U.S.T.H.B - Département Informatique
Module de Compilation - Section : A, B et C
An· ?004 / 2005

Série N° 1
Analyse lexicale

Exercice 1 :
Soit la table de transition suivante :

Etats	Symbole d'entrée	
	a	b
0	1, 2	
1	1, 2, 3	2, 3, 0
2	2	
3	0, 2	

L'état initial est 0 et l'unique état final est 3.
a-Construire un automate fini déterministe à partir de la table de transition.
b-Donner la grammaire régulière à droite correspondante.

Exercice 2 :

Un train de marchandise doit avoir une ou deux locomotives suivies de un ou plusieurs wagons de marchandise suivis d'une voiture de garder.
Faire l'automate correspondant ainsi que l'algorithme de reconnaissance d'une train de marchandise.

Exercice 3 :

a-Donner une grammaire régulière à gauche engendrant le langage forme de `a' et de 'b', mais ne contenant pas la sous chaine 'aba'.
b-déduire l'automate déterministe correspondant.
c-analyser les chaines bbaab# et aabaa#.

Exercice 4 :

Soit la portion de programme suivante :

```
Program Test
Var n :Int
Function Fib(i :Int, j :Int) :Int
Begin If n=0 Then Return i
                Else If n=1 Then Return j
                                Else Begin n:=n-1; Fib(j, i+j) End
End.
```

a-Donner les entités lexicales de la portion de programme.

b-Donner l'automate lexical, simple et déterministe décrivant les entités de la portion de programme.
c-Ecr ''analyseur lexical correspondant.

Exerc. .e 5 :

On veut analyser le langage L défini sur $\{0, 1=\}^*$ et présentant les caractéristiques suivantes :
-Le langage est formé des entités de type X, Y et Z (chaines binaires).
-Une entité X commence par '0' et ne contient pas deux '0' consécutifs.
-Une entité Y commence par un '1' qui est suivi d'un nombre pair de '0' (il y a au moins deux).
-Une entité Z commence par deux '0' ou bien deux '1' suivis par une séquence de '0' et de '1' ne contenant la sous chaine '101'.
Les entités du langage sont séparées par un ou plusieurs blancs.
a-Construire un automate déterministe reconnaissant les entités X, Y et Z.
b-Ecrire un algorithme qui fait l'analyse lexicale du langage.

U.S.T.H.B - Département Informatique
Module de Compilation - Section : A, B et C
Année : 2004 / 2005

Série N° 2
Analyse Syntaxique
Notions générales

Exercice 1 :

Eliminer la récursivité gauche dans les grammaires suivantes :

G1 : $S \rightarrow aAba / AB$
$A \rightarrow Bb / a$
$B \rightarrow Sa / c$

G2 : $S \rightarrow a / aA$
$A \rightarrow Sc / Ad / c$

G3 : $S \rightarrow Abc / Baa$
$A \rightarrow Aab / Ab / a$
$B \rightarrow Bb / b$

G4 : $S \rightarrow a / A$
$A \rightarrow AB / BC$
$B \rightarrow b / Cb$
$C \rightarrow Ab / S$

Exercice 2:

Soient les deux grammaires G_1 et G_2 :

G1 : $S \rightarrow bAab / AB$
$A \rightarrow aA / \varepsilon$
$B \rightarrow cB / \varepsilon$

G2 : $S \rightarrow A / B$
$A \rightarrow aB / bSB / \varepsilon$
$B \rightarrow AB / Ba$

a- Rendre G_1 et G_2 ε-libre
b- Eliminer la récursivité gauche dans G_2

Exercice 3 :

On considère la grammaire G : $S \rightarrow Abc / Baa$
$A \rightarrow AaB / Ab / \varepsilon$
$B \rightarrow bB / \varepsilon$

a- Transformer G sous forme F.N.C
b- Eliminer la récursivité gauche
c- Transformer G sous forme F.N.G

