Metatheory of Recursive Types

Types and Programming Langages - B. Pierce

07/02/2003

Élodie-Jane Sims.

Subject

– Checking membership in greatest fixed points

– Application to subtyping

Induction & Coinduction

$\mathcal{U}$: universal set
$F \in \mathcal{P}(\mathcal{U}) \to \mathcal{P}(\mathcal{U})$: generating function (monotone)

– X is F−closed : $F(X) \subseteq X$

– X is F−consistent : $X \subseteq F(X)$

– X is a fixed point of F : $X = F(X)$

– Th Knaster-Tarski
  - ♦ $\mu F = \bigcap\{X \mid F(X) \subseteq X\}$ (least fixed point)
  - ♦ $\nu F = \bigcup\{X \mid X \subseteq F(X)\}$ (greatest fixed point)

Principle of induction $F(X) \subseteq X \Rightarrow \mu F \subseteq X$
"To show that μF has a property we just show that this property is preserved by F"

Principle of coinduction $X \subseteq F(X) \Rightarrow X \subseteq \nu F$
"To check weither $x \in \nu F$ we just need an F-consistent set X containing x"

A Disgression on Transitivity

2 starndard formulations :

– declarative presentation with explicit transitivity rule

– algorithmic presentation without explicit transitivity rule

The declarative one work for induction definitions (finite case) but not for coinduction (infinite case).

The problem is to apply the rule $\frac{S<:U \ U<:T}{S<:T}$ *how to find* U *?*

Membership Checking

$x \in \mathcal{U}$, F generating function : $x \in \nu F$?

Idea of algo : start from νF and go backward with "F^{-1}" until reaching x or not.
Problems : there can be several way to go backward, danger of combinatorial explosion.

- **Generating set** $G_x = \{X \subseteq \mathcal{U} |\ x \in F(X)\}$
- F **invertible** : $\forall x \in \mathcal{U}$.
  G_x has at most one smallest element
- **Support set** ("F^{-1}") if F is invertible :

$$support_F(x) = \begin{cases} X \text{ if } X \in G_x \text{ and } \forall X' \in G_x . X \subseteq X' \\ \uparrow \text{ if } G_x = \emptyset \end{cases}$$

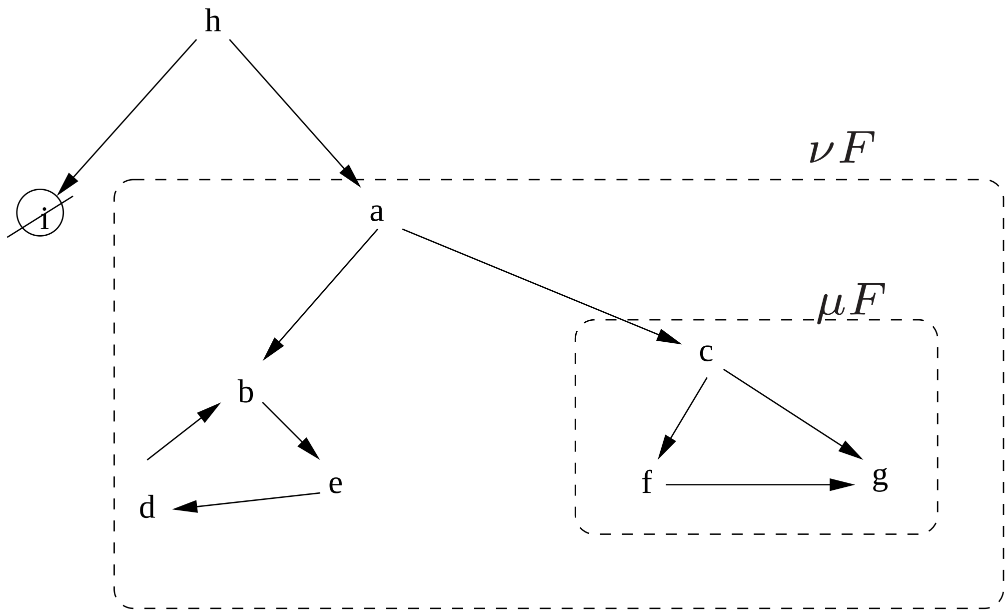
- x is F-supported if $support_F(x) \downarrow$
- x is F-unsupported if $support_F(x) \uparrow$
- **Support graph**:
  - nodes = F-supported and F-unsupported elements of $\mathcal{U}$
  - edge (x,y) if $y \in support_F(x)$

$\{F$-unsupported in the support graph$\} \cap reachable_F(x) = \emptyset$?

