Titre de la note 23/11/2021

Chapitre III : Analyse syntaxique.

Rappel : Etapes de compilation (Analyse) classes

prog. source ⟶ Analyseur lexical ⟶ Unités lexicales + erreurs lexicales ⟶ Analyseur syntaxique ⟶ arbre syntaxique + erreur syntaxique

(C, C++, Java)

= ensemble de mots (lexèmes)

tâche d'un analyseur syntaxique :

Reconnaître les phrases / instructions → correctes / erreurs

Exemple :

1) int a ; // inst correcte
2) float x // inst non correcte (oublier le ;)
3) x = 4 ; // correcte
4) x + 1 = 6 ; // incorrecte

affectation

Variable = Exp

~~Exp = variable / Exp~~

Comment : Pour reconnaître une phrase/instruction pour un langage (~~pg~~, naturel) on doit définir une grammaire pour ce langage.

Analyse

Déf : grammaire = ensemble de règles qui permet de reconnaître les phrases/instructions d'un langage et aussi de produire/générer des phrases/instruction de ce langage.

génération

Ist V = Exp

$x = x + 1$

Exeple 1 : Extrait du grammaire Français

R1 : Une phrase est composée d'un groupe nominal (GN) suivi du verbe voir au présent ensuite d'un autre groupe nominal

R1 : PHRASE $\xrightarrow[\text{peut s'écrire}]{}$ GN voit GN

R2 : Un groupe nominal peut être composé d'un determinant (DET) suivi d'un nom (NOM)

R2 GN $\longrightarrow$ DET NOM

R3 : Exeple de determinant = { le , un }

R3 : DET $\longrightarrow$ le || un

ou

R4 : Exeple de nom = { chat, chien }

NOM $\longrightarrow$ chat | chien

Question : Donner les étapes de génération / dérivation pour obtenir la phrase : le chat voit un chien.

PHRASE $\xrightarrow{R1}$ GN voit GN $\xrightarrow{R2}$ DET NOM voit GN $\xrightarrow{R3}$ le NOM voit GN

$\xrightarrow{R4}$ le chat voit GN $\xrightarrow{R2}$ le chat voit DET NOM $\xrightarrow{R3}$ le chat voit un NOM

$\xrightarrow{R4}$ $\underbrace{\text{le}}_{L1}$ $\underbrace{\text{chat}}_{L2}$ $\underbrace{\text{voit}}_{L3}$ $\underbrace{\text{un}}_{L4}$ chien ⟸ Dérivation / génération gauche

mot = intra

Définition : Une grammaire $G = (V_t, V_n, P, S)$ avec

1) V_t : ensemble des terminaux (lexèmes). $V_t = \{$voit, le, un, chat, chien$\}$

2) V_n : ensemble des non-terminaux (unité lexicale) $V_n = \{$PHRASE, NOM, DET, GN$\}$

3) P : règles de production $P = \{R1, R2, R3, R4\}$

4) $S \in V_n$: Axiome $S =$ PHRASE.

Arbre syntaxique

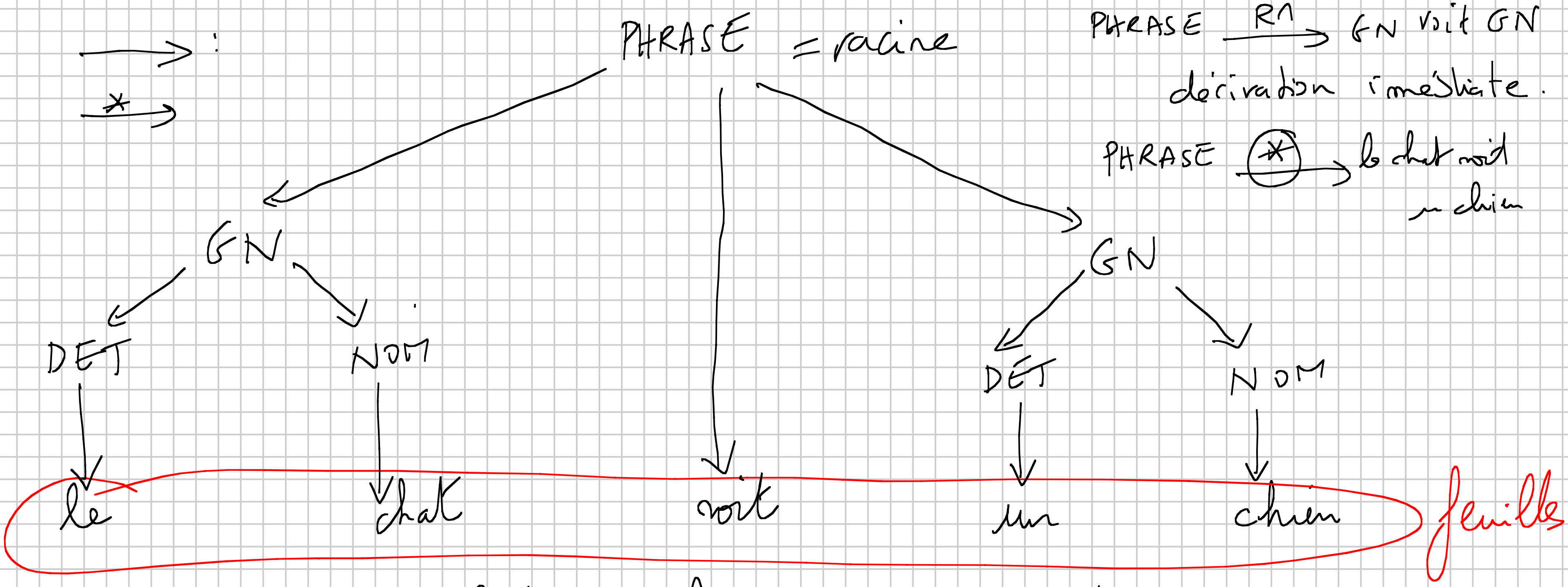
$\longrightarrow$:

$\xrightarrow{*}$

PHRASE $\xrightarrow{R1}$ GN voit GN

dérivation immédiate.