Exercice 4 :

On donner la grammaire suivante :

G : $S \rightarrow bB / aA / c$
$A \rightarrow bS / aAB / b$
$B \rightarrow bBA / aS / a$

a- Transformer G sous forme F.N.G
b- Analyser par la méthode descendante les chaines aabac# et bbac#
c- Même question avec retour arrière

U.S.T.H.B
Faculté de Génie Electrique
Département d'Informatique
Module de Compilation
Année 2004/2005

Série n°3
Analyse Syntaxique
Descendante déterministe

Exercice 1:

Construire les ensembles 'debuts' et 'suivants' pour chaque non-terminal pour les grammaires :

G_1: S→ Abc / Baa
A→ Aab / Ab / a
B→ Bb / b

G_2: S→ bAab / AB
A→ Ba / b
B→ Sb / c

G_3: S→ Bac / a
A→ bSBa / ε
B→ AbcS / ε

Exercice 2:

les grammaires suivantes sont-elles LL(1)?

G_1: S→ AB / aSb / CSB
A→ bA / ε
B→ dB / ε
C→ Cc / ε

G_2: S→ ASB / ε
A→ aAc / c
B→ bBA / ε

G_3: S→ A / B / a
A→ bBA
B→ dB / ε

Montrer que G_2 n'est pas LL(k).

Exercice 3:

Soit la grammaire:
G: S→ ASBa / ab
A→ aA / c
B→ bB / ε
- G est-elle LL(k)? Si oui analyser les chaines : acabba# et cbba#.

Exercice 4:

Soit G: S→ a / b(T)
T→ T,S / S
a- Eliminer la récursivité gauche dans G
b- Factoriser éventuellement la grammaire obtenue en a
c- La grammaire obtenue est-elle LL(1) ?
d- Ecrire un analyseur syntaxtique basé sur la descente récursive pour la grammaire obtenue en b
e- Analyser avec chacune des deux méthodes la chaine : b(a,a)#.

U.S.T.H.B
Faculté de Génie Electrique
Département d'Informatique
Module de Compilation
Année 2004/2005

Série n°4
Analyse Ascendante
Précédence Simple

Exercice 1:

Les grammaires suivantes sont-elles de Précédence Simple?

G_1: S→ aABd
A→ bb
B→ c

G_2: S→ {AbB / a
A→ a,A / B
B→ (A) / }

- Analyser les chaines abbcd# et abd# avec G_1?
- Sinon transformer, et analyser la chaine {a,}b}# avec G2

Exercice 2:

Soit G: S→ a / b / (T)
T→ S,T / S

a. G est-elle de PS ?
b. Sinon transformer, et ananlyser la chaine ((a,b),a)#

Exercice 3:

Soit la grammaire G: S→ {S, A} / A
A→ A (-A) / 0 / 1

a. G est-elle de PS ?
b. Peut-on la transformer?
c. Ananlyser la chaine 0(-1){0,(-1)}#
d. En examinant les règles de cette grammaire G, expliquer pourquoi, elle ne peut être de PS, (Sans construction de la table de précédence).

U.S.T.H.B - Département Informatique
Module de Compilation - Section : A et C
Année : 2004/ 2005

Série N° 5
Analyse Syntaxique Ascendante

Exercice 1 :
Soient les grammaires suivantes :

G1 : $S \rightarrow a/bA$, $A \rightarrow Sc / \varepsilon$

G2 : $S \rightarrow Sa / c / A$, $A \rightarrow bA / \varepsilon$

G3 : $S \rightarrow A/B/a$, $A \rightarrow bBA / c$, $B \rightarrow bBca / \varepsilon$

G4 : $S \rightarrow RT$, $T \rightarrow cT / c$, $R \rightarrow ARB / c$, $A \rightarrow 0D$, $B \rightarrow D1$, $D \rightarrow 1$

G5 : $S \rightarrow aB/abA / \varepsilon$, $A \rightarrow aA / a$, $B \rightarrow b$

1. Sont-elles LR(k) par la méthode des contextes ?
2. Construire la table LR(k) pour G2, puis analyser les chaines bbaa et cab.
3. Construire la table LR(k) pour G3, puis analyser les chaines bbcac et baa.

