INPG **ENSIMAG**
Année Spéciale **Année 2005-06**

Compilation

TD1 - Quelques aspects de syntaxe

Parlons « *compil* »

Qu'est ce que c'est ? Pourquoi l'étudier ? Quels sont ses buts ? Quelles sont les grandes phases du processus de compilation ? Qu'est ce que ce n'est pas ?

Lectures Vos prochains livres de chevet sont à choisir parmi [GBJ00].

Exercice 1 Donnez des exemples de programmes écrits en C ou en CAML comportant respectivement une erreur lexicale, syntaxique, de sémantique statique et de sémantique dynamique.

Expressions régulières

Exercice 2 Un *chiffre* est un caractère de l'ensemble $\{0,1,\ldots,9\}$. Une *lettre* est un caractère de l'ensemble $\{a,\ldots,z\}$. Un *signe* est un caractère de l'ensemble $\{-,+\}$.

Une *constante entière* est une séquence non vide de chiffres, sans 0 en tête, et pouvant débuter par un signe. Un *identificateur* est une séquence non vide de chiffres ou de lettres, débutant par une lettre.

Donnez une expression régulière caractérisant les ensembles *constante entière* et *identificateurs*.

Analyses lexicale et syntaxique

Exercice 3 Soit le code HTML suivant :

```
<html>
<head>
<title>Ecole Nationale Sup&eacute;rieure</title>
<style type="text/css">
<!--
a:hover {text-decoration: none}
-->
</style>
...
</body>
</html>
```

– Donnez les lexèmes (les *tokens*) qui sortiraient de votre analyseur lexical.
– La frontière que vous avez placé entre analyses lexicale et syntaxique est-elle univoque ? Quelles sont les conséquences de votre choix ?

Grammaires

Exercice 4 On considère la grammaire suivante qui décrit les instructions conditionnelles d'un langage de programmation :

$$\begin{array}{lcl} S & \longrightarrow & I \\ I & \longrightarrow & \text{if } e \text{ then } I \\ I & \longrightarrow & \text{if } e \text{ then } I \text{ else } I \\ I & \longrightarrow & a \end{array}$$

1. Montrez que cette grammaire est *ambiguë* : trouvez une phrase du langage qui admet deux arbres de dérivations distincts.
2. Modifiez la grammaire pour la rendre non *ambiguë* – bien entendu elle doit toujours décrire le même langage.

Exercice 5 Soit le langage régulier défini sur un vocabulaire V formé d'identificateurs $i \in V$.

1. Écrivez une grammaire hors-contexte qui décrit l'ensemble des expressions régulières bien formées de ce langage. Exemple : $(i+i)^*(i \cdot i)$.

2. Est-elle ambiguë ? Dans l'affirmative changez là.
3. Changez la grammaire de telle sorte que ses arbres de dérivations respectent les règles de priorité usuelles aux expressions régulières.
4. Donnez les arbres de dérivation (ou arbre syntaxiques) produits par votre grammaire pour les expressions :
   - $i + i \cdot i$
   - $i \cdot (i + i^*)^*$

Exercice 6 On considère le vocabulaire $V = \{a, b, c\}$. Proposez une grammaire régulière ou hors-contexte décrivant chacun des sept langages suivants :
- $L1 = a^*bc^*$
- $L2 = \{a^n bc^n \mid n \geq 0\}$
- $L3 = \{a^n bc^m \mid 0 < n < m\}$
- $L4 = \{w \mid w \text{ est un palindrome}\}$
- $L5 = \{a^n b^p \mid n \geq p \geq 0\}$
- $L6 = \{a^n b^p \mid n \neq p\}$
- $L7 = \{a^n b^p c^q \mid n + p = q\}$

Exercice 7 Donnez une grammaire non ambiguë pour le langage des parenthèses et crochets où chaque crochet fermant ferme aussi toutes les parenthèses ouvertes en suspens depuis le crochant ouvrant correspondant. Exemple de mot appartenant au langage : [((([(])]

Automates

Exercice 8 Donnez des automates déterministes reconnaissant les langages suivant :
- $a^*b + ab^*$
- $0(1^+0^*)^*1 + (01^+10)^*$
- langage de l'exercice **2**

Retour sur l'analyse On se rappelle que l'analyseur travaille sur un ensemble de lexèmes. Sous quelle forme les décrire et pourquoi ?

Vous pouvez maintenant jeter un oeil aux outils Lex et Yacc [LMB95].

Exercice 9 Implémentez votre premier analyseur lexical reconnaissant le langage de l'exercice **2** dans votre langage de programmation préféré.

Références

[GBJ00] D Grune, H.E. Bal, and Jacobs. *Modern Compiler Design*. John Wiley & Sons, 2000.

[LMB95] J. R. Levine, T. Mason, and D. Brown. *Lex & Yacc*. Unix Programming Tools. O' Reilly, 1995.