

Vocabulaire:

Année universitaire 2010-2011
Techniques de compilation

Thème : 2.5.0

a) Énumération de la récurseur à gauche

$$S \rightarrow S, S \cup w \text{ et } S \text{ id } E$$

$$\Rightarrow \begin{cases} S \rightarrow w C D S' \mid id = E \\ S' \rightarrow ; S S' \mid e \end{cases}$$

Soit $\{V, R, S\}$ avec $V = \{A, B, C, D, E\}$ et $R = \{S, E, id\}$.
 S est l'axiome et E est donné par les règles de production suivantes:

$$\begin{cases} S \rightarrow w C D S' \mid id = E \\ C \rightarrow id \mid OP id \end{cases}$$

$$\begin{cases} E \rightarrow E E' \mid id E' \\ E' \rightarrow id + id E' \mid e \end{cases}$$

2) Ensemble des règles suivants

$$\begin{cases} PR(c) = \{w, id\} \\ PR(S') = \{\cdot, \cdot\} \end{cases}$$

$$PR(c) = \{id\}$$

$$PR(OP) = \{id\}$$

$$PR(E) = \{\cdot, \cdot, \cdot\}$$

$$PR(E') = \{\cdot, \cdot, \cdot\}$$

Construction de la table d'analyse.

$$R R(\bar{c}) = \{id, -\}$$

$$R R(\bar{c}') = \{+, \cdot\}$$

Construction de la table d'analyse.

$$R R(\bar{S}) = \{+, -, \cdot\}$$

Construction de la table d'analyse.

$$R R(\bar{E}) = \{+, -, \cdot\}$$

Construction de la table d'analyse.

$$R R(\bar{E'}) = \{+, -, \cdot\}$$

Construction de la table d'analyse.

$$R R(\bar{id}) = \{+, -, \cdot\}$$

Construction de la table d'analyse.

$$R R(\bar{OP}) = \{+, -, \cdot\}$$

S'

S''

S'''

S''''

S'''''

S'''''

S''''''

S'''''''

S'''''''

S''''''''

S''''''''

S'''''''

institut technique industriel
équivalents à droite et à gauche
dans le tableau

tableau : 2.5.0

$$S \frac{2}{2}^+ w C d S | S \bar{S} i = E$$

Entrée

$$\begin{aligned} & \# wcdid = \\ & \# wcdidid = \\ & \# wcdididid = \\ & \# wcdidididid = \end{aligned}$$

Entrée

$$\begin{aligned} & \# wcdid = \\ & \# wcdidid = \\ & \# wcdididid = \\ & \# wcdidididid = \end{aligned}$$

Action

La chanson 'windy wind' est enregistrée par G.

```
7 Public static void main( ) { int y = 3 ; } }
```

Public

Halic
loid
main
() \rightarrow $\frac{1}{m} \rightarrow n = m + \infty$

二

Public static void main() {
 System.out.println("Hello World");
}

Public
1

have old
inconcern
paraphrase
paraphrase
accuse
survive
TCL
copular
verb
accuse

14

الله رب العالمين

卷之二

10. 11. 12. 13. 14. 15. 16. 17. 18. 19. 20.

Letters in blue
represent
the first
letter
of each
word.

1

letter is app
for 25

Correction TD n°1 : Analyse lexicale

2020-2021

Exercice 1.

| Lexème | UL |
|----------|-----------|
| fonction | function |
| max | ID |
| (| Para |
| : | ID |
| , | Separator |
| j | ID |
| : | Separator |
| integer | integer |
|) | Para |
| begin | begin |
| if | if |
| : | ID |
| > | OP |
| j | ID |
| then | then |
| max | ID |
| ::= | Aff |
| j | ID |
| end | end |
| ; | Separator |

Exercice 2:

$\langle ID, P_1 \rangle$ Pointeur vers la table de symbole associé à Montant

$\langle OP, Aff \rangle$

$\langle ID, P_2 \rangle$ Pointeur vers la table de symbole associé à Prix

$\langle Para, Pm \rangle$

$\langle ID, P_3 \rangle$ Pointeur vers la table de symbole associé à Quantité

$\langle Para, Fn \rangle$

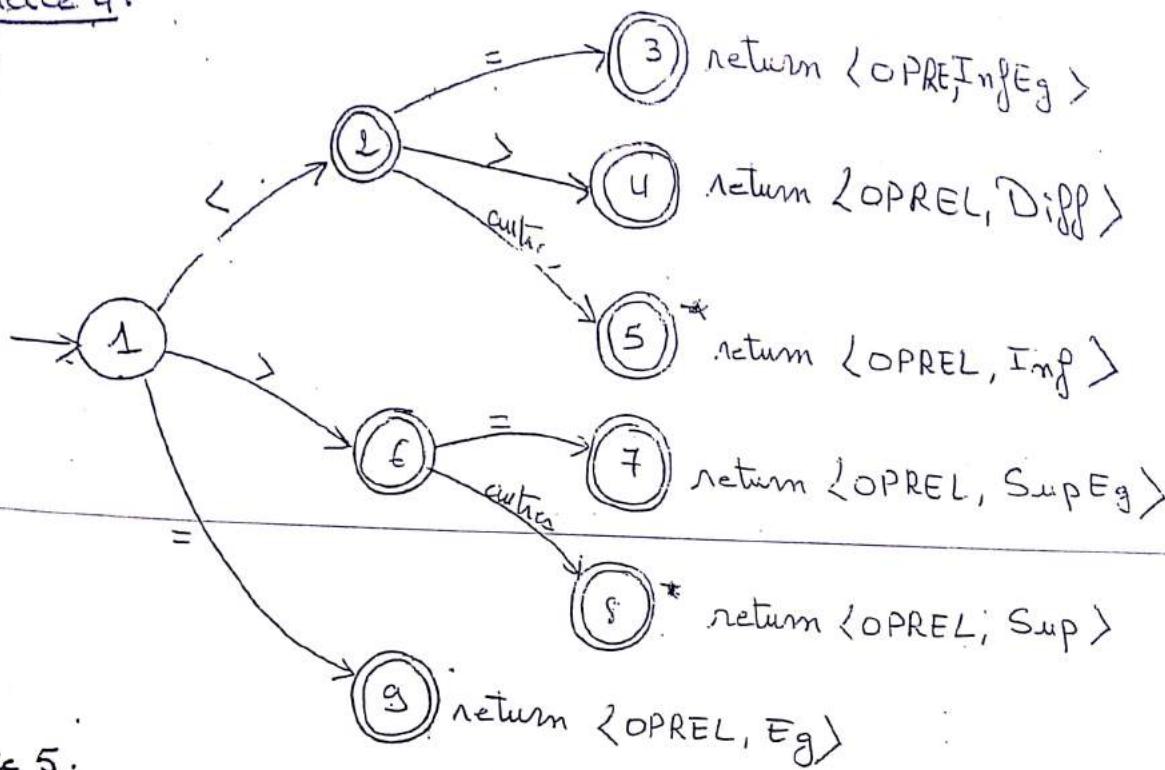
$\langle OP, ADD \rangle$

$\langle ALB, 12, F \rangle$

exercice 3:

| UL | Lexème | ER |
|---------|----------|-----------------------------|
| Class | Class | Class |
| ID | Cette | tette. (lettre chiffre)* |
| Acc | { | { } |
| Private | Private | Private |
| double | double | double |
| ID | rayon | lettre. (lettre chiffre)* |
| PV | ; | ; |
| Public | Public | Public |
| void | void | void |
| ID | setRayon | lettre. (lettre chiffre)* |
| Para | (| () |
| double | double | double |
| ID |) |) |
| Para | { | { } |
| Acc | rayon | lettre. (lettre chiffre)* |
| ID | = | = * |
| OP | r | lettre. (lettre chiffre)* |
| ID | ; | ; |
| PV | } | { } |
| Acc | Public | Public |
| Public | double | double |
| double | aire | lettre. (lettre chiffre)* |
| ID | (| () |
| Para |) | () |
| Para | { | { } |
| Acc | return | return |
| Return | rayon | lettre. (lettre chiffre)* |
| ID | * | * = |
| OP | rayon | lettre. (lettre chiffre)* |
| ID | * | * = |
| OP | PI | NB. NB+ NB |
| NB | ; | ; |
| PV | } | { } |
| Acc | } | { } |
| Acc | } | { } |

