UNIVERSITE DE BOURGOGNE
U.F.R. Sciences et Techniques
Filière : Licence 3 Informatique

Année 2022-2023
15 mai 2023

Session : 1

EPREUVE : Langages Formels et Compilation

Durée : 2 h 00 – (documents papiers - sauf livres - autorisés ; appareils électroniques interdits)

Les exercices sont indépendants. Le barème est donné à titre indicatif.

Exercice 1 – 4 points

Soit la grammaire G_1=({S}, {a, b}, S, {S→Sab|ab})

1. Donnez les listes d'analyse du mot abababab par l'algorithme d'Earley.
2. Selon les listes obtenues, quels sous-mots de abababab appartiennent au langage engendré par G_1 ? Justifiez votre réponse.
3. Que constatez-vous concernant les listes obtenues ? Que pouvez-vous en déduire sur les mots du langage ?

Exercice 2 – 6,5 points

Soit la grammaire G_2 = ({P, C, A, E}, {affectation, condition, if, then, else}, P, R),
avec R = { P → A P | C P | A | C , A → affectation, C → if condition then P E , E → else P | λ}).

1. Donnez l'arbre de dérivation et la dérivation la plus à gauche de la phrase
   « if condition then if condition then affectation else affectation ».
2. Montrez que G_2 est ambiguë.
3. Transformez G_2 en une grammaire équivalente sous forme normale de Greibach. Soit G_2' la grammaire ainsi obtenue.
4. A partir de G_2', construisez un automate à pile à un seul état reconnaissant les mots de $L(G_2)$ par pile vide.
5. Montrez que P → P A | P C | A | C est équivalent à P → A P | C P | A | C .

Exercice 3 – 3 points

Soit l'automate à mémoire linéairement bornée suivant :
$AMLB_3$ = ({q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_9, q_{10}, q_{11}, q_{12}}, {a, b, c, <, >}, {a, b, c, <, >, X, Y, Z}, t, q_0, {q_{12}})
Avec t définie par

(1) $t(q_0, <) = (q_1, <, D)$
(2) $t(q_1, a) = (q_2, X, D)$
(3) $t(q_2, a) = (q_2, a, D)$
(4) $t(q_2, Y) = (q_2, Y, D)$
(5) $t(q_2, b) = (q_3, Y, D)$
(6) $t(q_3, b) = (q_3, b, D)$
(7) $t(q_3, Z) = (q_3, Z, D)$
(8) $t(q_3, c) = (q_4, Z, G)$
(9) $t(q_4, Z) = (q_4, Z, G)$
(10) $t(q_4, b) = (q_4, b, G)$
(11) $t(q_4, Y) = (q_4, Y, G)$
(12) $t(q_4, a) = (q_5, a, G)$
(13) $t(q_4, X) = (q_6, X, D)$
(14) $t(q_5, a) = (q_5, a, G)$
(15) $t(q_5, X) = (q_1, X, D)$
(16) $t(q_6, Y) = (q_6, Y, D)$
(17) $t(q_6, b) = (q_7, Y, D)$
(18) $t(q_7, b) = (q_7, b, D)$
(19) $t(q_7, Z) = (q_7, Z, D)$
(20) $t(q_7, c) = (q_8, Z, G)$
(21) $t(q_8, Z) = (q_8, Z, G)$
(22) $t(q_8, b) = (q_9, b, G)$
(23) $t(q_8, Y) = (q_{10}, Y, D)$
(24) $t(q_9, b) = (q_9, b, G)$
(25) $t(q_9, Y) = (q_6, Y, D)$
(26) $t(q_{10}, Z) = (q_{10}, Z, D)$
(27) $t(q_{10}, c) = (q_{11}, Z, D)$
(28) $t(q_{11}, c) = (q_{11}, Z, D)$
(29) $t(q_{11}, >) = (q_{12}, >, G)$

1. Effectuez l'analyse du mot abbccc avec cet automate.
2. Quel est le langage reconnu par cet automate ? (Comment sont constitués les mots du langage ?)

Exercice 4 – 6,5 points (à rendre sur une feuille à part)

On souhaite utiliser Lex et Yacc pour analyser un ensemble de nombres et une opération associée. Un ensemble est écrit sous la forme {3840,526,9379} ou {123}. Les nombres (des entiers positifs) peuvent être quelconques. Les accolades '{' et '}' sont utilisées pour ouvrir et fermer l'ensemble, et doivent être reconnues. Il n'y a pas d'espace dans l'ensemble, le séparateur de nombres est la virgule ','. Un seul ensemble doit être reconnu (une seule ligne dans le fichier de données).

1. Dans un premier temps, le fichier de données comporte uniquement un ensemble de nombres. Lex doit reconnaître les nombres de l'ensemble et les transmettre à yacc (sous forme de nombres) afin que celui-ci les additionne et affiche le résultat.
   Vous devez écrire les sources Lex et Yacc en tenant compte des contraintes ci-dessous. Vous pouvez numéroter vos lignes de code pour effectuer des insertions nécessaires pour la suite du sujet.
   - Pour la partie Lex, donnez les définitions régulières utiles pour les éléments de syntaxe, les règles Lex, l'envoi des informations nécessaires au programme Yacc. Le programme n'effectue pas de calcul, la somme des nombres se fera dans les règles de Yacc ;
   - Pour la partie Yacc, donnez les token, le symbole de départ, les règles de grammaire et les expressions du calcul du nombre total. La somme des nombres sera écrite par Yacc, une fois l'ensemble fermé.

2. On souhaite étendre le langage et spécifier le type d'opération à effectuer sur l'ensemble sous la forme SUM{23,569,1}. Si on prévoit pour l'instant une seule opération « SUM » qui effectue la somme des éléments comme précédemment, que faut-il changer dans les 2 sources Lex et Yacc pour pouvoir afficher la valeur de la somme et la chaîne correspondant à l'opération ?

3. Pour aller plus loin, que faut-il mettre en place pour prendre en compte plusieurs types d'opérations ? Par exemple, toujours pour une seule ligne :
   SUM{1341,23432}
   ou
   MULT{1345134,15647}
   Les sources ne sont pas demandées ici, seulement une réflexion justifiée. On ne veut évidemment pas polluer les règles Yacc avec des conditions en C.