Maîtrise d'Informatique

Analyse Syntaxique et Compilation (ASC)

Mardi 25 janvier 2000

Durée 3 heures, documents manuscrits autorisés

Session de janvier 2000

Problème 1 : 5 points

On considère le langage L des expressions arithmétiques en notation infixe sans parenthèses et dont les variables ont pour nom a ou b ou c. On utilise l'alphabet terminal $A = \{a, b, c, +, *\}$

$$E \to T \quad E \to E + T \quad T \to F$$

$$T \to T * F, \quad F \to a \quad F \to b \quad F \to c$$

Question 1. Donner un automate fini qui reconnaît L.

Question 2. Donner une expression rationnelle de L.

On souhaite effectuer la traduction de ces expressions en expressions postfixes en utilisant un automate avec sortie. On utilise un symbole de fin de mot bien utile pour cette traduction; par exemple pour l'entrée :

$$a\ *\ b\ *\ c\ +\ b\ *\ c\ +\ a\ +\ a\ *\ b\ \$$$

la sortie de l'automate doit être :

$$a\ b\ *\ c\ *\ b\ c\ *\ +\ a\ +\ a\ b\ *\ +\ \$$$

Question 3. Donner un automate avec sortie qui réalise cette traduction dans le cas où il n'y a que des additions (le mot ne contient pas de lettre y).

Question 4. Modifier l'automate précédent pour traiter le problème dans le cas général (additions et multiplications mais pas de parenthèses).

Problème 2: 7 points

On considère la grammaire $\mathcal{G}$ sur l'alphabet terminal $A = \{a, b, c\}$ et non terminal $\Sigma = \{S, T, U, V, W\}$ qui engendre un langage fini et qui a pour règles :

$$S \to aT \quad S \to Ub \quad S \to Vc \quad T \to Uc$$

$$T \to Vb, \quad U \to W \quad V \to W \quad W \to \varepsilon$$

Question 1. Montrer que cette grammaire est $LL(1)$.

Question 2. En utilisant toutes les dérivations possibles de cette grammaire, calculer les langages $R_0(i)$ pour les 8 valeurs de i correspondant aux règles. En déduire que la grammaire n'est ni $LR(0)$ ni $SLR(0)$ (on peut se passer des 0-items).

Question 3. Donner un autre preuve de ce que $\mathcal{G}$ n'est pas $LR(0)$ en utilisant la définition de ces grammaires.

Question 4. Donner une grammaire qui engendre le même langage que $\mathcal{G}$ et qui soit $LL(1)$ et $LR(0)$.

Question 5. On considr̀e la grammaire suivante :

$$S \to Ta \quad S \to Ub \quad T \to Tcc$$

$$T \to c \quad U \to Ucc \quad U \to cc$$

Quel est le langage engendré? Montrer qu'elle n'est pas $LR(1)$. Donner une grammaire $LR(1)$ qui engendre le même langage.

Problème 3: 8 points

On étudie ici des grammaires qui engendrent des systèmes de parenthèses. On suppose d'abord que l'alphabet est $\{a, b\}$ et pour toute lettre x, on note $|f|_x$ le nombre de x qui apparaissent dans le mot f, le mot f est un système de paranthèses si $|f|_a = |f|_b$ et si pour tout g tel que $F = gh$ on a $|g|_a \geq |g|_b$. On appelle hauteur d'un mot g la quantité $|g|_a - |g|_b$.

$$S \to aSbS \quad S \to abS \quad S \to aSb \quad S \to ab$$

On a écrit le texte Bison suivant pour traduire chaque mot f engendré par cette grammaire en une suite d'entiers qui représentent les hauteurs de tous les facteurs gauches de f.

```
%{
#define NMAX 1000
#define YYSTYPE int
#include <stdio.h>

  int i= 0;
  int  cont[NMAX];

  void affiche(){
       int j;
       for (j = 0; j < i ; j++) printf("  %d  ", cont[j]);
       printf(" \n");}

  void decale (int k, int l){
       int j;
       for (j = k +1 ; j < l ; j++) cont[j]++;}

  void init (int k, int l){
        cont[k] = 1;
        cont[l] = 0;}
%}
```

```
  %token LA
  %token FIN
  %token LB

%%
  line : exp FIN   {affiche();}
  ;
  exp  : LA exp LB exp  {decale($1, $3); init($1, $3);}
       | LA exp LB   {decale($1, $3); init($1, $3);}
       | LA LB  exp   {init($1, $2);}
       | LA LB  {init($1, $2);}
  ;

%%

  yylex(){
      char  c;
      while ((c = getchar()) == ' ' || c == '\t' );
      ungetc(c,stdin);
      scanf("%c", &c);
      if    (c  == 'a')  {yylval = i; ++i; return LA;}
        else if (c == 'b') {yylval = i; ++i; return LB;}
        else if (c == ';')   return FIN;
        else yyerror("mauvais caractere");
     if (c == EOF) return 0;
  }

  main() {
    yyparse();}

 yyerror(char *s){
   printf ("%s\n",s);}
```

Question 1. Quelles sont les affichages obtenus pour chacun des mots suivants :

1. *ab*;
2. *aaabbbab*;
3. *aacbbbab*;
4. *aaabbbba*;

Question 2. On considère le langage L_1 des systèmes de parenthèses imbriquées sur l'alphabet $A = \{a, b, c, d\}$ où chaque a est fermé par un b et chaque c est fermé par un d. Ainsi a et b sont assimilés à des parenthèses ouvrantes et fermantes et c et d à des crochets ouvrants et fermants. Le mot *acdabbcd* appartient à L_1 mais *acbd* n'y appartient pas. Le mot vide n'appartient pas à L_1.

On demande de donner une grammaire de L_1.

Question 3. On généralise la définition précédente en admettant qu'entre un crochet ouvrant et le crochet qui le ferme il puisse y avoir des parenthèses ouvrantes qui ne sont pas fermées (mais pas de parenthèses fermantes non ouvertes). A l'extérieur des crochets toutes les parenthèses doivent être appariées. Cela défini un langage L_2 sur le même alphabet. Ainsi les mots suivants sont dans L_2 : $acadb$, $abcaaabcaabbadd$, par contre les mots suivants n'appartiennent pas à L_2 : $acbdabb$, $acd, acdbacd$.

Donner une grammaire pour L_2 en utilisant la grammaire obtenue à la question précédente et en ajoutant la possibilité d'introduire un nombre quelconque de a dans certains mots engendrés par cette grammaire.

A tout mot f de $\{a,b,c,d\}^*$ on associe un couple d'entiers $H_2(f) = (x,y)$ où $x = |f|_a - |f|_b$ et $y = |f|_c - |f|_d$.

Question 4. Donner toutes les modifications qu'il faut effectuer sur le programme Bison donné plus haut pour qu'il admette en entrée un mot de L_1 et qu'il affiche les valeurs de $H_2(f)$ pour tous les facteurs gauches de f. Pour le mot $f = aabccddb$ l'affichage doit être :

$$(1,0),(2,0)(1,0),(1,1)(1,2),(1,1),(1,0),(0,0)$$