Exercice 4:



Exercice 5:

- 1) a) $\forall X 1 \in 3$
- b) $\forall 1 2 5 4 A B C D X Z 1 3 5 4$
- c) $\forall C D C D Z \in 1 3 2$

- 2) $Cste \rightarrow \forall$
 $letter \rightarrow A 1 \dots 1 Z$
 $chiffre \rightarrow 1 1 2 1 3 1 4 1 5$
 $Variable \rightarrow cste \ (letter \mid chiffre)^*$
- 3)


```

graph LR
    S(( )) --> 1((1))
    1 -- "forall" --> 2((2))
    2 -- "chiffre" --> " "
    2 -- "letter" --> " "
  
```

Exercice 6:

- 1) $\frac{Com}{N} \frac{C1}{V} \frac{\{x, y\}}{P} \frac{[t; f]}{O}$
 $Com \ C2 \ [x; y]$

- 2) $c \rightarrow [a-z \ A-Z]$

$ID \rightarrow (c)^+$

$N \rightarrow ID$

$V \rightarrow (ID)?$

$P \rightarrow (\{C \mid C\}^*)?$

$O \rightarrow ([C \mid C]^*)?$

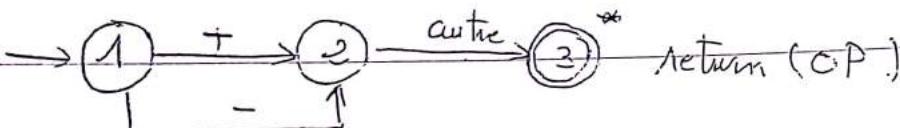
Exercice 7:

$$nb \rightarrow [0-9]^+$$

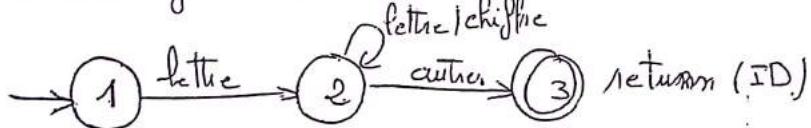
$$V \rightarrow [nb\ , nb]^* \mid V^+ \mid V^- \mid nb^* \mid V^* \mid nb$$

Exercice 8:

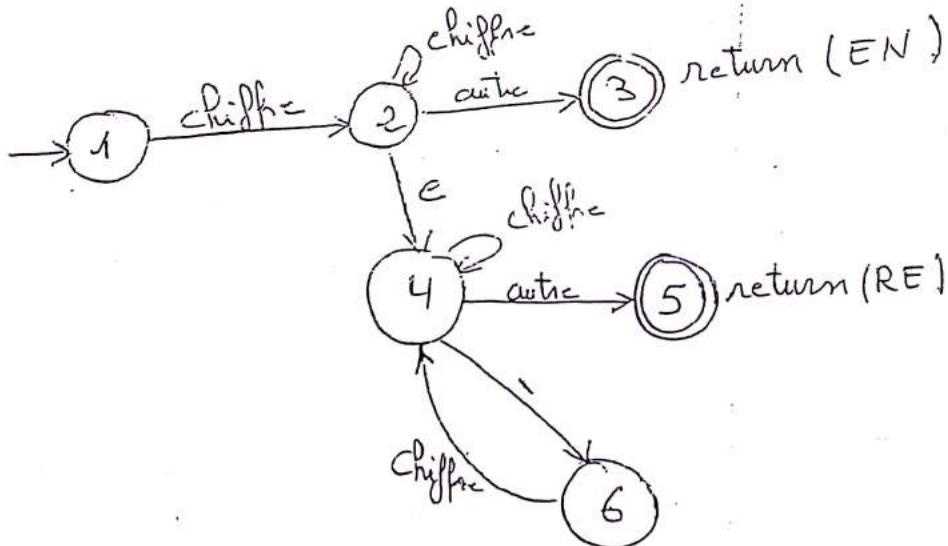
1) • Les opérateurs: OP



• Les identificateurs: ID
lettre | chiffre



• Les entiers et les réels: EN, RE



2) $OP \rightarrow [+ \mid -]$

ID \rightarrow lettre. (lettre | chiffre)*

EN \rightarrow chiffre (chiffre)*

RE \rightarrow (chiffre)* e (-1)? (chiffre)*

3) $+ \mid - ; \quad OP \cdot OP$

Aa + g; ID OP EN

Aeg + g; ID OP EN

ABc12; RE

6 b + a; EN ID OP ID

Exercice 10:

O₁- Nom commande : NC

Liste d'arguments: LA

Liste d'options: LO

Q2- ID → Lettre.(Lettre | chiffre)*

NC → ID

LA → (ID (ID)*)?

LO → ([OP (,OP)*])?

OP → - Lettre

TD 2 : Analyse Syntaxique

Exercice 1 :

1. Proposer une grammaire générant le langage de chaînes $\{(,)\}^*$ qui sont bien parenthésées.
2. Construire l'arbre syntaxique de la chaîne $(()) () ()$.

Exercice 2 :

Soit la grammaire non contextuelle :

$$S \rightarrow SS+ | SS^* | a$$

1. Montrer comment la chaîne $a+a+a^*$ est engendrée par la grammaire.
2. Construire un arbre syntaxique correspondant à cette chaîne.
3. Quel langage est engendré par cette grammaire ?

Exercice 3 :

Soit la grammaire suivante :

$$E \rightarrow T | E+T$$

$$T \rightarrow id | (E) | nb$$

1. Quels sont les unités lexicales étudiées par cette grammaire ?
2. Proposer une instruction syntaxiquement correcte (selon la grammaire).
3. Donner les dérivations gauche et droite de l'instruction proposée.

Exercice 4 :

Soit la grammaire suivante :

$$\begin{cases} E \rightarrow E+T \mid T \\ T \rightarrow T^*F \mid F \\ F \rightarrow (E) \mid id \end{cases}$$

1. Éliminer la récursivité à gauche et factoriser si nécessaire.
2. Donner la table d'analyse de la nouvelle grammaire. Est-elle LL(1) ?

Exercice 5 :

Soit la grammaire :

$$\begin{cases} S \rightarrow Ab|a|AA \\ A \rightarrow Sa|Ac|B \\ B \rightarrow Sd \end{cases}$$

1. Éliminer la récursivité à gauche.
2. Factoriser si nécessaire.
3. La grammaire obtenue est-elle LL(1) ?

Exercice 6 :

Soit la grammaire

$$\begin{cases} S \rightarrow a \mid b \mid T \\ T \rightarrow T, S \mid S \end{cases}$$

1. La grammaire est-elle LL(1) ?
 2. Eliminer la récursivité à gauche et factoriser si nécessaire.
 3. Montrer que la nouvelle grammaire est LL(1). Donner la table d'analyse.
 4. Expliciter le comportement de l'analyseur sur les mots :
- $m_1 = (a, a); m_2 = (a, (b, a), a)$

Exercice 7 :

Soit la grammaire :

$$\begin{aligned} S &\rightarrow a \mid A \\ A &\rightarrow Sa \mid Ac \mid b \end{aligned}$$

1. Eliminer la récursivité à gauche et factoriser si nécessaire.
2. Donner une dérivation gauche du mot « aabc » dans la nouvelle grammaire obtenue.
3. Calculer l'ensemble premier et l'ensemble suivant de chaque non terminal.
4. Donner la table d'analyse de l'analyseur descendant. La grammaire obtenue est-elle LL(1) ?
5. Retrouver la dérivation du mot « aabc » de la question 2 en effectuant une analyse descendante.

