dépiler$
   f. $< \$, \$, \delta > := succès$
      T($,$) = succès
   g. $< a_i\alpha, a_j\beta\$, S > := erreur\ si\ a_i \neq a_j$
      T(a,b) = erreur si a $\neq$b