PHRASE $\xrightarrow{*}$ le chat voit un chien



PHRASE2 : voit le chat un chien : erreur syntaxique.

Arbre syntaxique = Arbre de dérivation = représentation graphique d'une séquence de lexèmes ($\in V_t$). Les nœuds représentent les non-terminaux ($\in V_n$). Les arcs représentent les étapes de dérivation

Exemple 2 : Considérons la grammaire définie par :

R1 : $S \longrightarrow aSbT \mid cT \mid d$

R2 : $T \longrightarrow aT \mid bS \mid c$

avec $V_t = \{a, b, c, d\}$

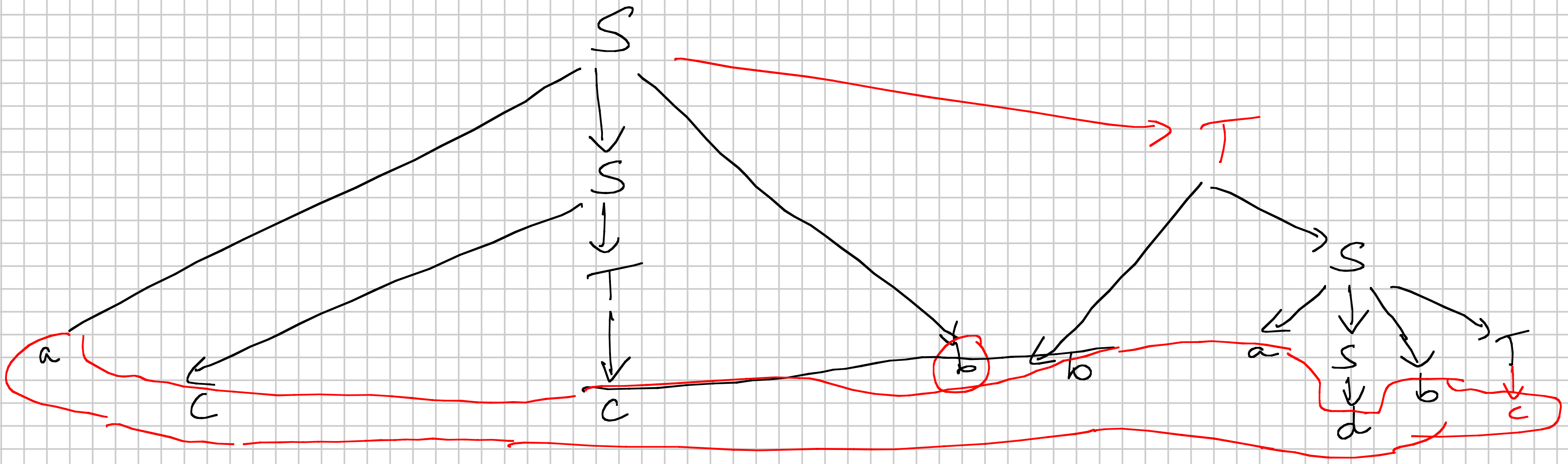
$V_n = \{S, T\}$

Axiome : S

Question : Donner l'arbre syntaxique pour le mot $w = accbbadbc$

lexème

instruction = mot ≡ if (x < y)


S
S
T
T
S
a
c
c
b
b
a
S
b
T
d
c

Résumé : Analyse syntaxique

Donnés : 1) phrase = ensemble de lexèmes

2) $G = (V_t, V_n, P, S)$ grammaire

Sortie : Arbre syntaxique

\+ erreurs syntaxique.

Exemple 2 : Soit la grammaire G définie par

R1 : $S \longrightarrow aB$

R2 $B \longrightarrow bc \mid bB$

$V_t = \{a, b, c\}$

$V_n = \{S, B\}$

$P = \{R1, R2\}$

Axiome $= S$.

LL(1)

LL(2)

LL(K)

Question : Est-ce que le mot $w = abc$ est accepté par G ?

$S \longrightarrow aB \longrightarrow abc$

(lire le prochain lexème pour décider)

Si on choisit $B \rightarrow bB$: $S \longrightarrow aB \longrightarrow abB$: echec

Rq : En général, on deux méthodes d'analyse syntaxique

1. Méthode d'analyse descendante (TOP DOWN) : On construit l'arbre syntaxique de haut en bas : en partant de la racine vers les feuilles.

2. Méthode d'analyse ascendante : On construit l'arbre syntaxique en partant des feuilles vers la racine.

Méthode utilisée : Analyse prédictive : méthode descendante ds laquelle, on peut tjs choisir une règle unique en se basant sur le prochain lexème et sans effectuer aucun retour arrière

L'analyse prédictive nécessite le calcul de certaine fonctions :
Nullable, Premier (First), Suivant (Follow).

La fct Nullable :

$G = (V_t, V_n, P, S)$

α : terminal
$\alpha = a$
α : non term
$\alpha = S$
$\alpha = aS$

Def : $\alpha \in (V_t \cup V_n)^*$ une forme de G, on dit que α est Nullable et on écrit $Nullable(\alpha) = Vrai$

$\Longleftrightarrow$ on peut dériver le mot vide ε à partir de α.