Exercice 8 :

Soit la grammaire des expressions post-séparées, G1 :

$$\begin{aligned} E &\rightarrow E E \mid n \\ O &\rightarrow + \mid \cdot \mid ^* \mid / \\ n &\rightarrow [0-9] \end{aligned}$$

Soit la grammaire G2 :

$$\begin{aligned} E &\rightarrow n F \\ F &\rightarrow E O F \mid \epsilon \\ O &\rightarrow + \mid \cdot \mid ^* \mid / \\ n &\rightarrow [0-9] \end{aligned}$$

1. Montrez que G2 est équivalente à G1.
2. À l'aide de l'algorithme de l'analyse syntaxique ascendante, vérifier si le mot « 54+2*37^+ » est engendré par G1.

Exercice 9 :

Soit la grammaire :

$$\begin{aligned} S &\rightarrow BB \\ B &\rightarrow aB \mid b \end{aligned}$$

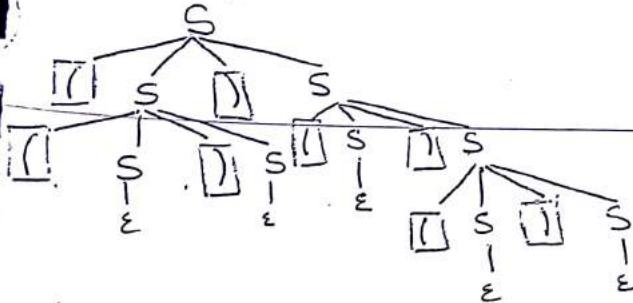
Expliciter le comportement de l'analyseur syntaxique ascendant sur la chaîne aabaab.

Compilateur
Correction du TD N°2: Analyse Syntaxique

25/12 - 2023

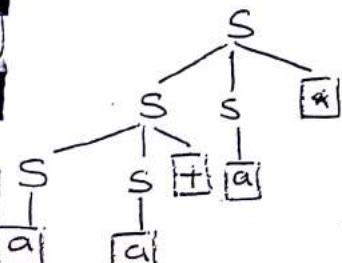
Exercice 2:

$$S \rightarrow (S)S \mid \epsilon$$



Exercice 3:

$$S \rightarrow SS^* \rightarrow SS + S^* \rightarrow aS + S^* \rightarrow aa + S^* \rightarrow aa + aa^*$$



Le langage des expressions arithmétiques postfixées (où un opérateur apparaît après ses opérandes)

Exercice 3:

i) Les unités lexicales : { "id", "(", ")" , "nb" , "+" }

ii) $(id + nb) + id$.

iii) D à G :

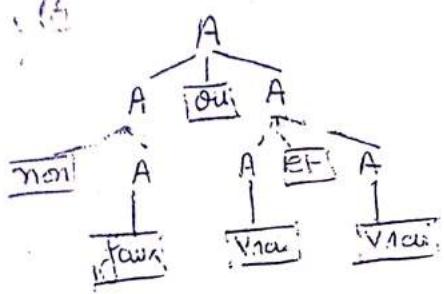
$$\rightarrow E + T \rightarrow T + T \rightarrow (E) + T \rightarrow (E + T) + T \rightarrow (T + T) + T \rightarrow (id + T) + T \rightarrow$$

iv) $id + nb + T \rightarrow (id + nb) + id$

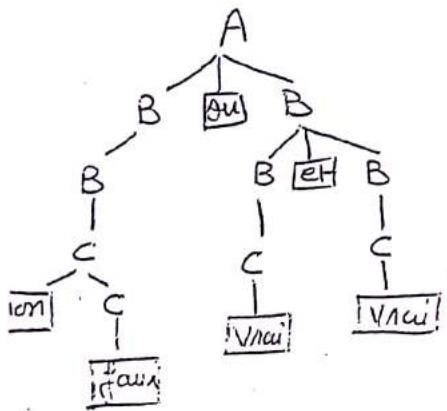
v) D :

$$\rightarrow E + T \rightarrow E + id \rightarrow T + id \rightarrow (E) + id \rightarrow (E + T) + id \rightarrow (E + nb) + id \rightarrow$$

vi) $T + nb + id \rightarrow (id + nb) + id$



On peut trouver d'autres arbres syntaxiques pour la même chaîne
 ⇒ La grammaire est ambiguë



Grammaire non ambiguë

Soit la chaîne : id * id + id

⇒ G₁
 $E \rightarrow E + E \rightarrow E * E + E \rightarrow id * E + E \rightarrow id * id + E \rightarrow id * id + id$

⇒ G₂
 $E \rightarrow E * E \rightarrow id * E \rightarrow id * E + E \rightarrow id * E + id \rightarrow id * id + id$

⇒ Pour la même chaîne id * id + id on a deux dérivations le plus à gauche

→ Grammaire ambiguë

En se basant sur l'associativité et la priorité des opérateurs :

$$\begin{aligned} E &\rightarrow E + T \mid T \\ T &\rightarrow T * F \mid F \\ F &\rightarrow (E) \mid id \end{aligned}$$

3) Elimination de la récursivité

$$\left\{ \begin{array}{l} E \rightarrow TE' \\ E' \rightarrow +TE' \mid \epsilon \\ T \rightarrow FT' \\ T' \rightarrow *FT' \mid \epsilon \\ F \rightarrow (E) \mid \text{id} \end{array} \right.$$

4) Calcul des ensembles premiers

$$PR(E) = \{(, \text{id}\}$$

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

$$PR(T) = \{(, \text{id}\}$$

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

$$PR(F) = \{(, \text{id}\}$$

Calcul des ensembles suivants

$$SV(E) = \{\$,)\}$$

$$SV(E') = \{+,), \$\}$$

$$SV(T) = \{\$\, ,)\}$$

$$SV(T') = \{+,), \$\}$$

$$SV(F) = \{*, +,), \$\}$$

Construction de la table d'analyse

| | | | | | | | |
|--------------|---------------------|---------------------|---------------------------|---------------------------|----------------------|---------------------------|----|
| V | T | id | (|) | + | * | \$ |
| E | $E \rightarrow TE'$ | $E \rightarrow TE'$ | | | | | |
| E' | | | $E' \rightarrow \epsilon$ | $E' \rightarrow +TE'$ | | $E' \rightarrow \epsilon$ | |
| T | $T \rightarrow FT'$ | $T \rightarrow FT'$ | | | | | |
| T' | | | $T' \rightarrow \epsilon$ | $T' \rightarrow \epsilon$ | $T' \rightarrow FT'$ | $T' \rightarrow \epsilon$ | |
| F | $F \rightarrow id$ | $F \rightarrow (E)$ | | | | | |

→ La table d'analyse ne contient aucune case définie de façon multiple
 ↳ grammaire LL(1)

Exercice 2

$B \rightarrow B \vee B \mid B \wedge B \mid B \Rightarrow B \mid \neg B \mid (B) \mid \text{id} \mid \text{vrai} \mid \text{faux}$

1) Soit la chaîne $\text{faux} \vee \text{faux} \Rightarrow \text{vrai}$

Dès lors :

$B \rightarrow B \Rightarrow B \rightarrow B \vee B \Rightarrow B \rightarrow \text{faux} \vee B \Rightarrow B \rightarrow \text{faux} \vee \text{faux} \Rightarrow B \rightarrow \text{faux} \vee \text{faux} \Rightarrow \text{vrai}$

$B \rightarrow B \vee B \rightarrow \text{faux} \vee B \rightarrow \text{faux} \vee B \Rightarrow B \Rightarrow \text{faux} \vee \text{faux} \Rightarrow B \Rightarrow \text{faux} \vee \text{faux} \Rightarrow \text{vrai}$
Deux dérivation le plus à gauche pour la même chaîne \Rightarrow la grammaire est ambiguë.

