Assignments for the course Functional Programming

Etienne van Delden

April 9, 2012

1 Programming with lists

We consider the datatype $\mathcal{LT}(B)$ of (finite) binary leaf-trees over type B , defined recursively as the smallest of all possible sets satisfying:

$$\langle b \rangle \in \mathcal{LT}(B), \text{ for all } b \in B,$$
$$\langle s, t \rangle \in \mathcal{LT}(B), \text{ for all } s, t \in \mathcal{LT}(B).$$

9. Function L, of type $\mathcal{LT}(B) \to \mathcal{L}_*(B)$, is defined by, for all $b \in B$ and $s, t \in \mathcal{LT}(B)$:

$$L \cdot \langle b \rangle = [b]$$
$$L \cdot \langle s, t \rangle = Ls + Lt$$

$$\ldots$$

This inspires us to consider the following generalization of function L , as a function F of type: $\mathcal{L}_*(\mathcal{LT}(B)) \to \mathcal{L}_*(B) \to \mathcal{L}_*(B)$, and with this specification, for all $ss \in \mathcal{L}_*(\mathcal{LT}(B))$ and $z \in \mathcal{L}_*(B)$:

$$F \cdot ss \cdot z = flt(L \bullet rev \cdot ss) + z.$$

(a) Prove that, as stated above, $L \cdot s +\!\!+ L \cdot t = flt \cdot (L \bullet [s, t])$.

$$\begin{array}{lll} & & flt \cdot (L \bullet [s, t]) \\ = & \{def \bullet\} & flt \cdot (L \cdot s +\!\!+ L \cdot t) \\ = & \{def flt\} & flt \cdot (L \cdot s) +\!\!+ flt \cdot (L \cdot t) \\ = & \{def flt\} & L \cdot s +\!\!+ L \cdot t \end{array}$$

(b) Show that function F indeed is a generalization of L , by showing how L can be defined in terms of F .

(c) Derive an efficient recursive declaration for function F in which L does not occur anymore. (d) Explain (the usefulness of) the presence of function rev in F s specification.

References

[XXXX] Name, *Title,* vol. X. pp. XX–YY. Editor, Town, Year.