$\Longleftrightarrow \alpha \xrightarrow{*} \varepsilon$

Exemple :
R1 : $R \rightarrow bR \mid \varepsilon$
R2 : $T \rightarrow R \mid aTc$

$R \rightarrow \varepsilon$

Nullable(R) = True

$T \xrightarrow{*} \varepsilon$ Nullable(T) = True $R \xrightarrow{*} \varepsilon$

$T \rightarrow R \rightarrow \varepsilon$

Règles de calcul de Nullable :

1) Nullable(ε) = True $\qquad \varepsilon \to \varepsilon$

2) $a \in V_t$, Nullable(a) = False $\qquad a \not\to \varepsilon$

3) $\alpha, \beta \in (V_t \cup V_n)^*$ $\quad$ Nullable($\alpha . \beta$) = Nullable(α) AND Nullable(β)

4) $\forall X \in V_n$ avec $X \longrightarrow \alpha_1 | \alpha_2 | \cdots | \alpha_n$ $\quad \alpha_i \in (V_t \cup V_n)^*$

Nullable(X) = Nullable(α_1) OR Nullable(α_2) OR ... Nullable(α_n)

<u>Exemple:</u> Soit la grammaire G définie par

$$T \longrightarrow R \mid aTc$$
$$R \longrightarrow bR \mid \varepsilon$$

<u>Question:</u> Calculer Nullable(T), Nullable(R).

Exemple :

$G = (V_t, V_n, P, S)$ avec

$\longrightarrow$ dérivation immédiate

$\overset{*}{\longrightarrow}$ dérivation multiple

$V_t = \{a, b, d, e\}$

$V_n = \{S, A, B, D\}$

P :

$S \longrightarrow AB \mid Da$

$A \longrightarrow aAb \mid \varepsilon$

$B \longrightarrow bB \mid \varepsilon$

$D \longrightarrow dD \mid e$

S : Axiome

Supposons AB dérive des mots qui commencent par a ou b

$AB \overset{*}{\Longrightarrow} au$

$u \in V_t^*$

$AB \xRightarrow{*} bu$

Supposons que Da dérive des mots qui commencent par d ou e

$Da \xrightarrow{*} dv$, $Da \xrightarrow{*} ev$

si tête $\in \{a, b\}$ $\longrightarrow$ $S \longrightarrow AB$ $\Rightarrow$ Premier $(AB) = \{a, b\}$

tête $\in \{d, e\}$ $\longrightarrow$ $S \longrightarrow Da$ Premier $(Da) = \{d, e\}$

$G = (V_T, V_n, P, S)$ avec

$V_t = \{a, b, d, e\}$

$V_n = \{S, A, B, D\}$

Ex1 : $S \longrightarrow AB \mid Da$

$A \longrightarrow aAb \mid \varepsilon$

$B \longrightarrow bB \mid \varepsilon$

$D \longrightarrow dD \mid e$

Pour remplir la table d'analyse $M[a, X]$

$M[a, X]$ = règle

il faut calculer la fct Nullable, premier, suivant.

Déf $\text{Nullable}(\alpha) = \begin{cases} \text{Vrai} & \alpha \overset{*}{\Rightarrow} \varepsilon \\ \text{faux} & \text{sinon} \end{cases}$

Fonction Nullable :

1) $\text{Nullable}(\varepsilon) = \text{Vrai}$

2) $\forall a \in V_t$ (terminal) : $\text{Nullable}(a) = \text{Faux}$

3) $\text{Nullable}(\alpha\beta) = \text{Nullable}(\alpha)$ AND $\text{Nullable}(\beta)$ $\quad \alpha, \beta \in (V_t \cup V_n)^+$

4) Si $X \longrightarrow \alpha_1 \mid \alpha_2 \mid \dots \mid \alpha_n$ $(\alpha_i \in (V_t \cup V_n)^+)$ alors

$\text{Nullable}(X) = \text{Nullable}(\alpha_1)$ OR $\text{Nullable}(\alpha_2) \dots$ OR $\text{Nullable}(\alpha_n)$

$\text{Nullable}(A) = \text{Nullable}(aAb)$ or $\text{Nullable}(\varepsilon)$

$= \text{Vrai}$

$\text{Nullable}(S) = \text{Nullable}(AB)$ or $\text{Nullable}(Da)$

$\text{Nullable}(B) = \text{vrai}$

$\text{Nullable}(D) = \text{Nullable}(dD)$ or $\text{Null}(e)$

$\text{Nullable}(dD) \overset{3}{=} \underbrace{\text{Nullable}(d)}_{\text{Faux (2)}} \text{ AND } \text{Nullable}(D) = \text{Faux}$

$\text{Nullable}(e) = \text{Faux}$

$\Longrightarrow \boxed{\text{Nullable}(D) = \text{Faux}}$

$\text{Nullable}(AB) \overset{3}{=} \overset{\text{Vrai}}{\text{Nullable}(A)} \text{ AND } \overset{\text{Vrai}}{\text{Nullable}(B)}$

$\boxed{\text{Nullable}(AB) = \text{Vrai}}$

$\Longrightarrow \boxed{\text{Nullable}(S) = \text{Vrai}}$

Définition : Premier (α) $\quad \alpha \in (V_t \cup V_n)^*$

$\alpha = a$

$\alpha = X$

$\alpha = Xa$

$\alpha = abXY$

Premier(α) contient l'ensemble des terminaux de V_t susceptible de commencer un mot de V_t^* dérivé de α.