2) Priorité des opérateurs : ' \neg ' > ' \wedge ' > ' \vee ' \Rightarrow

Commencant par le plus faible à partir de l'axiome.

$B \rightarrow \neg T \Rightarrow B \mid T$ (\Rightarrow associatif à gauche ' \neg ' est ajouté à droite)

$T \rightarrow T \vee F \mid F$ (\vee associatif à droite ' F ' est ajouté à gauche)

$F \rightarrow F \wedge R \mid R$ (\wedge " " "

$R \rightarrow \neg R \mid (B) \mid \text{id} \mid \text{faux} \mid \text{vrai}$

3) * Elimination de la récursivité à gauche :

- Il n'y a pas de récursivité cachée.

$B \rightarrow T \Rightarrow B \mid T$

$T \rightarrow FT'$

$T' \rightarrow \vee FT' \mid \epsilon$

$F \rightarrow RF'$

$F' \rightarrow \wedge RF' \mid \epsilon$

$R \rightarrow \neg R \mid (B) \mid \text{id} \mid \text{faux} \mid \text{vrai}$

* factorisation :

$B \rightarrow TB'$

$B' \rightarrow \Rightarrow B \mid \epsilon$

$T \rightarrow FT'$

$T \rightarrow \neg FT' \mid \epsilon$

$F \rightarrow RF'$

$F' \rightarrow \wedge RF' \mid \epsilon$

$R \rightarrow TR(B) \mid id \mid \text{faux} \mid \text{vrai}$

Calculation des ensembles premiers :

$$PR(B) = PR(T) \setminus \{\epsilon\} = \{\top, \perp, id, \text{faux}, \text{vrai}\}$$

$$PR(B') = \{\Rightarrow, \epsilon\}$$

$$PR(T) = PR(F) \setminus \{\epsilon\}$$

$$PR(T') = \{\vee, \epsilon\}$$

$$PR(F) = PR(R) \setminus \{\epsilon\} = \{\top, \perp, id, \text{faux}, \text{vrai}\}$$

$$PR(F') = \{\wedge, \epsilon\}$$

$$PR(R) = \{\top, \perp, id, \text{faux}, \text{vrai}\}$$

Calculation des ensembles suivants :

$$SV(B) = \{\$, ,\}$$

$$SV(B') = SV(B) = \{\$\}$$

$$SV(T) = PR(B') \setminus \{\epsilon\} \cup SV(B) = \{\Rightarrow, \$, ,\}$$

$$SV(T') = SV(T) = \{\Rightarrow, \$, ,\}$$

$$SV(F) = PR(T') \setminus \{\epsilon\} \cup SV(T) \cup SV(T') = \{\vee, \Rightarrow, \$, ,\}$$

$$SV(F') = SV(F) = \{\vee, \Rightarrow, \$, ,\}$$

$$SV(R) = PR(F') \setminus \{\epsilon\} \cup SV(F) \cup SV(F') = \{\wedge, \vee, \Rightarrow, \$, ,\}$$

Construction de la table d'analyse:

| T | id | $($ | $)$ | $vrai$ | $faux$ | \Rightarrow | T | \wedge | $v.$ | $\$$ |
|------|---------------------|---------------------|---------------------------|----------------------|----------------------|---------------------------|---------------------|-----------------------------|---------------------------|---------------------------|
| B | $B \rightarrow TB'$ | $B \rightarrow TB'$ | - | $B \rightarrow TB'$ | $B \rightarrow TB'$ | - | $B \rightarrow TB'$ | - | - | - |
| B' | - | - | $B' \rightarrow \epsilon$ | - | - | $B' \Rightarrow B$ | - | - | - | $B' \Rightarrow \epsilon$ |
| T | $T \rightarrow FT'$ | $T \rightarrow FT'$ | - | $T \rightarrow FT'$ | $T \rightarrow FT'$ | - | $T \rightarrow FT'$ | - | - | - |
| T' | - | - | $T' \rightarrow \epsilon$ | - | - | $T' \rightarrow \epsilon$ | - | - | $T' \rightarrow vFT'$ | $T' \rightarrow \epsilon$ |
| F | $F \rightarrow RF'$ | $F \rightarrow RF'$ | - | $F \rightarrow RF'$ | $F \rightarrow RF'$ | - | $F \rightarrow RF'$ | - | - | - |
| F' | - | - | $F' \rightarrow \epsilon$ | - | - | $F' \rightarrow \epsilon$ | - | $F' \rightarrow \wedge RF'$ | $F' \rightarrow \epsilon$ | $F' \rightarrow \epsilon$ |
| R | $R \rightarrow id$ | $R \rightarrow (B)$ | - | $R \rightarrow vrai$ | $R \rightarrow faux$ | - | $R \rightarrow TB$ | - | - | - |

de cas de définition de façon multiple (contient plus qu'une règle de production) \Rightarrow La grammaire est LL(1).

$(id \wedge id) \Rightarrow \neg id$

| Pile | Entrée | Action |
|---|--|--|
| \$ | $(id \wedge id) \Rightarrow \neg id \ $$ | (enlever B, mettre TB') |
| \$ \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$$ | (enlever T, mettre $T'F$) |
| \$ \$ \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$ \$$ | (enlever F, mettre RF') |
| \$ \$ \$ \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$ \$ \$$ | (enlever R, mettre (B)) |
| \$ \$ \$ \$ \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$ \$ \$ \$$ | (enlever ' ', avancer) |
| \$ \$ \$ \$ \$ \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$ \$ \$ \$ \$$ | (enlever B, mettre TB') |
| \$ \$ \$ \$ \$ \$ \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$ \$ \$ \$ \$ \$$ | (enlever T, mettre FT') |
| \$ \$ \$ \$ \$ \$ \$ \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$ \$ \$ \$ \$ \$ \$$ | (enlever F, mettre RF') |
| \$ \$ \$ \$ \$ \$ \$ \$ \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$ \$ \$ \$ \$ \$ \$ \$$ | (enlever R, mettre id) |
| \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$ \$ \$ \$ \$ \$ \$ \$ \$$ | (enlever id, avancer) |
| \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$$ | (enlever F', mettre $\wedge RF'$) |
| \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$$ | (enlever \wedge , avancer) |
| \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$$ | (enlever R, mettre id) |
| \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$$ | (enlever id, avancer) |
| \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$$ | (enlever F', mettre ϵ) |
| \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$$ | (enlever T', mettre ϵ) |
| \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$$ | (enlever B' , mettre ϵ) |
| \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$$ | (enlever ' ', avancer) |
| \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$$ | (enlever F', mettre ϵ) |
| \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$$ | (enlever T', mettre ϵ) |
| \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$$ | (enlever B' , mettre $\Rightarrow B$) |
| \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$$ | (enlever \Rightarrow , avancer) |
| \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$$ | (enlever B, mettre TB') |
| \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$$ | (enlever T, mettre FT') |
| \$ | $(id \wedge id) \Rightarrow \neg id \ $ \$$ | (enlever F, mettre RF') |

| | | |
|------------|--------|--------------------------|
| \$ | \$ | (enlever \$, mettre \$) |
| B'T'F'R | T,d \$ | (enlever T, ajouter \$) |
| B'B'T'F'R | id \$ | (enlever R, mettre id) |
| BB'T'F' id | id \$ | (enlever id, ajouter \$) |
| BB'T'F' | \$ | (enlever F, mettre \$) |
| BB'T' | \$ | (enlever T, mettre \$) |
| B'B | \$ | (enlever B, mettre \$) |
| B | \$ | Accepter |

