

Homework 4: Inductive and Co-inductive Types

15-814: Type Systems for Programming Languages

TA: Kumar Avijit (kavijit@cs.cmu.edu)

Out: October 10, 2009

Due: October 16, 2009 (11:59 PM)

1 Covariant action of type operators

In this section, we are going to recap the covariant action of type operators. For this question, we are concerned with the following open types (open because they may include a type variable):

$$\begin{array}{l} \text{Types } A ::= X \mid \mathbf{1} \mid \mathbf{0} \mid A_1 \rightarrow A_2 \mid A_1 + A_2 \mid A_1 \times A_2 \\ \text{Terms } m ::= x \mid \langle \rangle \mid \text{abort}_A(m) \mid \lambda x:A.m \mid \text{ap}(m_1; m_2) \\ \quad \mid \langle m_1; m_2 \rangle \mid \text{fst}(m) \mid \text{snd}(m) \mid \text{inl}_{A_1, A_2}(m) \\ \quad \mid \text{inr}_{A_1, A_2}(m) \mid \text{case } m \{ \text{inl } x \Rightarrow m_1 \mid \text{inr } y \Rightarrow m_2 \} \end{array}$$

Here X denotes a type variable. A positive type operator $X.A$ is defined using judgment of the form $\Delta \vdash X.A \text{ pos}$, where Δ is a context of type variables:

$$\Delta ::= \cdot \mid \Delta, X \quad (X \notin \Delta)$$

$$\begin{array}{c} \frac{}{\Delta \vdash X.X \text{ pos}} \qquad \frac{\Delta \vdash X.A_1 \text{ pos} \quad \Delta \vdash X.A_2 \text{ pos}}{\Delta \vdash X.A_1 \times A_2 \text{ pos}} \\ \frac{\Delta \vdash X.A_1 \text{ pos} \quad \Delta \vdash X.A_2 \text{ pos}}{\Delta \vdash X.A_1 + A_2 \text{ pos}} \qquad \frac{\Delta \vdash A_1 \text{ type} \quad \Delta, X \vdash X.A_2 \text{ pos}}{\Delta, X \vdash X.A_1 \rightarrow A_2 \text{ pos}} \\ \frac{}{\Delta \vdash X.\mathbf{1} \text{ pos}} \qquad \frac{}{\Delta \vdash X.\mathbf{0} \text{ pos}} \end{array}$$

In the above definition, we need an auxiliary judgment $\Delta \vdash A \text{ type}$, which says that A is a type under the context Δ .

$$\begin{array}{c} \frac{}{\Delta \vdash \mathbf{1} \text{ type}} \qquad \frac{}{\Delta \vdash \mathbf{0} \text{ type}} \qquad \frac{\Delta \vdash A_1 \text{ type} \quad \Delta \vdash A_2 \text{ type}}{\Delta \vdash A_1 \rightarrow A_2 \text{ type}} \\ \frac{\Delta \vdash A_1 \text{ type} \quad \Delta \vdash A_2 \text{ type}}{\Delta \vdash A_1 \times A_2 \text{ type}} \qquad \frac{\Delta \vdash A_1 \text{ type} \quad \Delta \vdash A_2 \text{ type}}{\Delta \vdash A_1 + A_2 \text{ type}} \\ \frac{}{\Delta, X \vdash X \text{ type}} \end{array}$$

The action of type operator $X.A$ on a type B , denoted by $\text{Map}[X.A](B)$, is simply $[B/X]A$.

Notice that for a type operator of the shape $X.(A_1 \rightarrow A_2)$ to be positive, A_1 should not contain any occurrences of X . This in turn implies the following:

Lemma 1 (–do not prove–). *If $X \vdash X.(A_1 \rightarrow A_2)$ pos then $\text{Map}[X.(A_1 \rightarrow A_2)](B) = A_1 \rightarrow \text{Map}[X.A_2](B)$.*

