Théorie des Langages

Magistère STIC

Examen du 31 mars 2005
durée 2 heures

Les documents sont autorisés.

Les exercices sont indépendants.

Toutes les réponses devront être correctement justifiées. La rigueur des raisonnements, la clarté des explications, et la qualité de la présentation influeront sensiblement sur la note.

1 Analyse LR

L'objectif est de construire l'analyseur LR(1) pour la grammaire G définie par

$$\begin{array}{rlcl} 1: & S' & \longrightarrow & S \\ 2: & S & \longrightarrow & aSS \\ 3: & S & \longrightarrow & b \end{array}$$

qui engendre le langage de Lukasiewicz.

a) Calculer l'automate $\mathcal{A}$ des contextes de réduction valides de cette grammaire. On rappelle que les états de $\mathcal{A}$ sont les contextes de réduction valides $C(\gamma)$ pour $\gamma \in \{a,b,S\}^*$ et les transitions sont de la forme $C(\gamma) \xrightarrow{x} C(\gamma x)$ pour $x \in \{a,b,S\}$.

b) Donner la table des actions $\mathrm{Act}(C,x)$ pour chaque $x \in \{\varepsilon,a,b\}$ et C contexte de réduction valide.

c) Donner le calcul de l'analyseur LR(1) de la grammaire G ainsi obtenu sur le mot *abaabbb*. On indiquera bien pour chaque étape de calcul le contenu de la pile qui alterne contextes de réduction valides et lettres.

2 Analyse LL

On rappelle qu'une grammaire est LL s'il existe un entier k tel qu'elle soit LL(k).

a) Montrer que si une grammaire G est ambiguë alors elle ne peut pas être LL.

b) Montrer que tout langage rationnel peut être engendré par une grammaire LL(1).

On considère la grammaire G définie par

$$\begin{array}{rlll} 1: & S & \longrightarrow & a \\ 2: & S & \longrightarrow & \texttt{if}\ c\ \texttt{then}\ S\ \texttt{endif} \\ 3: & S & \longrightarrow & \texttt{if}\ c\ \texttt{then}\ S\ \texttt{else}\ S\ \texttt{endif} \end{array}$$

dans laquelle l'alphabet terminal comporte les 6 symboles a, c, `if`, `then`, `else` et `endif`.

c) (*) Montrer que la grammaire G est non ambiguë.

d) Montrer que la grammaire G n'est pas LL.

e) Montrer que la grammaire G' définie par

$$\begin{array}{rlll} 1: & S & \longrightarrow & a \\ 2: & S & \longrightarrow & \texttt{if}\ c\ \texttt{then}\ S\ S' \\ 3: & S' & \longrightarrow & \texttt{endif} \\ 4: & S' & \longrightarrow & \texttt{else}\ S\ \texttt{endif} \end{array}$$

engendre le même langage que la grammaire G.

f) Montrer que la grammaire G' est fortement LL(1) et donner la table d'analyse LL(1) de G'.

3 Facteurs

Soit Σ un alphabet et $L \subseteq \Sigma^*$ un langage. On note $\mathrm{Pref}(L)$ (resp. $\mathrm{Suff}(L)$, $\mathrm{Fact}(L)$) l'ensemble des préfixes (resp. suffixes, facteurs) des mots de L.

On rappelle que le langage de Dyck D_1^* est engendré par la grammaire G définie par $S \longrightarrow aSbS + \varepsilon$.

a) Donner une grammaire qui engendre le langage $\mathrm{Pref}(D_1^*)$.
On prouvera bien que cette grammaire engendre ce langage.

b) Même question pour le langage $\mathrm{Suff}(D_1^*)$.

c) Même question pour le langage $\mathrm{Fact}(D_1^*)$.

d) Montrer que l'application Fact est une transduction rationnelle. Que peut-on en déduire sur $\mathrm{Fact}(L)$ lorsque L est un langage linéaire? algébrique? rationnel?