Exercice n° 7.

$$\left\{ \begin{array}{l} S \rightarrow Ab | a | AA \\ A \rightarrow Sa | Ac | B \\ B \rightarrow Sd \end{array} \right.$$

i) * Elimination de la récursivité à gauche cachée :

Pour $i=1$

Pos de récursivité à gauche

Pour $i=2$

$j=1$

Remplacer $A \rightarrow Sa | Ac | B$ par
 $A \rightarrow Ab | a a | AA a | AC | B$

Elimer la récursivité immédiate

$$\begin{aligned} A &\rightarrow aaA' | BA' \\ A' &\rightarrow baA' | AaA' | CA' | \epsilon \end{aligned}$$

Pour $i=3$

$j=1$

Remplacer $B \rightarrow Sd$ par

$$B \rightarrow A bd | a d | AA d$$

$j=2$

Remplacer $B \rightarrow Abd | ab | AA d$ par

$$B \rightarrow aaA'bd | BA'b d | ab | aaA'a a A'd | \\ aaA'BA'd | BA'B'A'd | BA'aaA'd$$

Elimer la récursivité immédiate

$$\begin{aligned} B &\rightarrow aaA'bdB' | abB' | aaA'a a A'dB' | aaA'BA'dB' \\ B' &\rightarrow A'bdB' | A'BA'dB' | A'aaA'dB' | \epsilon \end{aligned}$$

$$V(S') = \{a, b, c\}$$

$$V(D) = \{a, b\}$$

$$V(S') = \emptyset$$

$$V(S'') = \emptyset$$

$$(A) = \{a\}$$

$$V(A') = \{a, b, d\}$$

$$(B) = \{a, b, c\}$$

$$V(C) = \{a, b, c\}$$

$$(C') = \{a, b, c\}$$

$$V(B') = \{a, b, c\}$$

$$(D) = \{a, b, c\}$$

Construction de la table d'analyse:

| V_N | V_T | a | b | c | d | \$ |
|-------|---|--|----------------------|--|---|----|
| S' | $S' \rightarrow a$ | | - | - | - | - |
| S'' | $S'' \rightarrow A$ | | $S'' \rightarrow b$ | - | - | - |
| A | $A \rightarrow aaA'$
$A \rightarrow BA'$ | | - | - | - | - |
| A' | $A' \rightarrow AaA'$
$A' \rightarrow \epsilon$ | $A' \rightarrow baA'$
$A' \rightarrow \epsilon$ | | $A' \rightarrow caA'$
$A' \rightarrow \epsilon$ | - | - |
| B | $B \rightarrow aC$ | | - | - | - | - |
| C | $C \rightarrow aA'C'$
$C \rightarrow bB'$ | | | - | - | - |
| C' | $C' \rightarrow aaA'dB'$
$C' \rightarrow BA'dB'$ | $C' \rightarrow bdB'$ | | - | - | - |
| B' | $B' \rightarrow A'D$ | $B' \rightarrow A'D$ | $B' \rightarrow A'D$ | | - | - |
| D | $D \rightarrow aaA'dB'$
$D \rightarrow BA'dB'$ | $D \rightarrow bdB'$ | | - | - | - |

Il y a des cases dans la table d'analyse contenant plus qu'un règle de production \rightarrow La grammaire n'est pas LL(1)

Grammaire résultat :

$S' \rightarrow A b | a | AA$

$A \rightarrow aaA' | BA'$

$A' \rightarrow baA' | AaA' | cA' | \epsilon$

$B \rightarrow aaA' bdB' | abB' | aaA' ddB' | abB' | aaA' ddB' | \epsilon$

$B' \rightarrow A'b d B' | A'BA'd B' | A'aaA'd B' | \epsilon$

) 1^{re} itération :

$\rightarrow AS'' | a$

" $\rightarrow b | A$

$\rightarrow aaA' | BA'$

$\rightarrow baA' | AaA' | cA' | \epsilon$

$\rightarrow aC$

$\rightarrow aA' bdB' | bB' | aA'aaA'd B' | aA'BA'd B'$

$\rightarrow A'D | \epsilon$

$\rightarrow bdB' | BA'd B' | aaA'd B'$

2^{me} itération :

$S' \rightarrow AS'' | a$

$S'' \rightarrow b | A$

$A \rightarrow aaA' | BA'$

$A' \rightarrow baA' | AaA' | cA' | \epsilon$

$B \rightarrow aC$

$C \rightarrow aA'C' | bB'$

$C' \rightarrow bdB' | aaA'd B' | BA'd B'$

$B' \rightarrow A'D | \epsilon$

$D \rightarrow bdB' | BA'd B' | aaA'd B'$

\rightarrow Grammaire factorisée.

) Il faut construire la table d'analyse.

$$R(S') = PR(A) \setminus \{\epsilon\} = \{a\}$$

$$R(S'') = \{b, a\}$$

$$R(A) = \{a\}$$

$$R(A') = \{a, b, c, \epsilon\}$$

$$R(B) = \{a\}$$

$$R(C) = \{a, b\}$$

$$R(C') = \{a, b\}$$

$$m_1 = (a, a)$$

| Pile | Entrée | Action |
|------------|-----------|--------------------------|
| \$ S | (a, a) \$ | enlever S, mettre (T) |
|) T (| (a, a) \$ | enlever (, avancer |
| \$) T | a, a) \$ | enlever T, mettre ST' |
|) T' S | a, a) \$ | enlever S, mettre a |
| \$) T' a | a, a) \$ | enlever a, avancer |
|) T | , a) \$ | enlever T', mettre , ST' |
| \$) T' S, | , a) \$ | enlever), avancer |
|) T' S | a) \$ | enlever S, mettre a |
| \$) T' a | a) \$ | enlever a, avancer |
|) T' |) \$ | enlever T', mettre E |
| \$) |) \$ | enlever), avancer |
| | \$ | Accepter |

$$m_2 = (a, (b, a), a)$$

| Pile | Entrée | Action |
|----------------|-------------------|--------------------------|
| \$ S | (a, (b, a) a) \$ | enlever S, mettre (T) |
| \$) T (| (a, (b, a) a) \$ | enlever (, avancer |
|) T | a, (b, a) , a) \$ | enlever T, mettre ST' |
| \$) T' S | a, (b, a) , a) \$ | enlever S, mettre a |
|) T' a | a, (b, a) , a) \$ | enlever a, avancer |
| \$) T' | , (b, a) , a) \$ | enlever T', mettre , ST' |
|) T' S, | , (b, a) , a) \$ | enlever), avancer |
|) T' S | (b, a) , a) \$ | enlever S, mettre (T) |
| \$) T') T (| (b, a) , a) \$ | enlever (, avancer |
|) T') T | b, a) , a) \$ | enlever T, mettre ST' |
| \$) T') T' S | b, a) , a) \$ | enlever S, mettre b |
|) T') T' b | b, a) , a) \$ | enlever B, avancer |
| \$) T') T' | , a) , a) \$ | enlever T', mettre , ST' |
|) T') T' S, | , a) , a) \$ | enlever), avancer |
| \$) T') T' S | a) , a) \$ | enlever S, mettre a |
|) T') T' a | a) , a) \$ | enlever a, avancer |
|) T') T' | , a) \$ | enlever T', mettre E |
| \$) T' | , a) \$ | enlever), avancer |
|) T' | , a) \$ | enlever T', mettre , ST' |
| \$) T' S, | , a) \$ | enlever), avancer |
|) T' S | a) \$ | enlever S, mettre a |
|) T' a | a) \$ | enlever a, avancer |
| \$) T' |) \$ | enlever T', mettre E |
|) T' |) \$ | enlever), avancer |
| | \$ | Accepter |

Exercice N°3 :

