Langages et Traducteurs

Quick du jeudi 12 novembre 2015

Durée : une heure - Sans documents.

On s'intéresse à la syntaxe d'appels de fonction inspirée du langage CAML. Cette syntaxe peut être décrite par un langage hors-contexte défini sur le vocabulaire $V_t = \{i, (,), \#\}$ dans lequel i dénote un identificateur (un nom de fonction ou de paramètre) et $\#$ est une marque de fin de séquence.

Partie 1

On se restreint tout d'abord au cas où chaque fonction ne comporte que zéro ou un paramètre, et l'on note $\mathcal{L}_0$ le langage correspondant. Les séquences suivantes sont des éléments de $\mathcal{L}_0$:

```
i#           -- appel de fonction sans paramètre
(i i)#       -- appel de fonction avec un paramètre i
(i (i i))#   -- appel de fonction avec un paramètre (i i)
(i (i (i i)))# -- appel de fonction avec un paramètre (i (i i))
etc.
```

Le langage $\mathcal{L}_0$ peut être reconnu par la grammaire G_0 suivante :

$$\begin{aligned} Z &\longrightarrow F \# \\ F &\longrightarrow i \mid (i F) \end{aligned}$$

Question 1 (2 pt). Montrez que G_0 est LL(1).

Question 2 (2 pts).

En utilisant l'analyseur LL(1) déduit de la question 1 montrez que la phrase $(i (i i))\#$ appartient à $\mathcal{L}_0$.

Question 3 (4 pts). Montrez que G_0 est LR(0).

Partie 2

On définit la grammaire G_1 en ajoutant la règle suivante à la grammaire G_0 :

$$F \longrightarrow i F$$

Question 4 (4 pts).

Montrez que G_1 n'est pas LL(1). Transformez G_1 en une grammaire G'_1 qui soit LL(1)? Vous justifierez le fait que G'_1 est LL(1).

Question 5 (5 pts). Montrez que G_1 n'est pas LR(0) mais est SLR(1).

Indication : Une grammaire est SLR(1) ssi elle satisfait l'une des deux conditions suivantes

1. soit elle est LR(0);
2. soit, dans l'analyseur LR(0) obtenu :
 - pour tout conflit lecture/réduction de la forme $([X \rightarrow \alpha.a\beta], [Y \rightarrow \gamma.])$ alors $a \notin \text{Suivants}(Y)$
 - pour tout conflit réduction/réduction de la forme $([X \rightarrow \alpha.], [Y \rightarrow \beta.])$ alors $\text{Suivants}(X) \cap \text{Suivants}(Y) = \emptyset$

Partie 3

On généralise maintenant au cas où un appel de fonction peut comporter un nombre de paramètre quelconque, chaque paramètre étant lui-même un appel de fonction avec un nombre quelconque de paramètres. On note $\mathcal{L}$ ce nouveau langage, et on a $\mathcal{L}_0 \subset \mathcal{L}$. Les séquences suivantes sont des éléments de $\mathcal{L}$:

```
i#
(i i i)#
i (i (i i))#
(i (i i) i)#
etc.
```

Le langage $\mathcal{L}$ peut être reconnu par la grammaire G suivante :

$$\begin{aligned} Z &\longrightarrow F \# \\ F &\longrightarrow i \\ F &\longrightarrow iP \\ F &\longrightarrow (iP) \\ P &\longrightarrow F \\ P &\longrightarrow PF \end{aligned}$$

Question 6 (3 pts). Montrez que G est ambiguë.

Indication : une grammaire G est ambiguë si et seulement si il existe une phrase de $L(G)$ qui admet deux arbres de dérivations distincts.