F **finite state** if $\forall x . reachable_F(x)$ is finite

$\Rightarrow$ algo terminate if F is finite state

Example



Rules $\frac{}{g}$ $\frac{g}{f}$ $\frac{g}{c}$ $\frac{b}{d}$ $\frac{d}{e}$ $\frac{c}{b}$ $\frac{b,c}{a}$ $\frac{i,a}{h}$

F :

$$F(\emptyset) = \{g\}$$

$$F(\{g\}) = \{f\}$$

...

The algorithm and the principle of coinduction

I write F^{-1} without having it defined, it's just a hand show of how the coinduction principle is used.

We take $\{x\}$, compute $F^{-1}(\{x\})$, $F^{-1}(F^{-1}(\{x\}) \cup \{x\})$,...,$F^{-1}(F^{-1}(X_n) \cup X_n)$

Until $F^{-1}(F^{-1}(X_n) \cup X_n) \subseteq F^{-1}(X_n) \cup X_n$

Then this last set is F-consistent since :

$F^{-1}(X_n) = F(F^{-1}(F^{-1}(X_n) \cup X_n))$

and we also have x in it so by the principle of coinduction $x \in \nu F$

Tree-Types : univers $\mathcal{U}$

$\rightarrow$, $\times$, $\mathtt{Top}$: types constructors
$\bullet$: empty sequence
π, σ : concatenation of sequences

Tree-type partial function
$\mathtt{T} \in \{1,2\}^* \rightharpoonup \{\rightarrow, \times, \mathtt{Top}\}$ satisfying
- $\mathtt{T}(\bullet)$ defined
- $\mathtt{T}(\pi, \sigma)$ defined $\Rightarrow \mathtt{T}(\pi)$ defined
- $\mathtt{T}(\pi) = \rightarrow$ or $\times \Rightarrow \mathtt{T}(\pi, 1)$, $\mathtt{T}(\pi, 2)$ defined
- $\mathtt{T}(\pi) = \mathtt{Top} \Rightarrow \mathtt{T}(\pi, 1)$, $\mathtt{T}(\pi, 2)$ undefined

finite tree-type : $dom(\mathtt{T})$ finite
$\mathcal{T}$: set of all tree-types
$\mathcal{T}_{\mathsf{f}}$: set of all finite tree-types

$$\begin{array}{lcl} \mathtt{T} & := & \mathtt{Top} \\ & | & \mathtt{T} \times \mathtt{T} \\ & | & \mathtt{T} \rightarrow \mathtt{T} \end{array}$$

$\mathcal{U} =$ all finite and ∞ trees labelled with $\mathtt{Top}, \rightarrow, \times$
$\mathcal{T} = gfp$ of the generating function described by the grammar above
$\mathcal{T}_{\mathsf{f}} = lfp$ ...

Subtyping : *generating functions* F's

Finite subtyping : S is a subtype of T iff $(S,T) \in \mu S_f$

$$
\begin{array}{rcl}
S_f & : & \mathcal{P}(\mathcal{T}_{\mathsf{f}} \times \mathcal{T}_{\mathsf{f}}) \rightarrow \mathcal{P}(\mathcal{T}_{\mathsf{f}} \times \mathcal{T}_{\mathsf{f}}) \\
S_f(R) & = & \{(T, \mathtt{Top}) \mid T \in \mathcal{T}_{\mathsf{f}}\} \\
& \cup & \{(S_1 \times S_2, T_1 \times T_2) \mid (S_1, T_1), (S_2, T_2) \in R\} \\
& \cup & \{(S_1 \rightarrow S_2, T_1 \rightarrow T_2) \mid (T_1, S_1), (S_2, T_2) \in R\}
\end{array}
$$

Infinite subtyping : S is a subtype of T iff $(S,T) \in \nu S$

$$
\begin{array}{rcl}
S & : & \mathcal{P}(\mathcal{T} \times \mathcal{T}) \rightarrow \mathcal{P}(\mathcal{T} \times \mathcal{T}) \\
S(R) & = & \{(T, \mathtt{Top}) \mid T \in \mathcal{T}\} \\
& \cup & \{(S_1 \times S_2, T_1 \times T_2) \mid (S_1, T_1), (S_2, T_2) \in R\} \\
& \cup & \{(S_1 \rightarrow S_2, T_1 \rightarrow T_2) \mid (T_1, S_1), (S_2, T_2) \in R\}
\end{array}
$$

Regular Trees

Motivations : if types are "regular" then reachable sets remain finites and subtype checking algo always terminate.