$$\text{Premier}(\alpha) = \left\{ a \in V_t \;/\; \alpha \xrightarrow{*} \underbrace{a\beta}_{w} \quad \beta \in (V_t \cup V_n)^* \right\}$$

$x \in Premier(\alpha) \Longrightarrow \alpha \xrightarrow{*} x\beta$

Fonction Premier : $\alpha \in (V_t \cup V_n)^*$:

1) $Premier(\alpha = \varepsilon) = \phi$
2) $\alpha = a,\ a \in V_t,\ Premier(\alpha) = \{a\}$
3) $\alpha = a\beta,\ a \in V_t,\ \beta \in (V_t \cup V_n)^*$. $Premier(\alpha) = \{a\}$.
4) $\alpha = X,\ X \in V_n \quad X \longrightarrow \alpha_1 | \alpha_2 | \dots | \alpha_n$: $Premier(\alpha) = \bigcup Premier(\alpha_i)$
5) $\alpha = X\beta \quad X \in V_n,\ \beta \in (V_t \cup V_n)^*$
   - 5.1) $Nullable(X) = Vrai$: $Premier(\alpha) = Premier(X) \cup Premier(\beta)$
   - 5.2) $Nullable(X) = Faux$: $Premier(\alpha) = Premier(X)$

$S \to AB \mid Da$

$A \to aAb \mid \varepsilon$

$B \to bB \mid \varepsilon$

$D \to dD \mid e$

$Premier(\varepsilon) = \emptyset$

$Premier(S) \overset{4}{=} Premier(AB) \cup Premier(Da)$

$Premier(A) \overset{4}{=} Premier(aAb) \cup Premier(\varepsilon)$

$Premier(A) = \{a\}$ ε

$Premier(B) = \{b\}$, $Premier(D) = \{d, e\}$

Fonction Premier : $\alpha \in (V_t \cup V_n)^*$

1) $Premier(\alpha = \varepsilon) = \emptyset$

2) $\alpha = a,\ a \in V_t$; $Premier(\alpha) = \{a\}$

3) $\alpha = a\beta,\ a \in V_t,\ \beta \in (V_t \cup V_n)^*$; $Premier(\alpha) = \{a\}$

4) $\alpha = X,\ X \in V_n$ $X \to \alpha_1 \mid \alpha_2 \mid \ldots \mid \alpha_n$; $Premier(\alpha) = \bigcup Premier(\alpha_i)$

5) $\alpha = X\beta$ $X \in V_n,\ \beta \in (V_t \cup V_n)^*$

5.1) $Nullable(X) = Vrai$: $Premier(\alpha) = Premier(X) \cup Premier(\beta)$

5.2) $Nullable(X) = Faux$: $Premier(\alpha) = Premier(X)$

Premier (AB) $\stackrel{5.1}{=}$ Premier (A) $\cup$ Premier (B) = {a} $\cup$ {b} = {a, b}

(Nullable(A) = Vrai)

Premier (Da) $\stackrel{5.2}{=}$ Premier (D) = {d, e}

(Nullable(D) = Faux)

le lexème courant : s

$A \rightarrow \alpha_1$

$A \rightarrow \alpha_2$

$s \in$ Premier(α_1) : $A \rightarrow \alpha_1$

$s \in$ Premier(α_2) : $A \rightarrow \alpha_2$

Explication :

Premier (A) $\stackrel{4}{=}$ Premier ($a\underbrace{Ab}_{\beta}$) $\cup$ Premier (ε)

Premier ($a\underbrace{Ab}_{\beta}$) = Premier ($a\beta$) = {a}.

$A \rightarrow \alpha_1 \rightarrow L \in$ Premier(α_1)

$A \rightarrow \alpha_2 \rightarrow L \in$ Premier(α_2)

Rappel :

$S \longrightarrow AB \mid Da$
$A \longrightarrow aAb \mid \varepsilon$
$B \longrightarrow bB \mid \varepsilon$
$D \longrightarrow dD \mid e$

$\ell = \{R1, R2, R3, R4\}$

$A \longrightarrow \underbrace{aAb}_{R1} \mid \underbrace{\varepsilon}_{R2}$

$\neq$

Pour choisir entre $S \longrightarrow AB$ et $S \longrightarrow Da$

si lexème courant $\in \{a, b\}$: $S \longrightarrow AB$

si " $\in \{d, e\}$: $S \longrightarrow Da$

$A \longrightarrow \varepsilon$

R1 $A \longrightarrow aAb$

R2 $A \longrightarrow \varepsilon$

Premier$(\varepsilon) = \phi$

Question :

Comment choisir entre $A \longrightarrow aAb$ et $A \longrightarrow \varepsilon$ (Premier$(\varepsilon) = \phi$)

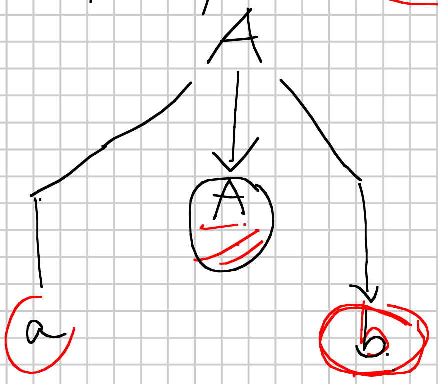
et si le lexème courant est # (marqueur fin de la phrase) (# $\notin$ Premier(AB) $\cup$ Premier(Da))

$\Longrightarrow$ L'ensemble (Premier ne suffit pas

Qd appliquer $A \longrightarrow \varepsilon$?

