TD 9 : Grammaires attribuées

On étudie dans ce TD des mécanismes déclaratifs pour associer une sémantique à un arbre de dérivation d'une grammaire algébrique.

Définition 1 (Domaine sémantique)**.** Un *domaine sémantique* est un ensemble $T(\Gamma)$ de termes où Γ est un alphabet ordonné, construits selon une syntaxe abstraite

$$t ::= \gamma(t_1, \ldots, t_n)$$

où γ un opérateur d'arité n de Γ.

Le but des formalismes qui suivent est de construire les termes de $T(\Gamma)$; on parle souvent d'*arbres abstraits* ou d'*arbres de syntaxe abstraite* (AST) pour ces termes.

La sémantique effective $[\![t]\!]$ d'un terme de $T(\Gamma)$ sera alors définie inductivement sur t. On va par exemple considérer dans ce TD le domaine sémantique $T(\mathbb{N} \cup \{+, *\})$ avec la sémantique évidente

$$\begin{aligned} [\![+(t_1, t_2)]\!] &= [\![t_1]\!] + [\![t_2]\!] \\ [\![*(t_1, t_2)]\!] &= [\![t_1]\!] * [\![t_2]\!] \\ [\![n]\!] &= n \ . \end{aligned}$$

Exercice 1 (Grammaires synchrones)**.**

1. On considère la grammaire algébrique

$$\begin{aligned} L &\to LD \mid D \\ D &\to 0 \mid 1 \mid \cdots \mid 9 \end{aligned}$$

   pour représenter des entiers en base 10. Proposer une grammaire synchrone sur les mêmes règles qui construit un terme de $T(\mathbb{N} \cup \{+, *\})$ qui s'évalue en la valeur représentée.
2. Proposer une grammaire synchrone pour les expressions arithmétiques.
3. Comment construire les termes de sortie d'une grammaire synchrone lors d'une analyse syntaxique ascendante ?

Exercice 2 (Grammaires attribuées)**.**

1. On complète la grammaire de représentation des entiers en permettant de choisir une base de représentation binaire, octale ou décimale :

$$\begin{aligned} N &\to LB \\ L &\to LD \mid D \\ D &\to 0 \mid 1 \mid \cdots \mid 9 \\ B &\to b \mid o \mid d \end{aligned}$$

Par exemple « 00101b » représente l'entier 5. Proposer une grammaire attribuée qui effectue ce calcul.

2. On considère un fragment d'un langage de programmation :

$$\begin{aligned}\langle prog\rangle &\to \langle block\rangle\\ \langle block\rangle &\to \langle stmts\rangle\\ \langle stmts\rangle &\to \langle stmts\rangle \texttt{ ; } \langle stmt\rangle \mid \langle stmt\rangle\\ \langle stmt\rangle &\to \texttt{decl } id \mid \langle exec\rangle \mid \texttt{begin } \langle block\rangle \texttt{ end}\end{aligned}$$

On souhaite calculer un attribut v pour chaque non terminal $\langle block\rangle$, $\langle stmts\rangle$, $\langle stmt\rangle$ et $\langle exec\rangle$ la liste des identifiants déclarés dans sa portée. En particulier, dans un programme de la forme

$$\langle exec\rangle_1 \texttt{ begin } \langle exec\rangle_2 \texttt{ ; decl i end}$$

l'identifiant `i` est dans la portée de $\langle exec\rangle_2$ mais pas de $\langle exec\rangle_1$.

Proposer une grammaire attribuée pour ce calcul, en considérant le domaine sémantique $T(V, \{\cdot, \bot\})$ des listes d'identifiants où V est l'ensemble des listes singleton contenant un identifiant, $\cdot$ l'opération binaire de concaténation de listes, et $\bot$ le symbole d'arité zéro pour la liste vide.

Exercice 3 (Grammaires S-attribuées)**.** Une *grammaire S-attribuée* ne contient que des attributs synthétisés.

Montrer que toute grammaire synchrone peut être simulée par une grammaire S-attribuée.

Exercice 4 (Grammaires L-attribuées)**.** Une *grammaire L-attribuée* est telle que pour toute règle $A_0 \to u_0 A_1 u_1 \dots A_m u_m$, les attributs $\langle a, i\rangle$ ne peuvent dépendre que des attributs hérités $\langle h, 0\rangle$ de A_0 et des attributs synthétisés $\langle s, j\rangle$ des A_j pour $j < i$.

1. Comment peut-on implémenter une grammaire L-attribuée sur la base d'un analyseur descendant ?
2. Comment le faire sur la base d'un analyseur ascendant ?

Exercice 5 (Détection des cycles)**.** Proposer un algorithme pour détecter les cycles dans les dépendances d'une grammaire attribuée.