Maîtrise d'Informatique

ANALYSE SYNTAXIQUE ET COMPILATION (ASC)

Durée 3 heures, documents manuscrits autorisés

Session de Septembre 2000

Problème 1 : Grammaire de langage de programmation *(4 pts)*

On considère un langage de programmation très simple qui n'utilise que 6 variables `a,b,c,x,y,z` et trois fonctions `f,g,h`. Les fonctions f et g sont d'arité 1, la fonction h est d'arité 2. Un programme commence par le mot réservé `begin` et se termine par `end`. Il contient une suite d'instructions qui sont soit d'affichage soit d'affectation. Les instructions sont séparées par des `;` . Une instruction d'affichage est une expression : variable ou appel d'une fonction. L'appel d'une fonction s'écrit comme dans tous les langages habituels avec des parenthèses qui entourent les paramètres, on sépare dans h les 2 paramètres par une virgule. Un exemple d'appel de fonction est le suivant :

```
f(h(a,g(b)))
```

Une instruction d'affectation contient le signe `=`, une variable à sa gauche et une expression à sa droite. Par exemple :

```
x = g(h(y,b))
```

Un exemple de programme qui contient toutes les possbilités du langage est ainsi :

```
begin
f(a);
z = h(a,x);
y = f(g(h(a,b)));
y;
z
end
```

Question On vous demande d'écrire une grammaire pour ce langage, l'alphabet terminal étant :

`a   b   c   x   y   z   f   g   h   ,   (   )   ;   =   begin   end`

On prendra dans l'alphabet auxilliare les lettres P (l'axiome pour programme), SI (suite d'instructions, IA (instruction d'affectation), E (expression), V (variable).

Problème 2 : Automates avec sortie et Lex *(8 pts)*

Dans cette question l'alphabet est $A = \{a, b, c\}$ on conside deux transformations ϕ et ψ sur les mots de A^* pour lesquelles on cherchera des automates avec sortie et des programmes en Lex permettant de les exprimer. Les deux transformations ne modifient pas la longueur des mots.

Transformation ϕ. Cette transformation ne modifie pas les lettres b et c, par contre elle transforme a en

- b s'il est précédé de c

- c s'il est précédé de b

et ne modifie pas a s'il est en début de mot ou s'il est précédé de a.

Ainsi par exemple

$$\phi(abbcaabcabac) = abbcbabcbbcc$$

Question 1.

Donner un automate avec sortie qui réalise ϕ.

Question 2.

Peut-on trouver f si on connaît $\phi(f)$?

Question 3. Donner le texte Lex qui réalise ϕ.

Transformation ψ Cette transformation considère le mot de gauche à droite. Pour toute lettre x on examine la lettre y qui la suit. On ne modifie pas x si $x = y$; par contre si $x \neq y$ on remplace x par la lettre z qui est égale ni à x ni à y. La lettre à la fin du mot n'est pas modifiée.

Par exemple

$$\phi(abbcaabcabac) = cbabacabccbc$$

Question 4. Donner le texte Lex qui réalise ψ.

Question 5. Donner un automate avec sortie qui réalise ψ on ajoutera pour cela la lettre d qui servira de symbole de fin de mot.

Problème 3: Analyse LR et LL *(8pts)*

On donne la grammaire $\mathcal{G} = (A, \Sigma = \{S, T, U\}, \mathcal{R}, S)$ où les notations sont celles du cours, S est l'axiome, l'alphabet terminal est $A = \{a, b, c, d\}$.

Les règles sont :

$$S \to SaT, \quad S \to Tb, \quad T \to Uc$$
$$U \to dU, \quad U \to d.$$

Question 1. Calculer les 0-items pour cette grammaire.

Question 2. Montrer qu'elle n'est pas $LR(0)$ en utilisant ces items.

Question 3. Expliquer pourquoi elle n'est pas $LR(0)$ sur un exemple d'analyse ascendante.

Question 4. Montrer que cette grammaire est SLR(1) en construisant un automate qui reconnaît les $R_0(i)Suiv(X)$.

Question 5. Montrer que cette grammaire n'est pas $LL(1)$ à l'aide d'un exemple.