S **sub-tree** T : $\exists \pi . S = \lambda\sigma.\mathtt{T}(\pi, \sigma)$

Regular tree : the set of subtrees is finite

$\mathcal{T}_{\mathsf{r}}$= all regular tree-types

S_{r} = restriction of S to $\mathcal{T}_{\mathsf{r}}$

$\boxed{S_{\mathsf{r}} \text{ is finite state}}$

➥ we can have a decision procedure which terminate but we need a finite representation for regular trees

μ-Types

$\{X_i\}$ fixed countable set of type variables

$\mathcal{T}_{\mathsf{m}}^{\mathsf{raw}}$: set of raw μ-types, set of expressions :

$$\begin{array}{lcl} \mathtt{T} & ::= & \mathtt{X} \\ & \mid & \mathtt{Top} \\ & \mid & \mathtt{T} \times \mathtt{T} \\ & \mid & \mathtt{T} \to \mathtt{T} \\ & \mid & \mu\mathtt{X}.\mathtt{T} \end{array}$$

raw μ-type contractive : for all subexpressions of the form $\mu\mathtt{X}.\mu\mathtt{X}_1 \ldots \mu\mathtt{X}_\mathtt{n}.S$ we have $S \neq \mathtt{X}$.
(because $\mu\mathtt{X}.\mathtt{X}$ can't be interpreted as a type)

$\mu - type$: contractive raw μ-type

$\mathcal{T}_{\mathsf{m}}$: set of μ-types

μ-types subtyping : S is a subtype of T iff $(S, T) \in \nu S_m$

$$\begin{array}{rcl} S_m & : & \mathcal{P}(\mathcal{T}_{\mathsf{m}} \times \mathcal{T}_{\mathsf{m}}) \to \mathcal{P}(\mathcal{T}_{\mathsf{m}} \times \mathcal{T}_{\mathsf{m}}) \\ S_m(R) & = & \{(T, \mathtt{Top}) \mid T \in \mathcal{T}_{\mathsf{m}}\} \\ & \cup & \{(S_1 \times S_2, T_1 \times T_2) \mid (S_1, T_1), (S_2, T_2) \in R\} \\ & \cup & \{(S_1 \to S_2, T_1 \to T_2) \mid (T_1, S_1), (S_2, T_2) \in R\} \\ & \cup & \{(S, \mu\mathtt{X}.T) \mid (S, [\mathtt{X} \mapsto \mu\mathtt{X}.T]T) \in R\} \\ & \cup & \{(\mu\mathtt{X}.S, T) \quad \mid \quad ([\mathtt{X} \mapsto \mu\mathtt{X}.S]S, T) \in R \\ & & \qquad\qquad\qquad T \neq \mathtt{Top} \\ & & \qquad\qquad\qquad T \neq \mu\mathtt{Y}.T_1\} \end{array}$$

the last adding conditions are for reversibility

Link from μ-types to tree-types

$treeof$: closed μ-types $\longrightarrow$ tree-types

$$
\begin{array}{lcl}
treeof(\mathtt{Top})(\bullet) & = & \mathtt{Top} \\
treeof(\mathtt{T}_1 \rightarrow \mathtt{T}_2)(\bullet) & = & \rightarrow \\
treeof(\mathtt{T}_1 \times \mathtt{T}_2)(\bullet) & = & \times \\
treeof(\mathtt{T}_1 \rightarrow \mathtt{T}_2)(i,\pi) & = & treeof(\mathtt{T}_i)(\pi) \\
treeof(\mathtt{T}_1 \times \mathtt{T}_2)(i,\pi) & = & treeof(\mathtt{T}_i)(\pi) \\
treeof(\mu\mathtt{X}.\mathtt{T})(\pi) & = & treeof([X \mapsto \mu\mathtt{X}.\mathtt{T}]\mathtt{T})(\pi)
\end{array}
$$

$$\boxed{(S,T) \in \nu S_m \text{ iff } treeof(S,T) \in \nu S}$$

Instanciation of the general algorithm for μ-types

$$support_{S_m}(S,T) = \begin{cases} \emptyset & \text{if} \quad T = \mathtt{Top} \\ \{(S_1,T_1),(S_2,T_2)\} & \text{if} \quad S = S_1 \times S_2 \text{ and} \\ & \phantom{\text{if}} \quad T = T_1 \times T_2 \\ \{(S_1,T_1),(S_2,T_2)\} & \text{if} \quad S = S_1 \to S_2 \text{ and} \\ & \phantom{\text{if}} \quad T = T_1 \to T_2 \\ \{(S,[\mathtt{X} \mapsto \mu\mathtt{X}.T_1]T_1)\} & \text{if} \quad T = \mu\mathtt{X}.T_1 \\ \{([\mathtt{X} \mapsto \mu\mathtt{X}.S_1]S_1,T_1)\} & \text{if} \quad S = \mu\mathtt{X}.S_1 \text{ and} \\ & \phantom{\text{if}} \quad T \neq \mu\mathtt{X}.T_1,\ T \neq \mathtt{Top} \\ \uparrow & \phantom{\text{if}} \quad \text{otherwise} \end{cases}$$

S_m is invertible because $support_{S_m}$ is well-defined.

$\boxed{\text{For any } \mu\text{-types } S,\ T,\ reachables_{S_m}(S,T) \text{ is finite}}$