Partiel : Compilation
ISIMA 2

P. Wodey

Mercredi 8 Décembre 2010

1 Grammaires

Exercice 1 *Nous considérons la grammaire ci-dessous où l'axiome est le non terminal* `A`.

```
A -> [B : C] | D
D -> a | [D]
B -> B : C | C
```

Questions :

1. *quels sont les symboles non terminaux et les symboles terminaux de cette grammaire ?*
2. *calculer les ensembles PREMIER pour les non terminaux et les différentes parties droites de règles, ainsi que les ensembles SUIVANT des symboles non terminaux;*
3. *en construisant l'automate et la table d'analyse, déterminer si la grammaire est SLR(1) ?*
4. *supprimer la récursivité à gauche du non terminal* `B`. *La grammaire obtenue est-elle LL(1) ?*

2 Lex et yacc

Exercice 2 Extension des expressions

Dans cette partie les symboles non terminaux S, LI, I, E, T et F sont ceux vus en cours et en TP. Rappel de l'ancienne grammaire :

```
S : LI ;
LI : I | LI ';' I;
I : IDENT AFFECT E | READ'('IDENT')' | WRITE '('IDENT')';
E : E '+' T | E '-' T | T;
T : T '*' F | T '/' F | F;
F : IDENT | ENTIER | '(' E ')' | LET IDENT AFFECT E IN E LET ;
```

Nous envisageons d'ajouter les instructions conditionnelles et tant que (while) à celles vues en TP. Ceci consiste à ajouter deux nouvelles instructions (dérivations de I) dont voici des exemples :

```
if (b + 2) then d := a; else c := a-5; e := 4; endif;
while a do
  c := 2*a;
  a := a-1;
enddo
```

*Si a est inférieur à b+2 alors on exécute l'instruction d := a, sinon on exécute c := a - 5 suivie de e:=4. Tant que a on exécute la séquence c:= 2*a suivi de a := a-1. Les symboles "if", "then", "else", "endif", "while", "do" et "enddo"sont des nouveaux mots-clés.*

Questions :

1. *que faut-il faire au niveau lexical pour ajouter les lexèmes "if", "then", "else", "endif", "while", "do" et "enddo" ? Indiquer les modifications dans le fichier analex.l et anasyn.y ;*

2. *donner les règles de grammaire de I avec les nouvelles dérivations ;*

3. *nous considérons que nous créons des classes pour les arbres d'instructions I et les listes d'instructions LI. Décrire les deux classes abstraites et les classes dérivées de pour I avec les attributs;*

4. *les non terminaux I et LI sont maintenant typés en tant que pointeur sur un arbre instruction (classes I et LI). Donner la partie du fichier anasyn.y (yacc) qui décrit le typage des non terminaux concernés;*

5. *donner l'action sémantique de création du noeud de l'arbre pour les différentes instructions ;*