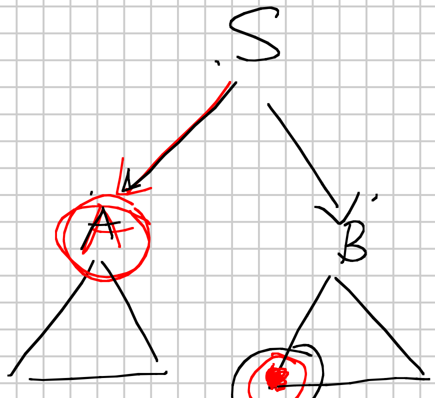
Réponse : Qd le lexème courant correspond à un terminal qui peut suivre A

$A \longrightarrow aAb$



$b \in$ Suivant(A)

$S \longrightarrow AB$



Suivant(A) contient Premier(B)

Calcul de Suivant(X) $X \in V_n$

$G = (V_t, V_n, P, S)$.

$V_t^* = \{\varepsilon\}$

$V_t^+ = \{$

Déf

Suivant(X) contient l'ensemble des terminaux de V_t susceptibles de suivre X ds un mot de V_t^+ dérivé de l'axiome S

phrase à analyser : on ajoute un marqueur de fin de la phrase #

Formellement

$$\text{Suivant}(X) = \{a \in V_t \,/\, S \xrightarrow{*} \beta X a \delta,\ \beta, \delta \in (V_t \cup V_n)^*\}.$$

$a\delta$: Mot $\in V_t^+$

$S \xrightarrow{*} w\#$

Rqs.

1) Suivant $(X = \varepsilon) = \emptyset$

2) Par convention $\# \in$ Suivant(S) (S : axiome)

Algorithme de Calcul Suivant :

Règle

Fonction Suivant : # : marqueur fin de phrase. $G = (V_t, V_n, P, S)$

1). $\# \in$ Suivant(axiome $= S$)

Soit $P_X \subset P$: ensemble des règles p dans lesquelles X apparaît en partie droite.

$$\text{Suivant}(X) \stackrel{Def}{=} \bigcup_{p \in P_X} \text{Suivant}_p(X)$$

avec $\text{Suivant}_p(X) =$

1) $p = Y \rightarrow \alpha X$: Suivant(Y)
2) $p = Y \rightarrow \alpha X \beta$ et Nullable(β) = Faux : Premier(β)
3) $p = Y \rightarrow \alpha X \beta$ et Nullable(β) = Vrai : Premier(β) $\cup$ Suivant(Y)

$S \rightarrow AB \mid Da$
$A \rightarrow aAb \mid \varepsilon$
$B \rightarrow bB \mid \varepsilon$
$D \rightarrow dD \mid e$

$P_S =$ vide

$P_A = \{S \rightarrow AB,\ A \rightarrow aAb\}$

$P_B = \{S \rightarrow AB,\ B \rightarrow bB\}$

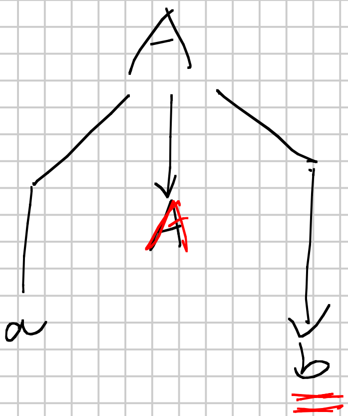
$P_D = \{S \rightarrow Da,\ D \rightarrow dD\}$

Pour calculer Suivant(X), on regarde les règle ds lesquelles apparaît en partie droite

Règle : partie gauche $\longrightarrow$ partie droite

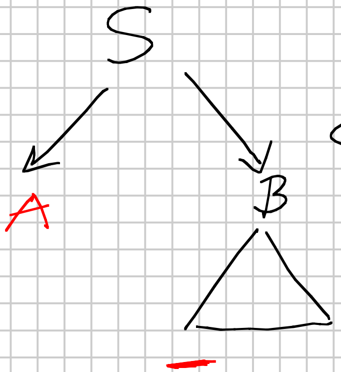
Pour suivant (A) :

$$A \longrightarrow aAb$$



Pour l'axiome, on ajoute le marqueur de fin de la phrase

$$S : \longrightarrow AB$$



$$\text{Suivant}(S) = \{\#\} \cup \bigcup_{p \in P_S} \text{Suivant}(p)$$

$S \to AB \mid Da$
$A \to aAb \mid \varepsilon$
$B \to bB \mid \varepsilon$
$D \to dD \mid \varepsilon$

Fonction Suivant : # : marqueur fin de phrase.

$\# \in$ Suivant(axiome = S)

Soit $P_X \subset P$: ensemble des règles p dans lesquelles X apparaît en partie droite.

$$\text{Suivant}(X) = \bigcup_{p \in P_X} \text{Suivant}_p(X)$$

avec $\text{Suivant}_p(X) =$

1) $p = Y \to \alpha X$: Suivant(Y)
2) $p = Y \to \alpha X \beta$ et Nullable(β) = Faux : Premier(β)
3) $p = Y \to \alpha X \beta$ et Nullable(β) = Vrai : Premier(β) $\cup$ Suivant(Y)

$$\text{Suivant}(A) = \underbrace{\text{Suivant}(A)}_{A \to aAb} \cup \underbrace{\text{Suivant}(A)}_{S \to AB}$$

$= \underbrace{\text{Premier}(b)}_{\{b\}} \cup \underbrace{\text{Premier}(B)}_{\{b\}} \cup \text{Suivant}(S) = \boxed{\{b, \#\}}$

$\boxed{\text{Suivant}(S) = \{\#\}}$

$A \cup A = A$

$$\text{Suivant}(B) = \underbrace{\text{Suivant}(B)}_{S \to AB} \cup \underbrace{\text{Suivant}(B)}_{B \to bB} =$$

