

Compilation

Maîtrise informatique — Université Paris 7

Examen du 19.1.1999

durée 3h

*Notes de cours autorisées. Autres documents interdits.
Les réponses non justifiées seront considérées comme fausses.*

Exercice 1 Écrire un source `lex` pour effectuer l'*ensemble* des actions suivantes sur un texte :

- supprimer les blancs (espaces et tabulations) en fin de ligne,
- remplacer toutes les tabulations par un nombre fixé d'espaces, 8 par exemple,
- mettre un blanc unique après les signes de ponctuation [.,;:!?]
- rétablir les majuscules oubliées pour les lettres qui suivent un des signes de ponctuation [.,;!:].

Exercice 2 Soit G la grammaire suivante, où l'alphabet des terminaux est $\{+, \text{id}, \text{nb}, =, (,), _ , \Pi, : \}$:

$$E \longrightarrow EE+ \quad (1)$$

$$E \longrightarrow \text{nb} \quad (2)$$

$$E \longrightarrow \text{id} \quad (3)$$

$$E \longrightarrow \Pi(\text{id} = \text{nb} _ \text{nb} : E) \quad (4)$$

Si besoin, on pourra ajouter une règle $S \longrightarrow E\$$, S devenant l'axiome. Dans ce cas, ajouter aussi un $\$$ à la fin des chaînes des questions 2) et 6).

Les trois premières règles de G permettent d'engendrer les expressions arithmétiques postfixées (avec l'opérateur $+$). La dernière règle sert à représenter des produits de plusieurs termes. Ainsi par exemple, $\Pi(i = 2 _ 4, x \ i+)$ désigne le produit, pour i allant de 2 à 4 des $(x + i)$, c'est-à-dire $(x + 2)(x + 3)(x + 4)$.

- La grammaire G est-elle ambiguë? Justifier.
- Donner un arbre de dérivation de $\Pi(\text{id}=\text{nb} _ \text{nb}:\text{id} \text{id} +)$.
- La grammaire G est-elle LL(1)? Justifier.
- Construire l'automate LR(0) de G .
- La grammaire G est-elle LR(0)? SLR(1)? Justifier.
- Simuler le comportement d'un analyseur syntaxique ascendant sur l'entrée $\Pi(\text{id}=\text{nb} _ \text{nb}:\text{id} \text{id} +)$.

Exercice 3 Soit G la grammaire suivante, engendrant des listes :

$$S \longrightarrow (L) \quad (1)$$

$$S \longrightarrow \mathbf{a} \quad (2)$$

$$L \longrightarrow L, S \quad (3)$$

$$L \longrightarrow S \quad (4)$$

L'ensemble des terminaux est $\mathcal{T} = \{ , , \mathbf{a},) , (\}$. L'élément lexical \mathbf{a} est appelé un atome.

Étant donnée une liste l engendrée par G , on cherche à calculer le nombre d'atomes, de composants et la profondeur d'emboîtement de l . Par exemple, pour la liste $(\mathbf{a}, (\mathbf{a}), ((\mathbf{a}), \mathbf{a}), \mathbf{a})$, le nombre d'atomes est 5, le nombre de composants est 4 (le nombre de composants d'une liste (l_1, \dots, l_k) est k , le nombre de composants d'un atome est 0) et la profondeur d'emboîtement de la liste est 3.

- Pourquoi G n'est-elle pas LL(1)? Justifier la réponse.

- 2) Trouver une grammaire G' qui engendre le même langage que G et qui est LL(1).
- 3) Calculer les ensembles Premier et Suivant de chaque non-terminal de G' , et calculer sa table d'analyse LL(1).
- 4) Définir des attributs et associer à chaque règle de G des actions sémantiques pour calculer :
 - le nombre d'atomes d'une liste ;
 - le nombre de composants d'une liste ;
 - la profondeur d'emboîtement d'une liste ;

Pour chaque attribut, dire s'il est synthétisé ou hérité.

- 5) Donner l'arbre de dérivation du mot $(\mathbf{a}, ((\mathbf{a}), \mathbf{a}))$, décoré avec les valeurs de ces attributs.
- 6) Proposer un schéma de traduction pour la grammaire G' pour calculer uniquement la profondeur d'une liste. Pour chacun des attributs définis pour G' , dire lesquels sont hérités et lesquels sont synthétisés. La grammaire obtenue est-elle L -attribuée ?
- 7) Donner l'arbre de dérivation décoré du mot $(\mathbf{a}, ((\mathbf{a}), \mathbf{a}))$ avec les attributs de G' .

Exercice 4 Soit $G = (\mathcal{V}, \mathcal{T}, S, P)$ une grammaire LL(1) sans ε -production. \mathcal{V} est l'ensemble des non-terminaux, \mathcal{T} l'ensemble des terminaux, S l'axiome et P les productions de G . On suppose que tous les non-terminaux sont utiles. On considère son automate déterministe LR(0), noté \mathcal{A} . Un *item* sera noté $X \rightarrow \alpha \bullet \beta$, où $X \rightarrow \alpha\beta$ est une règle de G .

On dit qu'un état q de l'automate \mathcal{A} vérifie

- la propriété (\mathcal{P}) si q ne contient pas deux items distincts de la forme $X \rightarrow \alpha \bullet x\beta$ et $Y \rightarrow \gamma \bullet x\delta$ où X est un non-terminal, x est soit un terminal soit un non terminal, et $\alpha, \beta, \gamma, \delta$ sont dans $(\mathcal{V} \cup \mathcal{T})^*$.
- la propriété (\mathcal{Q}) si q contient au plus un item pour lequel le marqueur \bullet n'est pas en début de règle.

- 1) Montrer qu'il n'y a pas de conflit dans l'état initial de \mathcal{A} .
- 2) Montrer que l'état initial vérifie la propriété (\mathcal{Q}).
- 3) Montrer que l'état initial vérifie la propriété (\mathcal{P}). On pourra montrer que si l'état initial contient deux items distincts de la forme $X \rightarrow \alpha \bullet x\beta$ et $Y \rightarrow \gamma \bullet x\delta$, alors G ne peut pas être LL(1). On pourra distinguer deux cas, suivant que $X = Y$ ou $X \neq Y$.
- 4) Soit q un état de l'automate. On suppose que q vérifie (\mathcal{P}) et (\mathcal{Q}). Soit x un symbole (terminal ou non) et q' l'état obtenu à partir de q par la transition étiquetée x . Montrer que q' vérifie aussi les propriétés (\mathcal{P}) et (\mathcal{Q}). Pour (\mathcal{P}), on pourra encore raisonner par l'absurde et supposer que q' contient deux items distincts de la forme $X \rightarrow \alpha \bullet x\beta$ et $Y \rightarrow \gamma \bullet x\delta$. On montrera qu'alors $\alpha = \varepsilon$ ou $\gamma = \varepsilon$, et on étudiera les cas $\alpha = \gamma = \varepsilon$, $\alpha \neq \gamma = \varepsilon$.
- 5) En déduire que tout état de \mathcal{A} a les propriétés (\mathcal{P}) et (\mathcal{Q}), puis que G est LR(0).
- 6) Donner un exemple de grammaire LR(0) sans ε -production qui n'est pas LL(1).