The action on functions is defined inductively below:

$$\begin{array}{c}
\frac{}{\text{map}[X.1](m) = \lambda x:1.\langle \rangle} (\mathbf{1}\text{-map}) \qquad \frac{}{\text{map}[X.0](m) = \lambda x:0.\text{abort}_0(x)} (\mathbf{0}\text{-map}) \\
\\
\frac{}{\text{map}[X.X](m) = m} (X\text{-map}) \\
\\
\frac{m : B_1 \rightarrow B_2 \quad \text{map}[X.A_2](m) = m'}{\text{map}[X.A_1 \rightarrow A_2](m) = \lambda f:\text{Map}[X.A_1 \rightarrow A_2](B_1).\lambda x:\text{Map}[X.A_1](B_2).\text{ap}(m'; \text{ap}(f; x))} (\rightarrow\text{-map}) \\
\\
\frac{m : B_1 \rightarrow B_2 \quad \text{map}[X.A_1](m) = m_1 \quad \text{map}[X.A_2](m) = m_2}{\text{map}[X.A_1 \times A_2](m) = \lambda x:\text{Map}[X.A_1 \times A_2](B_1).\langle \text{ap}(m_1; \text{fst}(x)); \text{ap}(m_2; \text{snd}(x)) \rangle} (\times\text{-map}) \\
\\
\frac{m : B_1 \rightarrow B_2 \quad \text{map}[X.A_1](m) = m_1 \quad \text{map}[X.A_2](m) = m_2 \quad A'_1 = \text{Map}[X.A_1](B_2) \quad A'_2 = \text{Map}[X.A_2](B_2)}{\text{map}[X.A_1 + A_2](m) = \lambda x:\text{Map}[X.A_1 + A_2](B_1).\text{case } x \{ \text{inl } x \Rightarrow \text{inl}_{A'_1, A'_2}(\text{ap}(m_1; x)) \mid \text{inr } x \Rightarrow \text{inr}_{A'_1, A'_2}(\text{ap}(m_2; x)) \}} (+\text{-map})
\end{array}$$

Task 1. *Prove that if $X \vdash X.A$ pos, and, $m : B_1 \rightarrow B_2$, and, $\text{map}[X.A](m) = m'$, then $m' : \text{Map}[X.A](B_1) \rightarrow \text{Map}[X.A](B_2)$.*

2 Regular Trees

2.1 Definition

A regular tree is an infinite structure that is “closed” under the following operations:

- $\text{node}(m)$ returns the element at the root node. We will consider trees with natural numbers at the nodes.
- $\text{left}(m)$ returns the left sub-tree, and
- $\text{right}(m)$ returns the right sub-tree.

Let us introduce a new type `rtree` for regular trees. We define this type co-inductively using the following elimination forms:

$$\frac{\Gamma \vdash m : \mathbf{rtree}}{\Gamma \vdash \mathbf{node}(m) : \omega} (\mathbf{rtree}\text{-E}_n) \qquad \frac{\Gamma \vdash m : \mathbf{rtree}}{\Gamma \vdash \mathbf{left}(r) : \mathbf{rtree}} (\mathbf{rtree}\text{-E}_l)$$

$$\frac{\Gamma \vdash m : \mathbf{rtree}}{\Gamma \vdash \mathbf{right}(r) : \mathbf{rtree}} (\mathbf{rtree}\text{-E}_r)$$

The introductory form assembles a tree by having a seed element that serves as the internal state, and two functions, to describe the evolution of the left and right parts. In addition there is a function to map the internal state to a natural number:

$$\frac{\Gamma \vdash m : A \quad \Gamma \vdash n : A \rightarrow \omega \quad \Gamma \vdash l : A \rightarrow A \quad \Gamma \vdash r : A \rightarrow A}{\Gamma \vdash \mathbf{tree} m.\{\mathbf{nd} \Rightarrow n \ \& \ \mathbf{lt} \Rightarrow l \ \& \ \mathbf{rt} \Rightarrow r\} : \mathbf{rtree}}$$

The dynamic semantics is given by the following rules:

$$\frac{}{\mathbf{node}(\mathbf{tree} m.\{\mathbf{nd} \Rightarrow n \ \& \ \mathbf{lt} \Rightarrow l \ \& \ \mathbf{rt} \Rightarrow r\}) \rightsquigarrow \mathbf{ap}(n; m)} (\mathbf{rtree} \rightsquigarrow_n)$$

$$\frac{}{\mathbf{left}(\mathbf{tree} m.\{\mathbf{nd} \Rightarrow n \ \& \ \mathbf{lt} \Rightarrow l \ \& \ \mathbf{rt} \Rightarrow r\}) \rightsquigarrow \mathbf{tree} \mathbf{ap}(l; m).\{\mathbf{nd} \Rightarrow n \ \& \ \mathbf{lt} \Rightarrow l \ \& \ \mathbf{rt} \Rightarrow r\}} (\mathbf{rtree} \rightsquigarrow_l)$$