$= (\text{Premier}(B) \cup \text{Suivant}(S)) \cup \text{Suivant}(B)$ $\qquad A \cup A = A$

$= \{b\} \cup \{\#\} = \{b, \#\}.$

$\text{Suivant}(D) = \underbrace{\text{Suivant}(D)}_{D \to dD} \cup \underbrace{\text{Suivant}(S)}_{S \to Da} = \text{Suivant}(D) \cup \text{Premier}(a)$

$= \{a\}$

Remplir la table d'analyse (descendante)

= matrice $M[X, a]$ avec $X \in V_n$, $a \in V_t$

	... a ... V_t
V_n X	règle de production

Remplissage de la table d'analyse :

Soit G une grammaire, ses ensembles premiers et suivants. $G = (S, V_n, V_t, P)$

Sortie : table d'analyse.

Pour tout règle/production $X \to \alpha \in P$

Pour tout $a \in Premier(\alpha)$

ajouter $X \to \alpha$ à $M[X, a]$

si $Nullable(\alpha) = vrai$

pour tout $b \in Suivant(X)$

faire $M[X, b] = X \to \alpha$

Case vide = erreur.

$S \rightarrow AB \mid Da$

$A \rightarrow aAb \mid \varepsilon$

$B \rightarrow bB \mid \varepsilon$

$D \rightarrow dD \mid \varepsilon$

Après le calcul, on obtient :

	a	b	d	e	#
S	$S \rightarrow AB$	$S \rightarrow AB$	$S \rightarrow Da$	$S \rightarrow Da$	$S \rightarrow AB$
A	$A \rightarrow aAb$	$A \rightarrow \varepsilon$	erreur	erreur	$A \rightarrow \varepsilon$
B	erreur	$B \rightarrow bB$	erreur	erreur	$B \rightarrow \varepsilon$
D	erreur	erreur	$D \rightarrow dD$	$D \rightarrow e$	erreur

Algo d'analyse syntaxique dirigé par table :

<u>Entrée</u> : . phrase à analyser (avec # comme marqueur fin de la phrase)

. pile (initialement contient # et l'axiome au sommet.

. Table d'analyse $\underline{M}$

<u>Sortie</u> : phrase acceptée / rejetée

Principe : On compare le lexème courant t avec le sommet de la pile X. On a 3 cas :

1) si $X = t = \#$, fin d'analyse et la phrase est acceptée.

2) si $X = t \neq \#$, on dépile X et on avance vers le lexème suivant.

3) si $X \in V_n$, on consulte l'entrée $M[X, t]$:

si $M[X, t] =$ "$X \rightarrow X_1 X_2 X_3 \ldots X_m$", on remplace X par $X_m \ldots X_1$ (X_1 au sommet de la pile.

si $M[X, t] =$ "erreur" : erreur syntaxique

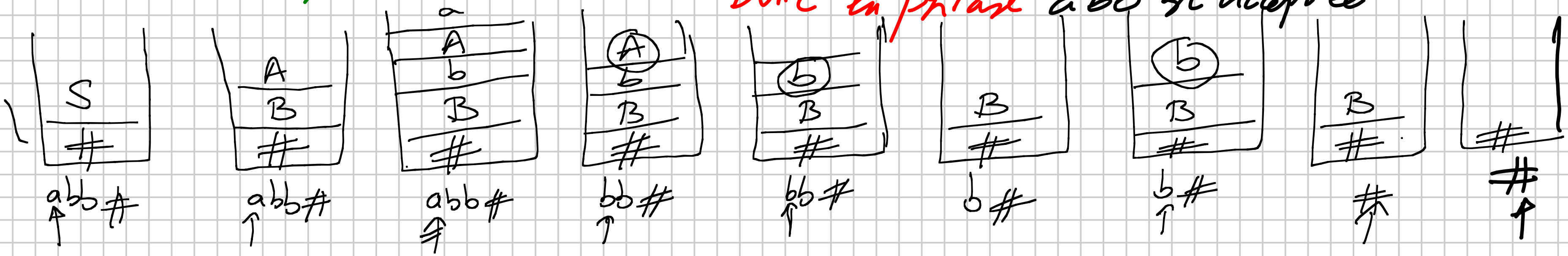
Exemple : Analyse de la phrase $w = abb\#$

	a	b	d	e	#
S	$S \to AB$	$S \to AB$	$S \to Da$	$S \to Da$	$S \to AB$
A	$A \to aAb$	$A \to \varepsilon$	erreur	erreur	$A \to \varepsilon$
B	erreur	$B \to bB$	erreur	erreur	$B \to \varepsilon$
D	erreur	erreur	$D \to dD$	$D \to e$	erreur

Pile	Phrase à analyser	règle appliquée
#S	abb#	$S \to AB$
#BA	abb#	$A \to aAb$
#BbAa	abb#	
#BbA	bb#	$A \to \varepsilon$
#Bb	bb#	
#B	b#	$B \to bB$
#Bb	b#	
#B	#	$B \to \varepsilon$
#	#	

Sommet de la pile (#) = lexème courant (#) = #

Donc la phrase abb est acceptée



Note Title 12/01/2022

Grammaire LL(1), Récursivité à gauche, factorisation à gauche

Pour effectuer l'analyse prédictive, il faut que la grammaire $\underline{G}$ vérifie certaines conditions :

1) G est non ambigüe
et 2) G est $\underline{\text{non}}$ récursive à gauche
et 3) G est factorisée à gauche.

on dit que G est une grammaire LL(1).