- 1) La grammaire n'est pas LL(1) car elle est récursive à gauche
 2) Élimination de la récursivité à gauche:

$$\left\{ \begin{array}{l} S \rightarrow a \mid b \mid T \\ T \rightarrow ST' \\ T' \rightarrow , ST' \mid \epsilon \end{array} \right.$$

→ Grammaire factorisée

$$PR(S) = \{ a, b, "(" \}$$

$$PR(T) = \{ a, b, "(" \}$$

$$PR(T') = \{ ;, \epsilon \}$$

$$SV(S) = \{ $, :, ;, ")" \}$$

$$V(T) = \{ ")" \}$$

$$, T' = \{ ")" \}$$

Table d'analyse

| $V_N \setminus V_T$ | a | b | (|) | , | \$ |
|---------------------|---------------------|---------------------|---------------------|---------------------------|------------------------|----|
| S | $S \rightarrow a$ | $S \rightarrow b$ | $S \rightarrow (T)$ | - | - | - |
| T | $T \rightarrow ST'$ | $T \rightarrow ST'$ | $T \rightarrow ST'$ | - | - | - |
| T' | - | - | - | $T' \rightarrow \epsilon$ | $T' \rightarrow , ST'$ | - |

→ Pas de cas de multiples → grammaire LL(1)

TB 3 : L'technique de compilation Analyse syntaxique descendante

Exercice 1 :

- Proposer une grammaire pourtant le langage de chaînes $\{a\}^*$ qui sont tous parenthétiques
- constitue l'addition syntaxique de la chaîne $(())^*$

Soit la grammaire non contextuelle :

$$S \rightarrow SS \mid S^* \mid a$$

- Montrer comment la chaîne a^m est engendrée par la grammaire.
- Quel langage est engendré par cette grammaire ?

Exercice 2 :

Soit la grammaire non contextuelle :

$$E \rightarrow T \mid E + T$$

- Quel sont les unités lexicales générées par cette grammaire ?
- Proposer une instruction syntaxique et corrective (selon la grammaire).
- Donner les dérivations gauche et droite de l'instruction préquête.

Exercice 3 :

Soit la grammaire suivante des expressions booléennes :

$$\wedge \rightarrow A \wedge B \mid A \wedge A \mid \text{Vrai} \mid \text{Faux}$$

- Donner l'arbre de dérivation pour le mot : non faux ou vrai et vrai
- Que pouvez-vous conclure.

Exercice 4 :

Soit la grammaire suivante des expressions booléennes :

$$A \rightarrow A \vee B \mid B$$

$$B \rightarrow B \vee C \mid C$$

$$C \rightarrow \text{non } C \mid (A) \mid \text{Vrai} \mid \text{Faux}$$

- Donner l'arbre de dérivation pour le mot : non faux ou vrai et vrai
- Que pouvez-vous conclure.

Exercice 5 :

Soit la grammaire suivante :

$$E \rightarrow E + E \mid E \cdot E \mid id$$

- Montrer que cette grammaire est ambiguë.
- Construire une grammaire équivalente non ambiguë.
- Éliminer la récursivité à gauche et factoriser si nécessaire.
- Donner la table d'analyse de la nouvelle grammaire. Est-elle LL(1) ?

Exercice 6 :

Soit la grammaire suivante :

$$B \rightarrow B \vee B \mid B \wedge B \mid \neg B \mid id \mid \text{vrai} \mid \text{faux}$$

- Montrer que cette grammaire est ambiguë.
- Construire une grammaire équivalente non ambiguë.
- Éliminer la récursivité à gauche et factoriser si nécessaire.
- Donner la table d'analyse de la nouvelle grammaire. Est-elle LL(1) ?
- Expliquer le comportement de l'analyseur sur le mot : (id)id $\Rightarrow \neg id$

Exercice 7 :

Soit la grammaire

$$\begin{cases} S \rightarrow AB \mid a \mid \lambda \\ A \rightarrow SA \mid \lambda \mid B \\ B \rightarrow SB \end{cases}$$

- Montrer la récursivité à gauche
- Factoriser si nécessaire
- La grammaire obtenue est-elle LL(1)

Exercice 8 :

Soit la grammaire

$$\begin{cases} S \rightarrow ab(T) \\ T \rightarrow TS \mid S \\ U \rightarrow SU \end{cases}$$

- Montrer que la grammaire est-elle LL(1).
- Éliminer la récursivité à gauche et factoriser si nécessaire.
- Montrer que la nouvelle grammaire est LL(1). Donner la table d'analyse.
- Expliquer le comportement de l'analyseur sur les mots : m1 = (u, a) ; m2 = (s, a, (b, a)u)

Exercice 9 :

Soit la grammaire suivante des expressions booléennes :

$$A \rightarrow A \wedge B \mid A \wedge A \mid \text{Vrai} \mid \text{Faux}$$

- Donner l'arbre de dérivation pour le mot : non faux ou vrai et vrai
- Que pouvez-vous conclure.

Exercice 10 :

Soit la grammaire suivante :

$$E \rightarrow E + E \mid E \cdot E \mid id$$

en sil

$$\begin{array}{l} \text{3:} \\ \text{A} \rightarrow Ab \mid a \mid AA \\ A \rightarrow Sa \mid Ac \mid B \\ B \rightarrow Sd \end{array}$$

1. ordonner les non-terminaux S, A, B.

pour $i = 2$ pas de récursivité à gauche.

pour $i = 2 \quad j = 1$

$A \rightarrow Sa \mid Ac \mid B$ remplacer par.

$$A \rightarrow Ab \mid \underline{aa} \mid \underline{AA} \mid \underline{Ac} \mid \underline{B}$$

Éliminer la récursivité immédiate.

$$A \rightarrow aa \mid A' \mid BA'$$

$$A' \rightarrow ba \mid Aa \mid CA' \mid \epsilon$$

pour $i = 3 \quad j = 1$

$B \rightarrow Sd$ remplacer par

$$B \rightarrow Abd \mid ad \mid AA^d$$

$j = 2$

$B \rightarrow Abd \mid ab \mid AA^d$ remplacer par

$$\begin{aligned} B \rightarrow & aaA'bd \mid BA'bd \mid ab \mid aaA'aA'd \mid \\ & aaA'BA'd \mid BA'BA'd \mid BA'aA'd \mid \end{aligned}$$

Éliminer la récursivité immédiate.

$B \rightarrow aaA'bdB' \mid abB' \mid aaA'aaA'dB' \mid aaA'\epsilon$

$B' \rightarrow A'bdB' \mid A'BA'dB' \mid A'aaA'dB' \mid \epsilon$

grammaire résultante:

$s' \rightarrow Ab \mid a \mid AA$

$A \rightarrow aaA' \mid BA'$

$A' \rightarrow baA' \mid AaA' \mid cA' \mid \epsilon$

$B \rightarrow aaA'bdB' \mid abB' \mid aaA'aaA'dB' \mid aaA'BA'dB'$

$B' \rightarrow A'bdB' \mid A'BA'dB' \mid A'aaA'dB' \mid \epsilon$

2. Factorisation

Simple en se basant sur l'rigole.

$A \rightarrow a\beta \mid a\alpha$

$\Leftrightarrow A \rightarrow a^+ B^-$

$B^- \rightarrow B \mid \alpha$

3. Pour montrer que cette grammaire est LL(1)

il faut construire la table d'analyse et décliner.

n° 6:

Grammaire :

$$B \rightarrow B \vee B \mid B \wedge B \mid B \Rightarrow B \mid T B \mid (B) \mid id \mid vrai \mid faux$$

1. Soit la chaîne vrai \wedge vrai \Rightarrow faux.

admet deux dérivation le plus à gauche.

$$(B \rightarrow B \Rightarrow B \rightarrow B \wedge B \Rightarrow B \dots)$$

$$B \rightarrow vrai \wedge B \rightarrow vrai \wedge B \rightarrow vrai \wedge B \Rightarrow B \dots)$$

d'où la grammaire est ambiguë.

