TD COMPILATION — FEUILLE 3

Grammaires attribuées.

Exercice 1. On définit les arbres binaires étiquetés par des entiers dans l'intervalle `V_MIN..V_MAX` par la grammaire suivante :

$$\begin{array}{lll} \text{arbre} & \rightarrow & () \\ & \rightarrow & (\ \textbf{num}\ \text{arbre arbre}\ ) \end{array}$$

▷ **Question 1** *Donner un système d'attributs qui permet d'évaluer la hauteur d'un arbre.*

▷ **Question 2** *Donner un système d'attributs qui permet de reconnaître si un arbre est un arbre de recherche.* **On donnera deux solutions : une solution avec uniquement des attributs synthétisés et une seconde solution utilisant aussi des attributs hérités**.

Rappel : un arbre de recherche est tel que, pour chaque noeud de l'arbre d'étiquette n, toutes les étiquettes du sous-arbre gauche sont inférieures ou égales à n et toutes les étiquettes du sous-arbre droit sont supérieures ou égales à n.

Exercice 2. Transformation de calcul d'attributs.

▷ **Question 1** *Soit une règle de la forme $A \rightarrow \omega\ \beta_1 \mid \omega\ \beta_2$ avec comme système d'attributs :*

$$\begin{array}{lll} A \uparrow f_1(v, v_1) & \rightarrow & \omega \uparrow v\ \beta_1 \uparrow v_1 \\ A \uparrow f_2(v, v_2) & \rightarrow & \omega \uparrow v\ \beta_2 \uparrow v_2 \end{array}$$

Proposer un système d'attributs équivalent sur les règles obtenues après factorisation :

$$\begin{array}{lll} A & \rightarrow & \omega\ A' \\ A' & \rightarrow & \beta_1 \mid \beta_2 \end{array}$$

▷ **Question 2** *Soit une règle de la forme $A \rightarrow A\ \alpha \mid \beta$, avec A n'apparaissant pas en première position dans β, et le système d'attributs*

$$\begin{array}{lll} A \uparrow f(v, w) & \rightarrow & A \uparrow v\ \alpha \uparrow w \\ A \uparrow g(v) & \rightarrow & \beta \uparrow v \end{array}$$

Proposer un système d'attributs équivalent sur les règles obtenues après élimination de la récursivité à gauche :

$$\begin{array}{lll} A & \rightarrow & \beta A' \\ A' & \rightarrow & \alpha A' \mid \epsilon \end{array}$$

▷ **Question 3** *On veut évaluer les expressions arithmétiques définies par la grammaire suivante :*

$$\begin{array}{lll} E & \rightarrow & E + T \\ E & \rightarrow & E - T \\ E & \rightarrow & T \\ T & \rightarrow & T * F \\ T & \rightarrow & F \\ F & \rightarrow & NUM \\ F & \rightarrow & (E) \end{array}$$

Donner un système d'attributs sur cette grammaire et sur une grammaire LL(1) de ce langage.

Exercice 3. Soit la grammaire suivante :

$$\begin{array}{lcl} \text{affect} & \rightarrow & \text{var} := \text{exp} \ ; \\ \text{exp} & \rightarrow & \text{var} \\ \text{var} & \rightarrow & \text{idf} \\ \text{var} & \rightarrow & * \ \text{var} \end{array}$$

On s'intéresse à la définition d'un interpréteur permettant de calculer la valeur d'une expression et l'effet de l'affectation sur la mémoire. Pour cela on dispose des définitions suivantes :

– Env : le type des fonctions partielles associant à des identificateurs des adresses mémoire.
– Mem : le type des fonctions totales associant à des adresses mémoire des valeurs.

Les valeurs sont des entiers qui peuvent représenter des adresses mémoire. On supposera que les expressions et les affectations sont bien typées. On utilisera la notation $dom(f)$ pour désigner le domaine d'une fonction, $f(x)$ pour désigner la valeur de la fonction f au point x et $f \leftarrow \{(x, v)\}$ pour désigner la fonction dont la nouvelle valeur au point x est v et qui est sinon égale à f.

▷ **Question 1** *Définir un système d'attributs permettant de calculer l'effet d'une instruction d'affectation sur la mémoire. Appliquer ce calcul à l'affectation* $*x := **y$;.