Exercice 2:
On considère la grammaire paramétrée suivante :

G_n : $S \rightarrow Sb^nc / AB$ $\quad A \rightarrow aA / b^n$ $\quad B \rightarrow Bb / \varepsilon$ $\quad n \geq 0$

1. G est-elle LR(K) par la méthode des contextes ?
2. Construire la table LR(k) pour G_1, puis analyser les chaines abb et ca.

Exercice 3 :
Les grammaires suivantes sont-elles LR(1) ? SLR(1) par la méthode des contextes ?

G1 : $S \rightarrow A0A / B1B0$, $A \rightarrow aA / \varepsilon$, $B \rightarrow Bb / \varepsilon$

G2 : $S \rightarrow AaAb / BbBa$, $A \rightarrow \varepsilon$, $B \rightarrow \varepsilon$

Exercice 4 :
Les grammaires suivantes sont-elles SLR(1) par la méthode des items ? Sont-elles LALR(1) ? LR(1) ?

G1 : $S \rightarrow Aa / bAc / Bc / bda$, $A \rightarrow d$, $B \rightarrow d$

G2 : $S \rightarrow AS / b$, $A \rightarrow SA / a$

Exercice 5 :
Soit la grammaire suivante : $S \rightarrow BA$ $\quad A \rightarrow aB / \varepsilon$ $\quad B \rightarrow Bb / \varepsilon$

1. G est-elle LR(1), SLR(1) par la méthode des items ?
2. Est-elle LALR(1) ? Comparer les trois tables d'analyse ?

Exercice 6 :
Soit la grammaire G suivante :

$E \rightarrow E \cap E / E \cup E / E^* / E+ / a / b$

a- G est-elle ambiguë ? est-elle LR(k) ?
b- Construire la table d'analyse SLR(1) par la méthode des items.
c- Peut-on éliminer les multidéfinitions en adoptant certaines conventions ? Si oui, analyser la chaine $a \cap b \cup a+$.

Exercice 7 :
Soit la grammaire suivante : $S \rightarrow Ab / Bc$ $\quad A \rightarrow aA / \varepsilon$ $\quad B \rightarrow bB / \varepsilon$

1. G est-elle LR(2) par la méthode des items ?
2. Donner la table d'analyse et analyser la chaine bbc?

U.S.T.H.B
Département Informatique
Module de Compilation
Année 2004/2005

Série n°7
Traduction dirigée par la syntaxe

Exercice 1:

Considérons l'instruction suivante :
<inst-repeat> → repeat <inst> until <cond>
Donner le schéma de traduction sous forme préfixée en utilisant un analyseur ascendant.

Exercice 2:

Considérons l'instruction suivante :
<do-while> → Do id := if <cond1> then <exp1> else <exp2> while <cond2>
Donner le schéma de traduction sous forme postfixée en utilisant un analyseur descendant.

Exercice 3:

Soit l'instruction :
Id := moyenne (<exp1>,<exp2>, … <expn>)
Donner le schéma de traduction sous forme de quadruplets avec une analyse ascendante.

Exercice 4:

Soit l'instruction :
Id := Mult ((exp1,cond1), …., (expn,condn))
On multiplie les expressions dont les conditions sont vraies. Si toutes les conditions sont fausses, le résultat est 1.
Donner le schéma de traduction sous forme de triplés avec une analyse descendante.

Exercice 5:

<inst-case> → case exp of <list-inst-etiq> end ;
<list-inst-etiq> → <list-inst-etiq> ;<inst-etiq>/<inst-etiq>
<inst-etiq> → <list-etiq> :<inst>
<list-etiq> → <list-etiq>,etiq / etiq
~~<list-etiq> → <list-etiq>,etiq / etiq~~

Donner le schéma de traduction sous forme de triplés dans le cas d'une analyse descendante.