La nomination LL(1) signifie :

* première lettre L : on parcourt la phrase de la gauche vers la droite
* deuxième lettre L : on effectue la dérivation gauche.

* le "1" : on lit le prochain lexème (un seul lexème) pour prendre la décision (càd choisir la bonne règle !

Th1 G est LL(1) $\Longleftrightarrow$
$G = (V_t, V_n, P, S)$

1) G non récursive à gauche
et
2) $\forall\ A \rightarrow \alpha_1 \mid \alpha_2 \in P$: $Premier(\alpha_1) \cap Premier(\alpha_2) = \emptyset$
et
3) $\forall\ A \rightarrow \delta \mid \varepsilon$: $Premier(\delta) \cap Suivant(A) = \emptyset$

Th2 G est LL(1) $\Longleftrightarrow$ Table d'analyse M :
$\forall X \in V_n$
$\forall a \in V_t$
$M[X, a] = $ 1 règle / vide = erreur

Récursivité immédiate à gauche :

$G = (V_t, V_n, P, S)$ est immédiatement récursive à gauche si elle contient **au moins** une règle de la forme $X \to X\alpha$ où $\alpha \in (V_t \cup V_n)^*$

Alg

Eliminer récursivité immédiate à gauche :

Remplacer

$$\begin{cases} X \to X\alpha \\ \text{et } X \to \beta_1 \mid \beta_2 \mid \dots \mid \beta_m \end{cases}$$

par

$$X \to \beta_1 Y \mid \beta_2 Y \mid \dots \mid \beta_m Y$$

Exemple :

$X \to X\alpha$

$\neq$

$X \to X\alpha \mid \varepsilon$

$G = (V_t, V_n, P, S)$ avec

$$\begin{cases} S \to ScB \mid C \\ B \to Ba \mid \varepsilon \\ C \to Cc \mid b \mid d \end{cases}$$

$S \to ScB \mid C$

$\alpha = cB$ $\quad \beta_1$

$B \to Ba \mid \varepsilon$

$\alpha = a$ $\quad \beta'$

$C \to Cc \mid b \mid d$

$\alpha = c$ $\quad \beta_1 \quad \beta_2$

$S \to CX$

$X \to cBX \mid \varepsilon$

$B \to \varepsilon Y = Y$

$Y \to aY \mid \varepsilon$

$C \to bZ \mid dZ$

$Z \to cZ \mid \varepsilon$

$Y \to \alpha Y \mid \varepsilon$

$\Longrightarrow G'$

$S \to CX$

$X \to cBX \mid \varepsilon$

$B \to Y$

$Y \to aY \mid \varepsilon$

$C \to bZ \mid dZ$

$Z \to cZ \mid \varepsilon$

Factorisation à gauche :

Def : $G = (V_t, V_n, P, S)$ est non factorisée à gauche si elle contient au moins une alternative de la forme

$$X \rightarrow \alpha\beta \mid \alpha\delta \quad \text{où } \alpha, \beta, \delta \in (V_t \cup V_n)^*$$

Exemple : G

$S \rightarrow aEbS \mid aEbSeB \mid a$ $\quad \alpha = aEbS$

$E \rightarrow bcB \mid bca$ $\quad \alpha = bc$

$B \rightarrow ba$

n'est pas factorisée à gauche. $\Longrightarrow$ G n'est pas LL(1)

Alg de factorisation à gauche d'une grammaire

1) Pour chaque non terminal X, trouver α, le plus long préfixe commun à deux alternatives ou plus. $\alpha, \beta_i, \delta \in (V_t \cup V_n)^*$

2) si $\alpha \neq \varepsilon$ remplacer toutes les règles

$X \rightarrow \alpha\beta_1 \mid \alpha\beta_2 \mid \dots \mid \alpha\beta_n \mid \delta$: avec δ : représente toutes les alternatives qui ne commencent pas par α

par : $X \rightarrow \alpha Y \mid \delta$ et $Y \rightarrow \beta_1 \mid \beta_2 \mid \dots \mid \beta_n$.

3) Répéter l'étape 2 jusqu'à ce qu'aucune des alternatives

- d'un nm non-terminal n'ait de préfixe commun :

R1 $S \rightarrow aEbS \mid aEbSeB \mid a$

R2 $E \rightarrow bcB \mid bca$

R3 $B \rightarrow ba$

1ʳᵉ itération :

Pour R1 : $S \rightarrow aEbSX \mid a$ et $X \rightarrow \varepsilon \mid eB$

Pour R2 : $E \rightarrow bcY$ et $Y \rightarrow B \mid a$

Pour R3 : pas de changement

2ème itération :

Pour $S \rightarrow aEbSX \mid a$: $S \rightarrow aZ$

$Z \rightarrow EbSX \mid \varepsilon$

pour les autres règles pas de changement, Donc $\Rightarrow G'$

$S \rightarrow aZ$

$Z \rightarrow EbSX \mid \varepsilon$

$E \rightarrow bcY$

$Y \rightarrow B \mid a$

$B \rightarrow ba$

Compilation
1) Analyse
2) Génération

Rappel

1) Analyse lexicale → Sortie = Suite d'unités lexicales avec leurs lexèmes

2) Analyse syntaxique → sortie
- instruction correcte
- erreur syntaxique

→ Arbre syntaxique

3)

⋮

descendante

ascendante

Exemple: → Analyse prédictive basée sur table d'analyse table d'analyse = Matrice M (lignes = Non terminaux $\in V_n$; Colonnes = terminaux $\in V_t$)

$G = (V_t, V_n, P, S)$

$X \in V_n$

$a \in V_t$

$M[X, a] = \begin{cases} \text{règles} \in P \\ \text{vide} = \text{erreur} \end{cases}$

, On doit répondre à 2 questions :

1) Comment obtenir la table d'analyse :

$\Rightarrow$ Fonctions : Nullable, Premier, Suivant

2) Comment utiliser la table d'analyse pour vérifier si l'instruction est correcte ou non.