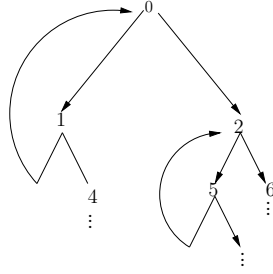
$$\frac{}{\mathbf{right}(\mathbf{tree} m.\{\mathbf{nd} \Rightarrow n \ \& \ \mathbf{lt} \Rightarrow l \ \& \ \mathbf{rt} \Rightarrow r\}) \rightsquigarrow \mathbf{tree} \mathbf{ap}(r; m).\{\mathbf{nd} \Rightarrow n \ \& \ \mathbf{lt} \Rightarrow l \ \& \ \mathbf{rt} \Rightarrow r\}} (\mathbf{rtree} \rightsquigarrow_r)$$

Task 2. What terms represent the following regular trees:

1. $T(\mathbf{z})$ where $T(n) ::= \mathbf{Tree}(n, T(2n+1), T(2n+2))$. The notation $\mathbf{Tree}(n, T_1, T_2)$ represents a tree with n being the node element, and T_1 and T_2 being the left and right sub-trees respectively. We assume multiplication and addition operations on elements of type ω without defining them.
2. The tree depicted in the following figure, which has a similar structure as the previous tree, except that for every node n , the left child of the left child of n is n .

2.2 Characterization using co-inductive types

In the class, we gave an account of the type ω of natural numbers in a language with inductive types. We will do a similar formalization of the type \mathbf{rtree} here, but using co-inductive types instead. We use the language $\mathcal{L}\{\mu_i, \mu_f\}$ from Chapter 19 of PFPL. Please refer to the chapter for its definition. We use the notation $\nu X.A$ from the lecture to denote co-inductive types, instead of the book's notation $\mu_f(t, \tau)$.



Task 3. Characterize the type `rtree` as a co-inductive type, $\nu X.A$. Check for yourself that $X.A$ in your definition is a positive type operator.

A co-inductive type $T = \nu X.A$ has an associated $\text{unfold}[X.A](m)$ operation. We saw in the class that for any term $x.m : B \rightarrow \text{Map}[X.A](B)$, there exists a unique term $\text{gen}[X.A](x.m_1; m_2)$ for which the following diagram commutes. Here Map and map are the covariant actions of the type operator on types and abstractors resp. (we did not show uniqueness and will ignore it for the time being).

$$\begin{array}{ccc}
 \text{Map}[X.A](T) & \xleftarrow{x.\text{ap}(f; x)} & \text{Map}[X.A](B) \\
 \uparrow y.\text{unfold}[X.A](y) & & \uparrow x.m \\
 T & \xleftarrow{y.\text{gen}[X.A](x.m; y)} & B
 \end{array}$$

where $\text{map}[X.A](\lambda y:B.\text{gen}[X.A](x.m; y)) = f$.

In the diagram, we use the notation $A \xrightarrow{x.m} B$ to denote a term m of type B with a free variable x of type A . Sequential composition of arrows denotes composition; thus $A \xrightarrow{x.m} B \xrightarrow{y.n} C$ denotes the term $x.[m/y]n$, after suitably renaming x, y to avoid capture.

Task 4. Express the introductory and elimination forms of the type `rtree` in terms of $\text{unfold}[X.A]()$ and $\text{gen}[X.A](x.m;)$, where $X.A$ is the type operator you defined in Task 3. In order to express 3-tuples, you can generalize pairs to n -ary products as given in Appendix.

A Products

$$\frac{\Gamma \vdash m_1 : A_1 \quad \dots \quad \Gamma \vdash m_n : A_n}{\Gamma \vdash \langle m_1, \dots, m_n \rangle : A_1 \times \dots \times A_n} \quad \frac{\Gamma \vdash m : A_1 \times \dots \times A_n}{\Gamma \vdash \pi_i(m) : A_i} (i \in [1..n])$$