2. En principe que l'exercice n° 3.

on a $T > \wedge > \vee > \neg$ $\xrightarrow{\text{associativité}}$ (priorité).

commençons par le plus faible à partir de l'axiomatique.

$$B \rightarrow T \Rightarrow B \mid T \quad (= \text{associativité à gauche})$$

$$T \rightarrow T \vee F \mid F \quad ("T" \text{ ajouté à droite})$$

$$F \rightarrow F \wedge R \mid R$$

$$R \rightarrow T \mid R \mid (B) \mid id \mid faux \mid vrai$$

(\vee est associatif à droite " F " ajouté à gauche)

(\wedge " " " " " " R " " " ").

⇒ Elimination d'un caractère

$$B \rightarrow T \Rightarrow B \mid T$$

$$T \rightarrow FT'$$

$$T' \rightarrow \vee FT' \mid \epsilon$$

$$F \rightarrow RF'$$

$$F' \rightarrow \wedge RF' \mid \epsilon$$

$$R \rightarrow \exists R \mid (B) \mid id \mid \text{fonction}$$

Factorisation: $B \rightarrow T \Rightarrow B \mid T$

$$\Leftrightarrow B \rightarrow TB'$$

$$B' \rightarrow = B \mid \epsilon$$

grammaire résultante:

$$B \rightarrow TB'$$

$$B' \rightarrow = B \mid \epsilon$$

B. avancé

$$T \rightarrow FT'$$

$$T' \rightarrow \vee FT' \mid \epsilon$$

$$F \rightarrow RF'$$

$$F' \rightarrow \wedge RF' \mid \epsilon$$

$$R \rightarrow \exists R \mid (B) \mid id \mid \text{fonction}$$

cette grammaire → non-analyseable

→ non-recursivité algébrique

→ Factorisable

⇒ LH(1).

table d'analyse:

$$\text{premier}(B) = \{\Rightarrow, \varepsilon\}$$

$$\text{premier}(T') = \{\vee, \varepsilon\}$$

$$\text{premier}(F') = \{\wedge, \varepsilon\}$$

$$\text{premier}(R) = \{T, (, id, \text{faux}, \text{vrai})\}$$

$$\text{premier}(F) = \text{premier}(R) = \{T, (, id, \text{faux}, \text{vrai})\}$$

$$\text{premier}(T) = \text{premier}(F) = \{T, (, id, \text{faux}, \text{vrai})\}$$

$$\text{premier}(B) = \text{premier}(T) = \{T, (, id, \text{faux}, \text{vrai})\}$$

$$\text{suivant}(B) = \{\$,)\}$$

$$\text{suivant}(B') = \text{suivant}(B) = \{\$,)\}$$

$$\text{suivant}(T) = \{\text{premier}(B') \mid \varepsilon\} \cup \text{suivant}(B) = \{\Rightarrow, \$,)\}$$

$$\text{suivant}(F) = \{\text{premier}(T') \mid \varepsilon\} \cup \text{suivant}(T) = \{\vee, \Rightarrow, \$,)\}$$

$$\text{suivant}(T') = \{\text{suivant}(T)\} = \{\Rightarrow, \$,)\}$$

$$\text{suivant}(F') = \{\text{suivant}(F)\} = \{\vee, \Rightarrow, \$,)\}$$

$$\text{suivant}(R) = \{\text{premier}(F) \mid \varepsilon\} \cup \text{suivant}(F) =$$

$$= \{\wedge, \vee, \Rightarrow, \$,)\}$$

| | termi
n | id | (|) | vrai | faux | \Rightarrow | ? | ^ | v |
|------|------------|---------------------|---------------------|---------------------------|----------------------|----------------------|---|---------------------|-----------------------------|------------------------------|
| B | | $B \rightarrow TB'$ | $B \rightarrow TB'$ | | $B \rightarrow TB'$ | $B \rightarrow TB'$ | $B \rightarrow BB'$ | $B \rightarrow FB'$ | | |
| B' | | | | $B' \rightarrow \epsilon$ | | | $B' \Rightarrow B$ | | | $B' \rightarrow$ |
| T | | $T \rightarrow FT'$ | $T \rightarrow FT'$ | | $T \rightarrow FT'$ | $T \rightarrow FT'$ | | $T \rightarrow FT'$ | | |
| T' | | | | $T' \rightarrow \epsilon$ | | | $T \rightarrow \epsilon$ | | | $T \rightarrow VF'$ |
| F | | $F \rightarrow RF'$ | $F \rightarrow RF'$ | | $F \rightarrow RF'$ | $F \rightarrow RF'$ | | $F \rightarrow RF'$ | | |
| F' | | | | $F' \rightarrow \epsilon$ | | | $F' \rightarrow \epsilon$ | | $F' \rightarrow \wedge RF'$ | $F' \rightarrow \epsilon F'$ |
| R | | $R \rightarrow id$ | $R \rightarrow (B)$ | | $R \rightarrow vrai$ | $R \rightarrow faux$ | | $R \rightarrow ?R$ | | |

Ré: Lh(1): pas de cases définies façon multiple

chaîne ($id \wedge id$) \Rightarrow / / /

| pile | entrée | action |
|---------------------|------------------------------------|-------------------------|
| \$ B | $(id \wedge id) \Rightarrow id \#$ | (enlever B, mettre TB#) |
| \$ B'T | $(id \wedge id) \Rightarrow id \#$ | (" T, mettre T'F#) |
| \$ B'T'F | $(id \wedge id) \Rightarrow id \#$ | (" F, mettre RF#) |
| \$ B'T'F'R | $(id \wedge id) \Rightarrow id \#$ | (" R, mettre (B)) |
| \$ B'T'F'BC | $(id \wedge id) \Rightarrow id \#$ | (" " "(, avancer) |
| \$ B'T'F'B | $(id \wedge id) \Rightarrow id \#$ | (" B, mettre TB#) |
| \$ B'T'F'BT | $(id \wedge id) \Rightarrow id \#$ | (" T, mettre FF#) |
| \$ B'T'F'BT'F | $(id \wedge id) \Rightarrow id \#$ | (" F, mettre RF#) |
| \$ B'T'F'BT'FR | $(id \wedge id) \Rightarrow id \#$ | (" R, mettre id#) |
| \$ B'T'F'BT'FR# | $(id \wedge id) \Rightarrow id \#$ | (" id, avancer) |
| \$ B'T'F'BT'FR' | $(id \wedge id) \Rightarrow id \#$ | (" F', mettre NRF#) |
| \$ B'T'F'BT'F' | $\wedge id \Rightarrow id \#$ | (" ^, avancer) |
| \$ B'T'F'BT'F'R | $\wedge id \Rightarrow id \#$ | (" R, mettre id#) |
| \$ B'T'F'BT'F'R | $id \Rightarrow id \#$ | (" id, avancer) |
| \$ B'T'F'BT'F'R; id | $id \Rightarrow id \#$ | (" id, mettre E#) |
| \$ B'T'F'BT'F'R; E | $) \Rightarrow id \#$ | (" F', mettre E#) |
| \$ B'T'F'BT'F'R; E | $) \Rightarrow id \#$ | (" T', mettre E#) |
| \$ B'T'F'BT'F'R; E | $) \Rightarrow id \#$ | (" B', mettre E#) |
| \$ B'T'F'BT'F'R; E | $\Rightarrow id \#$ | (" " ", avancer) |
| \$ B'T'F'BT'F'R; E | $\Rightarrow id \#$ | (" F', mettre E#) |
| \$ B'T'F'BT'F'R; E | $\Rightarrow id \#$ | (" T', mettre E#) |
| \$ B'T'F'BT'F'R; E | $\Rightarrow id \#$ | (" B', mettre E#) |
| \$ B'T'F'BT'F'R; E | $\Rightarrow id \#$ | (" id, avancer) |
| \$ B'T'F'BT'F'R; E | $\Rightarrow id \#$ | (" B, mettre TF#) |
| \$ B'T'F'BT'F'R; E | $\Rightarrow id \#$ | (" T, mettre FT#) |

| | | |
|------------|---------|------------------|
| \$BT | T'io \$ | C" |
| \$BT'F | T'ib \$ | C" F, |
| \$B'T'F'R | T'id \$ | C" R, mettre |
| \$BT'F'R7 | T'id \$ | C" T, avancer |
| \$BT'F'R | id \$ | C" R, mettre id |
| \$BT'F' id | id \$ | C" id, avancer |
| \$BT'F' | \$ | C" F', mettre e) |
| \$BT | \$ | C" T, mettre e) |
| \$B' | \$ | C" B', mettre e) |
| \$ | | accepter |