Exercice : Soit la grammaire

$T \to R \mid aTc$

$R \to bR \mid \varepsilon$

Question : Effectuer l'analyse par table pour le mot

$w = aabbbcc\#$

$Premier(aTc) = \{a\}$

$Premier(R) \overset{4}{=} Premier(bR) \cup Premier(\varepsilon)$

$\overset{3}{=} \{b\}$

$Premier(T) = Premier(R) \cup Premier(aTc)$

$= \{b\} \cup \{a\} = \{a, b\}$

Fonction Premier : $\alpha \in (V_t \cup V_n)^*$

1) $Premier(\alpha = \varepsilon) = \phi$

2) $\alpha = a, a \in V_t$, $Premier(\alpha) = \{a\}$

3) $\alpha = a\beta$, $a \in V_t$, $\beta \in (V_t \cup V_n)^*$ $Premier(\alpha) = \{a\}$

4) $\alpha = X$, $X \in V_n$ $X \to \alpha_1 \mid \alpha_2 \mid \dots \mid \alpha_n$ $Premier(\alpha) = \cup\, Premier(\alpha_i)$

5) $\alpha = X\beta$ $X \in V_n$, $\beta \in (V_t \cup V_n)^*$

5.1) $Nullable(X) = Vrai$: $Premier(\alpha) = Premier(X) \cup Premier(\beta)$

5.2) $Nullable(X) = Faux$: $Premier(\alpha) = Premier(X)$

Fonction Suivant : $\#$: marqueur fin de phrase

$\# \in Suivant(axiome = S)$

Soit $P_X \subset P$: ensemble des règles p dans lesquelles X apparaît en partie droite

$Suivant(X) = \bigcup_{p \in P_X} Suivant_p(X)$

avec $Suivant_p(X) =$

1) $p = Y \to \alpha X$: $Suivant(Y)$

2) $p = Y \to \alpha X \beta$ et $Nullable(\beta) = Faux$: $Premier(\beta)$

3) $p = Y \to \alpha X \beta$ et $Nullable(\beta) = Vrai$: $Premier(\beta) \cup Suivant(Y)$

T : a sitne $\Rightarrow Suivant(T) = \{\#\} \cup \underset{T \to aTc}{Suivant}(T)$

$\ast \underset{T \to aTc}{Suivant}(T) = Premier(c) = \{c\}$ $(Nullable(c) = Faux) \Longrightarrow Suivant(T) = \{\#, c\}$

$Suivant(R) = \underset{T \to R}{Suivant}(R) \cup \underset{R \to bR}{Suivant}(R) = Suivant(T) \cup Suivant(R)$

$\Longrightarrow Suivant(R) = \{c, \#\}$

Matrice M :

	a	b	c	#
T	$T \to aTc$	$T \to R$	$T \to R$	$T \to R$
R	erreur	$R \to bR$	$R \to \varepsilon$	$R \to \varepsilon$

Alg

Remplissage de la table d'analyse :

Entrée : une grammaire G, ses ensembles Premier et Suivant

Sortie : Table d'analyse M

Algo : Pour toute production $X \to \alpha \in P$

Pour toute $a \in Premier(\alpha)$: ajouter $X \to \alpha$ à $M[X, a]$

si $Nullable(\alpha) = Vrai$ alors

pour tout $b \in Suivant(X)$ faire :

$M[X, b] = X \to \alpha$.

fsi

fpour

Ajouter erreur dans les entrées de T restées vides.

Analyse descendante par table d'analyse du mot

Pile	phrase à analyser	règle appliquée
# T	aabbbcc#	initialement
# c T a	aabbbcc #	r1 $T \to aTc$
# c T	abbbcc #	
# cc T a	abbbcc #	r2 $T \to aTc$
→ # cc T $M[T, b]$	bbbcc #	
# cc R $M[R, b]$	bbbcc #	r3 $T \to R$
# cc R b	bbbcc #	r4 $R \to bR$
# cc R $M[R, b]$	bbcc #	
# cc R b	bcc #	r5 $R \to bR$
# cc R $M[R, b]$	bcc #	
# cc R b	bcc #	r6 $R \to bR$
# cc R $M[R, c]$	cc #	

M	a	b	c	#
T	$T \to aTc$	$T \to R$	$T \to R$	$T \to R$
R	erreur	$R \to bR$	$R \to \varepsilon$	$R \to \varepsilon$

$W = aabbbcc\#$

- pile (initialement [redacted] contient # et l'axiome au sommet).
- Table d'analyse M

3 cas :

1) si $X = t = \#$, fin d'analyse et la phrase est acceptée.

2) si $X = t \neq \#$, on dépile X et on avance vers le lexème suivant.

3) si $X \in V_N$, on consulte l'entrée $M[X,t]$:

si $M[X,t] = "X \to X_1X_2X_3 \ldots X_n"$, on remplace X par $X_n \ldots X_1$ (X_1 au sommet de la pile.

si $M[X,t]$ = "erreur" : erreur syntaxique

cε (c.c.ε ≡ cc) εc # r7 $\boxed{R \to \varepsilon}$

c c

Acceptée

le mot w = aabbbbcc est acceptée

(r1 . . . r7) $\Rightarrow$ arbre syntaxique.