Exercice 6:

Soit la grammaire des expressions booléennes suivante :
EL→ EL or EL/ EL and EL/ id/ Not EL/ True/ False
Donner le schéma de traduction sous forme de triplés dans le cas d'une analyse ascendante.

U.S.T.H.B
Faculté d'Informatique et Electronique
Module de Compilation
Année 2004/2005

Série n°8

Exercice 1:
On considère le tableau suivant A[-1 :1 ; 0 :2 ; -2 :0]

a- Donner la représentation ligne/ligne des éléments de A en mémoire dans une zone contiguë.
b- .....colonne/colonne
c- .....sous forme de schéma d'Illife :
   c1- ligne/ligne
   c2- colonne/colonne

Exercice 2:
Soit la matrice creuse d'ordre n suivante, n est impaire.

M = | 0 | x | x | 0 |

On veut représenter dans une zone contiguë que les éléments non nuls de M.

a- Donner la relation liant i et j, si M[i,j]≠0
b- Donner l'adresse de l'élément M[i,j] si ils rangés ligne/ligne
c- On associe à chaque éléments du tableau un bit :
   bit(k) =0 si M[i,j]=0
   bit(k)=1 sinon

Les éléments non nuls sont rangés ligne/ligne dans une zone contiguë. Donner l'adresse de l'élément M[i,j].

Exercice 3:
Même question pour la matrice :

M =

0	x
x	0

U.S.T.H.B
Faculté d'Informatique et Electronique
Module de Compilation
Année 2004/2005

Série n°9

Exercice 1:
Soit le programme Fortran suivant :

```
Pragram
   Integer A, B, C, F, G
   Integer D(5), E(3)
   Common A, G
   Equivalence G, D(2)
   Equivalence D(4), E(1)
   .....
   Call Sub(B, F)
   Stop
End.
Subroutine Sub(x, y)
   Integer H, I, B(5), L(3), J
   Common H, I
   Equivalence B(3), J
   Equivalence L(1), B(4)
   .....
   Return
End.
```

Donner les zones de données du PP, du SP et du Common

Exercice 2:
Soit le programme suivant :

```
Begin
   Real A, E; Integer array B1[1:A]; Integer K;
L1: Begin
       Integer array B2[1:E]; Integer I;
       Procedure X(C, D); Real C,D;
    Begin
          Array B3[1:K; 1:2*K];
     L3: B3[I, 1]:= B1[I]+A;
       End;
    L2: B1[I]:=I-A;
   End;
A:=E+K;
End;
```

Donner le contenu de la pile de données aux instructions L1, L2 et L3 .
Calculer les adresses absolues des variables de L3.

Exercice 3 :

Eliminer les sous-expressions communes dans la portion de programme suivante:

1-i :=m-1
2-j :=n
3-t_1:=4*n
4-v:=a[t_1]
5-i :=i+1
6-t_2 :=4*i
7-t_3 :=a[t_2]
8-Si t_3<v aller à (5)
9-j :=j-1
10-t_4 :=4*j
11-t_5 :=a[t_4]
12-Si t_5>v aller à (9)
13-Si i>j aller à (23)
14-t_6 :=4*i
15-x :=a[t6]
16-t_7 :=4*i
17-t_8 :=4*j
18-t_9 :=a[t_8]
19-a[t_7]:=t_9
20-t_{10}:=4*j
21-a[t_{10}] :=x
22-aller à (5)
23-t_{11} :=4*i
24-x :=a[t_{11}]
25-t_{12}:=4*i
26-t_{13}:=4*n
27-t_{14}:=a[t_{13}]
28-a[t_{12}] :=t_{14}
29-t_{15}:=4*n
30-a[t_{15}] :=x