X-

$$S \rightarrow a \mid b \mid T$$

$$T \rightarrow T, S \mid S$$

Non. (Récurseur à gauche.)

$$S \rightarrow a \mid b \mid T$$

$$T \rightarrow ST'$$

$$T' \rightarrow , ST' \mid \epsilon$$

B. $\text{Premier}(S) = \{a, b, "\")\}$

$$\text{premier}(T') = \{ , , \epsilon \}$$

$$\text{premier}(T) = \text{premier}(S) \cup \{ \epsilon \} = \{a, b, "\")\}$$

$$\text{suivant}(T) = \{"\")\}$$

$$\begin{aligned}\text{suivant}(S) &= \{\$\} \cup \{\text{premier}(T') \cup \epsilon\} \cup \{\text{suivant}(T)\} \\ &= \{\$, , , "\")\}\end{aligned}$$

$$\text{suivant}(T') = \{\text{suivant}(T)\} = \{"\")'\}$$

Table d'analyse

| | a | "b" | (|) | , | # |
|----|-------------------|---------------------|---------------------|---------------------|---------------------------|----------------------|
| S | $S \rightarrow a$ | $S \rightarrow b$ | $S \rightarrow (T)$ | | | |
| T | | $T \rightarrow ST'$ | $T \rightarrow ST'$ | $T \rightarrow ST'$ | | |
| T' | | | | | $T' \rightarrow \epsilon$ | $T' \rightarrow ST'$ |

Année Universitaire : 2020-2021

Examen Session Principale

Niveau d'étude : L2CS

Semestre : 2

Matière : Techniques de Compilation

Documents : Non Autorisés

Nombre de pages : 02

Exercice 1 : (9 points)

Soit la grammaire G1 suivante, d'axiomé S et l'ensemble des terminaux

$$V_T = \{w, d, ;, id, \{, \}, >, =, +, -\}$$

$$S \rightarrow wCdS \mid S ; S \mid id = E$$



$$C \rightarrow id \text{ OP id}$$

$$\text{OP} \rightarrow < \mid >$$

$$E \rightarrow E + id \mid - E \mid id$$

1. Donner une dérivation à gauche pour le mot : w id < id d id = -id ?
2. Cette grammaire est-elle récursive à gauche? Si oui, éliminer la récursivité.
3. Cette grammaire est-elle non factorisée à gauche? Si oui, factoriser-la.
4. Déterminer les ensembles PREMIER et SUIVANT
5. Construire la table d'analyse
6. La grammaire est-elle LL(1)? Justifier

Exercice 2 : (4points)

Soit la grammaire G2 suivante, d'axiomé Inst et l'ensemble des terminaux

$$V_T = \{if, then, a, :=, else\}$$

$$Inst \rightarrow if \text{ ExpB then Inst ElseI} \mid E := E \mid E := E \text{ OPA E}$$

$$ElseI \rightarrow else Inst \mid \epsilon$$

$$E \rightarrow a$$

$$\text{ExpB} \rightarrow E \text{ OPB E}$$

$$\text{OPB} \rightarrow < \mid >$$

$$\text{OPA} \rightarrow + \mid -$$

Soit la chaîne $\omega = \text{if } a < a \text{ then } a := a \text{ else } a := a + a$

1. Présenter l'analyse ascendante de cette chaîne dans une table comme suit :

| Pile | Chaine entrée | Action |
|------|---------------|--------|
| | | |

2. Donner (si possible) les dérivation droites sur la chaîne.

Exercice 3 : (7 points)

Soit la grammaire G3 suivante d'axiome S et l'ensemble des terminaux

$$V_T = \{*, /, \text{int}\}:$$

$$\begin{aligned} S &\rightarrow E \\ E &\rightarrow E^*T \mid T \\ T &\rightarrow T/F \mid F \\ F &\rightarrow \text{int} \end{aligned}$$

1. Sachant que int est un entier qui peut être positif, négatif ou nul,

□ Donner l'arbre syntaxique pour le mot $2^* - 3/1$

□ 2) En utilisant un attribut sign, transformer G3 en DDS qui a pour objectif de déterminer et afficher le signe (POSITIF ou NEGATIF ou ERREUR) de l'expression, sachant que la valeur de int est récupérée à partir de l'analyseur lexical.

Exemples : Pour le mot : $2^* - 3$, le résultat : NEGATIF

Pour le mot : $-4^*5/-2$, le résultat : POSITIF

NB : On ne demande pas la valeur de l'expression, seulement son signé.

3. Expliquer brièvement l'intérêt de chaque attribut utilisé dans la DDS proposée ainsi que son type.

□ 4. Donner l'arbre décoré pour le mot $2^* - 3/1$

?

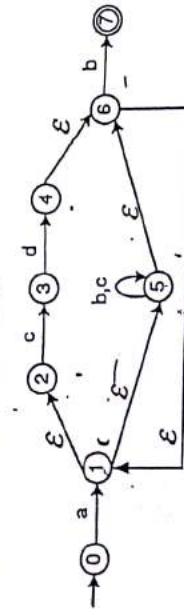
Exercice 1 :

Soit l'expression régulière suivante : $(c \mid a^*b)^*d(b \mid cd)$.

Proposez un automate reconnaissant le même langage ($X = \{a, b, c, d\}$).

Exercice 2 :

Soit $X = \{a, b, c, d\}$. Soit l'automate décrit comme suit :



1. Cet automate est-il déterministe ? Justifier.
2. S'il est non déterministe, déterminiser-le.

Exercice 3 :

Soit le langage X qui décrit l'unité lexicale Opel (Opérateur Relationnel) et OpAff (Opérateur d'Affectation) comme suit :

| Unité lexicale | Lexème | Valeur d'attribut |
|----------------|---------|-------------------|
| Opel | \leq | ✓ |
| Opel | $>$ | ✓ |
| Opel | $\leq:$ | ✓ |
| Opel | $\geq:$ | ✓ |
| Opel | $\geq:$ | ✓ |
| Opel | $::$ | ✓ |
| Opel | $::$ | ✓ |
| OpAff | \neq | ✓ |
| OpAff | $:$ | Affect |

Donnez le diagramme de transition permettant de reconnaître un opérateur relationnel et l'opérateur d'affectation (dans le même diagramme)

Exercice 4 :

On considère le mini-langage naturel L. L'alphabet du langage L est formé :

- des lettres [a,...,z]
- des 6 signes de ponctuation (signePonct) {., ;, ', !, ?}
- des articles définis (articleDéf) sont : le, la, les et l'
- des articles indéfinis (articleIndef) sont : un, une, des et d'
- des blancs

- 1) Donner la définition régulière de : lettre, signePonct, articleDéf, articleIndef et blanc.
- 2) Donner le diagramme de transition permettant de reconnaître l'unité lexicale articleDéf et articleIndef (sur un même diagramme).