Licence d'informatique – Université Paris 7

Grammaires et Analyse

Examen du 30 janvier 2003, durée : 2 heures 30 minutes

Il va sans dire que la rigueur des raisonnements, la clarté des explications, mais aussi la qualité de la présentation influeront sensiblement sur la note.

Notes de cours manuscrites autorisées. Livres non autorisés.

Le barème n'est donné qu'à titre indicatif et pourra donc êtr= e modifié.

Exercice 1 (4 points)

Repondre `vrai` ou `faux` à chacune des assertions suivantes en donnant une justification (**de deux lignes max !**) pour chaque repon= se.

- $+0,5$ point pour chaque reponse **et** justification correcte ;
- $-0,5$ point pour chaque reponse **ou** pour chaque justification = non correcte ;
- 0 point pour chaque reponse **ou** pour chaque justification non= donnée.

a) Pour toute grammaire algébrique G il existe une grammaire G' équivalente à G et sans symboles inutiles.

b) Pour toute grammaire algébrique G il existe une grammaire G' propre et équivalente à G.

c) Pour toute grammaire algébrique G il existe une grammaire G' sous la forme normale de Chomski et équivalente à G.

d) Pour toute grammaire algébrique G il existe une grammaire G' sous la forme normale de Greibach et équivalente à G.

e) Pour toute grammaire algébrique G il existe une grammaire G' non ambiguë et équivalente à G.

f) Pour toute grammaire algébrique G il existe une grammaire G' sans récursion gauche et équivalente à G.

g) Pour tout langage algébrique L il existe un automate à pile déterministe qui reconnaît L.

h) Pour tout langage algébrique L il existe un automate à pile avec un seul état qui reconnaît L.

Exercice 2 (10 points + 4 points optionnels)

Des expressions arithmétiques sont composées des entités suivantes :

- des nombres réels ;
- des nombres complexes ;
- des parenthèses ouvrantes "`(`" ;
- des parenthèses fermantes "`)`";
- des opérateurs arithmétiques parmi les suivants :
  - "+" (opérateur d'addition) ;
  - "−" (opérateur de soustraction) ;
  - "∗" (opérateur de multiplication) ;

  - "/" (opérateur de division) ;
  - "**" (opérateur de puissance) ;
  - "//" (opérateur de logarithme : $a//b$ représente le lo= garithme de a en base b) ;

- des délimiteurs tels que les trois caractères `/b` (blanc), = `/t` (tabulation) et `/n` (nouvelle ligne).

On veut construire par la méthode vue en cours un analyseur lexical qui reconnaît les cinq unités lexicales suivant= es : ***nbreel***, ***nbcompl***, ***ouv***, ***ferm***, = ***op*** correspondant aux entités de la liste ci-dessus.

a) Donner les définitions rationnelles qui spécifient les cinq unités lexicales. Pour déterminer correctement ces définitions rationnelles, on lira d'abord les informations sur les modèles données à la fin de l'exercice.

b) Construire les diagrammes de transition correspondant à chaque unité lexicale.

- On essaiera d'utiliser un nombre total d'états qui soit le plus pet= it possible.

- On expliquera le choix de l'ordre dans lequel les diagrammes seront donnés (qui correspond à l'ordre dans lequel l'an= alyseur les utilisera pour reconnître un lexème).

- En correspondance des états terminaux de chaque diagramme, on indiquera les actions que l'analyseur devra effectuer quand il arrive sur l'état et on=

  donnera une description de ces actions.

- Au sujet de ces actions, on n'oubliera pas de prévoir le retour de = certains attributs pour les unités lexicales où cela s'avererait nécessaire.

- On n'oubliera pas d'indiquer si, en correspondance de certains des=

  états terminaux, il faut reculer d'une case le curseur qui parcourt le tampon où se trouve le flot d'entrée.

c) Après avoir écrit les diagrammes de transition complétés des a= ctions associées aux états terminaux, on passe à l'implémentation d'un programme (par exemple en C) qui simule le comportement de cet "automate enrichi". On vous demande d'écrire juste la section du programme qui simule le ou=

les diagramme(s) relatif(s) aux unités lexicales ***nbreel*** et ***nbcompl*** .

d) Écrire maintenant la grammaire qui engendre les expressions arithmétiques décrites dans cet exercice, c'est-à-dire la grammaire qui serait utilisée pour l'analyse syntaxique du flot d'unités lexicales produit par notre analyseur lexical.

e) (Optionnelle) Comment faut-il changer les définition rationnelles si l'on suppose que les nombres réels et les parties réelle et imaginaire des nombres complexes peuvent être précédés par un signe + ou − ?

Quels problèmes se présentent à l'analyseur lexical si l'on fait cette hypothèse et comment faut-il modifier les diagrammes de transition ?

Informations sur les modèles :

- Un nombre réel (sans signe) est constitué d'une partie entière, suivie d'une partie décimale qui commence par le point décimal suivi par les décimales. Chacune des deux parties entière et décimale peut éventuellement manq= uer mais les deux parties ne peuvent pas manquer en même temps. Il n'y a pas de partie exponentielle. Il n'y a pas de zé= ros inutiles en tête.

  Exemples : "21.0377", "471", ou ".125".

- Un nombre complexe est constitué d'une partie réelle (un nombre r= éel sans signe) suivi de la partie imaginaire. La partie imaginaire est constituée d'un s= igne + ou d'un signe − suivi de la lettre i et du coefficient imaginaire (un autre réel sans signe). Il n'y a pas de délimiteur entre la partie réelle et la partie imaginai= re, ni entre le signe de celle-ci et le i, ni entre le i et du coefficient imaginaire. La partie réelle peut éventuellement manquer, dans ce cas le signe devant le i manque aussi (ceci est convenu pour éviter de pouvoir retrouver deux signes qui se suivent) et on dit qu'il s'agit d'un nombre *purement imaginaire* (il n'y a pas d'unité lexicale spécifique pour ces nombres).

  Exemples : "$45.8 + i12.46$", "$471 - i.5$", ou "$i.125$"

- Les symboles utilisés pour l'addition et la soustraction sont les mêmes que les symboles initiaux de la partie imaginaire d'un nombre=

  complexe.

- Deux lexèmes distincts sont toujours séparés par au moins un délimiteur.

Exercice 3 (6 points)

Soit la grammaire G :

$$S \quad \rightarrow \quad aSb \mid bSa \mid SS \mid ab \mid ba$$

a) Quel est le langage engendré par G ? Donner une justification intuitive de votre reponse qui ne fasse pas plus de 10 lignes de texte.

b) En appliquant les méthodes presentées en cours, trouver une gramma= ire équivalente à G sous la forme normale de Chomski et ensuite trouver une grammaire équivalente à G sous la forme normale de Greibach.

Note. *Au moment du renommage des variables que l'on fait au début de la mise sous forme normale de Greibach, on fera particulièrement attention à attribuer les noms $A_1, A_2, \ldots$, etc., aux différentes variables de façon telle que, pour le plus grand nombre possible de règles, l'index i de la variable A_i qui constitue le membre gauche de la règle soit inférieur à l'index j de la variable A_j par laquelle commence le membre droit. On ne perdra pas=*

de vue que l'axiome reste toujours la variable A_i que l'on fait correspondre à la variable S.

c) En appliquant la méthode presentée en cours, écrire un automate à pile qui reconnaît le langage engendré par G.

d) Simuler le comportement de cet automate (y compris l'évolution de la pile) pendant le calcul de reconnaîssa= nce du mot $w = baaabb$.