LIFLC – Logique classique
TD2 – Induction

Licence informatique UCBL – Automne 2017–2018

Les (parties d') exercices noté(e)s avec † sont plus difficiles.

Exercice 1 : Ensembles inductifs

Soit E et F deux ensembles. Expliquer quels sont les ensembles inductifs définis à partir des 3 ensembles de règles suivants. Expliquer les différences entre ces trois ensembles.

1. Ensemble N
   — $leaf_N \in N$
   — si $n_1 \in N$, $n_2 \in N$ et si $e \in E$ alors $node_N(n_1, e, n_2) \in N$
2. Ensemble L
   — si $e \in E$, alors $leaf_L(e) \in L$
   — si $n_1 \in L$ et $n_2 \in L$ alors $node_L(n_1, n_2) \in L$
3. Ensemble LH
   — si $e \in E$ alors $leaf_{LH}(e) \in LH$
   — si $n_1 \in LH$, $n_2 \in LH$ et $f \in F$, alors $node_{LH}(n_1, f, n_2) \in LH$

Exercice 2 : Ensemble inductif des arbres binaires de recherche

Soit E un ensemble muni d'un ordre total $\leq_E$.

1. Donner une définition par induction de l'ensemble Bin_E des arbres binaires contenant des éléments de E.
2. Définir la fonction récursive *elements* qui renvoie l'ensemble des éléments de E contenus dans un arbre binaire de recherche. On commencera par donner la signature (domaine et co-domaine) de cette fonction.
3. Donner une définition par induction de l'ensemble $BinRech_E$ des arbres binaires de recherche contenant des éléments de E.
4. Définir la fonction récursive *plusPetitElement* qui renvoie le plus petit élément d'un arbre binaire de recherche[1]. Dans le cas où un tel élément n'existe pas, la fonction reverra la valeur spéciale $\lozenge$. On commencera par donner la signature de cette fonction.
5. Montrer que la fonction *plusPetitElement* est correcte. Pour cela, montrer par induction sur Bin_E que : soit $plusPetitElement(a) = min(elements(a))$, soit $elements(a) = \emptyset$ et $plusPetitElement(a) = \lozenge$.

1. on souhaite ici que la complexité de la fonction soit linéaire dans la hauteur de l'arbre

Exercice 3 : Plus petit ensemble stable par des règles de constructions †

Soit E un ensemble. On considère les deux règles de construction de listes suivantes :

- [] est une liste (on note cette règle $\mathcal{R}_1$)
- si l' est une liste et si $e \in E$, alors $cons(e, l')$ est une liste (on note cette règle $\mathcal{R}_2$)

Soit F un ensemble tel qu'il admet au moins un sous-ensemble G stable par ces règles de construction (*i.e.* $[] \in G$ et si $e \in E$ et $l' \in G$, alors $cons(e, l') \in G$). Montrer qu'il existe un unique plus petit (au sens de l'inclusion) sous-ensemble de F stable par ces règles de construction.

Indice : considérer l'intersection de tous les sous-ensembles de F stables par ces règles de construction.

Exercice 4 : Ordre bien fondé sur un ensemble inductif †

Soit E un ensemble. On considère l'ensemble inductif des listes $List_E$ construit à partir des règles suivantes :

- [] est une liste
- si l' est une liste et si $e \in E$, alors $cons(e, l')$ est une liste

Soit la relation binaire $\lhd$ définie sur $List_E \times List_E$ par $l' \lhd l$ si et seulement si on peut trouver $e \in E$ tel que $l = cons(e, l')$. Soit $\leq$ la fermeture réflexive transitive de $\lhd$. On suppose que :

a. *cons* est injective ;
b. qu'il n'existe pas de liste l et d'élément e tels que $[] = cons(e, l)$.

Montrer que $\leq$ est un ordre bien fondé sur $List_E$.

Indices :

- les conditions (a.) et (b.) sont utiles pour montrer l'antisymmétrie. Si $l \leq l'$,$l' \leq l$ et $l \neq l'$, on peut considérer la suite $l_1, \dots, l_k, \dots, l_m$, telle que $l_i \lhd l_{i+1}$, $l_1 = l_m = i$ et $l_k = l'$.
- on pourra montrer par induction que pour toute liste l, il n'existe pas de suite infinie strictement décroissante commençant par une liste $l' \leq l$.