Université Paris 7 - DEUG-MIAS Année 2004-2005

TD de *Logique et Circuits* n° 6

Définitions inductives: les arbres

Exercice 1 L'ensemble $\mathcal{A}(\mathbb{N})$ des arbres binaires étiquetées sur les entiers est la clotûre inductive de l'ensemble des règles suivantes:

$$\frac{}{nil \in \mathcal{A}(\mathbb{N})} \qquad \frac{n \in \mathbb{N}, a_1 \in \mathcal{A}(\mathbb{N}) \text{ et } a_2 \in \mathcal{A}(\mathbb{N})}{cons(n, a, a') \in \mathcal{A}(\mathbb{N})}$$

Définir les fonctions $hauteur, taille : \mathcal{A}(\mathbb{N}) \to \mathbb{N}$.

Soit $\mathcal{R} \subseteq \mathcal{A}(\mathbb{N}) \times \mathcal{A}(\mathbb{N})$ le relation définie par $(a_1, a_2) \in \mathcal{R}$ si $hauteur(a_1) \leq hauteur(a_2)$ ou $hauteur(a_1) = hauteur(a_2)$ et $taille(a_1) \leq taille(a_2)$.

1. Montrer qu'il existe deux arbres a_1, a_2 tels que $(a_1, a_2) \in R$, $(a_2, a_1) \in R$, et $a_1 \neq a_2$.
2. Prouver que $\mathcal{R}$ est un preordre.
3. Quel est l'ensemble ordonné canoniquement associé à $\mathcal{R}$?
4. L'ordre de cet ensemble ordonné est-il total, bien fondé?

Exercice 2 Calculer la clôture inductive des ensembles de règles suivantes:

1.
$$\frac{}{nil \in \mathcal{A}(\mathbb{N})} \qquad \frac{n \in \mathbb{N} \text{ et } a \in \mathcal{A}(\mathbb{N})}{cons(n, nil, a) \in \mathcal{A}(\mathbb{N})}$$

2.
$$\frac{}{nil \in \mathcal{A}(\mathbb{N})} \qquad \frac{a \in \mathcal{A}(\mathbb{N})}{cons(taille(a), a, a) \in \mathcal{A}(\mathbb{N})}$$

Exercice 3 On considère le type Ocaml `type arbre = Av | Arb of int * arbre * arbre;;`

Un *parcours* d'un arbre est une fonction de type `arbre ->int list` qui renvoi la liste des étiquettes de son argument.

On définit trois parcours, en fonction de l'ordre dans lequel les actions suivantes sont exécutées:

(a) Visiter la racine
(b) Visiter le sous-arbre gauche
(c) Visiter le sous-arbre droit

1. Parcours préfixé: a b c
2. Parcours infixé: b a c

3. Parcours postfixé: b c a

   Définir une fonction pour chaque parcours.
   Tester les trois fonctions sur `Arb(1, Arb (2, Av, Av), Arb (3, Av, Av))`.

Exercice 4 Un arbre binaire de recherche (abr) est un arbre tel que:

pour chaque nœud $n(m, g, d)$, les étiquettes contenues dans le sous-arbre gauche g sont inférieures ou égales à m, et celles contenues dans le sous-arbre droit d sont supérieures à m.

Écrire une fonction `test : arbre ->bool` qui vérifie si un arbre est un abr.

Écrire une fonction `recherche : arbre ->int ->bool` testant la présence d'un élément dans un abr.