Correction du QCM Compilation : Langages et Grammaires

EPITA – Promo 2006

27 Novembre 2003

À quelle(s) classe(s) appartiennent chacune des grammaires suivantes ? (NB : si une grammaire est de type A et $A \subset B$, cocher A et B).

a. Régulière

b. Hors Contexte

c. Non ambiguë

d. Engendre un langage rationnel

e. Engendre un langage non vide

1.
```
<args>  ::= <args1> | ε
<args1> ::= <args1> ',' 'arg'
         |  'arg'
```

Correction: Linéaire à gauche, donc régulière. Elle est non ambiguë. `<args1>` engendre une liste de une ou plusieurs `arg` *séparés* par des `,`. Donc, cette grammaire engendre une liste de zéro ou plusieurs `arg` *séparés* par des `,`. Ce langage est infini, régulier (puisqu'engendré par une grammaire régulière) : type 3. Il existe donc un automate, comme par exemple :

```
                      ,----.
                      |    |
                      `----'
                       |   ^
                 `arg' |   | `,'
                       v   |
        ,----.  `arg' ,----.
    --->|    |------->|    |--->
        `----'        `----'
          |
          v
```

2.
```
<args>  ::= <args1> | ε
<args1> ::= <args1> ',' <args1>
         |  'arg'
```

Correction: Cette grammaire est très visiblement une version ambiguë de la grammaire précédente. On pourrait dire que dans la grammaire précédente l'opérateur `,` est associatif à gauche, ici il est associatif (i.e., à droite et à gauche) : une phrase comme `arg , arg , arg` peut se lire comme `(arg , arg) , arg` ou `arg , (arg , arg)`. Le langage, lui, reste évidemment de type 3, et reconnu par le même automate.

3.
```
A → a B
B → A b | c
```

Correction: $\mathtt{a}^n\mathtt{cb}^n$... Ç'aurait été étonnant qu'il ne nous la colle pas celle-là vu comme il est créatif... Pas régulière, hors-contexte, pas ambiguë, pas reconnaissable (ce langage est l'archétype du non rationnel). Et oui : non $\mathtt{a}^n\mathtt{cb}^n$ n'est pas vide.

4. $$\begin{array}{lcl} \mathtt{X} & \rightarrow & \mathtt{x\ X\ Y\ Z} \\ \mathtt{X} & \rightarrow & \mathtt{X\ y\ Z} \\ \mathtt{Z\ Y} & \rightarrow & \mathtt{Y\ Z} \\ \mathtt{y\ Y} & \rightarrow & \mathtt{y\ y} \\ \mathtt{y\ Z} & \rightarrow & \mathtt{y\ z} \\ \mathtt{z\ Z} & \rightarrow & \mathtt{z\ z} \end{array}$$

Correction: Cette grammaire est visiblement monotone, non hors contexte. On voit rapidement qu'il n'y a aucun moyen de se "débarrasser" de X : cette grammaire ne produit aucun mot ! Elle ne peut donc être ambiguë, et le langage qu'elle engendre, $\emptyset$, est bien sûr reconnaissable par un automate fini.