Program Extraction (part 7)

Kenji Miyamoto (LMU)

3.10 – 8.10 Aurachhof, Fischbachau.

BHK - interpretation

(i) p proves $A \to B$ iff p is a construction transforming any construction q of A into a construction $p(q)$ of B.

(ii) $\bot$ (falsity) is a proposition without construction.

(iii) p proves $\forall x \in D\, A$ iff p is a construction s.t. for all $d \in D$ $p(d)$ proves $A(d)$.

- Theory of Computable Functionals (TCF)
  - Formal theory for program extraction
  - Behind theory of the proof assistant Minlog
- Application
  - Example of program extraction
  - See how it works in Minlog

Types and Algebras

Simple types with algebras as ground types.

$$\tau, \tau' ::= \alpha \mid \tau \to \tau' \mid \mu_\xi \left(\left(\rho_{ij}(\xi) \right)_{j<n_i} \to \xi \right)_{i<k}$$

where α, ξ are type variables, and

ξ can occur strictly positively in $\rho_{ij}(\xi)$.

For an algebra $\iota := \mu_\xi \left(\left(\rho_{ij}(\xi) \right)_{j<n_i} \to \xi \right)_{i<k}$,

each $\left(\rho_{ij}(\iota) \right)_{j<n_i} \to \iota$ is a constructor type,

and we provide a constructor C_i of this type.

Examples of algebras

$\mu_\xi(\xi, \xi \to \xi) =: \mathbb{N}$ (Natural numbers)

$\mu_\xi(\xi, \alpha \to \xi \to \xi) =: \mathbb{L}_\alpha$ (Lists of α)

$\mu_\xi(\alpha \to \xi, \beta \to \xi) =: \alpha + \beta$

$\mu_\xi(\alpha \to \beta \to \xi) =: \alpha \times \beta$

Terms

We take an extension of Gödel's T.

$$t, t' ::= x^{\tau} \mid \lambda x^{\tau} t \mid t t' \mid C \mid D$$

where τ ranges over types,

l over algebras,

C over constructors, eg. $0^{\mathbb{N}}, S^{\mathbb{N} \to \mathbb{N}}$

D over defined constants

Defined Constants with Computation Rules

for a defined constant D we can give equations $D\vec{P}_i(\vec{x}_i) = M_i$ where if $i \neq j$

- $\vec{P}_i$ and $\vec{P}_j$ are non-unifiable
- $\vec{P}_i$ and $\vec{P}_j$ have disjoint free variables

Examples of Defined Constants

For each algebra $L = \mu_\xi(K_0(\xi), K_1(\xi), \cdots, K_{k-1}(\xi))$

we define recursion operator $\mathcal{R}_L^\tau$ whose type is

$$L \to \delta_0(L,\tau) \to \delta_1(L,\tau) \to \cdots \to \delta_{k-1}(L,\tau) \to \tau$$

where for $K_i(\xi) = (P_{iv}(\xi))_{v<n_i} \to \xi$

let $\delta_i(L,\tau)$ be $(P_{iv}(L \times \tau))_{v<n_i} \to \tau$.

Conventionally, we remove "x" by currying.

Formulas and Predicates

We simultaneously define formulas A and predicates P.

$$A ::= P\vec{t} \mid \forall_x A' \mid A' \to A''$$

$$P ::= Q \mid \{\vec{x} \mid A\} \mid \mu_X^{\rho}\left(\forall_{\vec{x}_i}\left(\left(B_{iv}(X)\right)_{v<n_i} \to X\vec{t}_i\right)\right)_{i<k}$$

where $Q, X \in \mathrm{Pvars}$ and

X can occur strictly positively in formulas $B_{iv}(X)$.

Formulas and Predicates

We simultaneously define formulas A and predicates P.

$$A ::= P\vec{t} \mid \forall_x A' \mid A' \to A'' \mid \forall_x^{nc} A' \mid A' \to^{nc} A''$$

$$P ::= Q \mid \{\vec{x} \mid A\} \mid \mu_X\left(\forall_{\vec{x}_i}^{(nc)}\left(\left(B_{iv}(X)\right)_{v<n_i} \to^{(nc)} X\vec{t}_i\right)\right)_{i<k}$$

$$\mid Q^{nc} \mid \mu_X^{nc}\left(\forall_{\vec{x}_i}\left(\left(B_{iv}(X)\right)_{v<n_i} \to X\vec{t}_i\right)\right)_{i<k}$$

where $Q, X \in Pvars$, $Q^{nc} \in$ n.c. $Pvars$ and X can occur strictly positively in formulas $B_{iv}(X)$

Non-computational formulas / predicates

Formula C is non-computational if

$fp(C)$ is either X^{nc} or I^{nc}

$fp(X) = X$, $fp(X^{nc}) = X^{nc}$, $fp(P\vec{F}) = fp(P)$,

$fp(\mathcal{I}) = \mathcal{I}$, $fp(\mathcal{I}^{nc}) = \mathcal{I}^{nc}$,

$fp(\{\vec{x} \mid A\}) = fp(A)$,

$fp(A \overset{(nc)}{\to} B) = fp(B)$, $fp(\forall^{(nc)}_x A) = fp(A)$

Introduction and Elimination Axioms

Inductive definition $I \stackrel{(nc)}{=} \mu_X^{(nc)}(\ldots)$ extends our first-order minimal logic by adding introduction axioms I_i^+

$$\forall \vec{x}_i \left(\left(B_{iv}(I) \right)_{v<n_i} \to I\, \vec{t}_i \right)$$

and an elimination axiom I^-

$$\forall \vec{x} \left(I\, \vec{x} \to \left(\forall \vec{x}_i \left(\left(B_{iv}(I \cap P) \right)_{v<n_i} \to P\, \vec{t}_i \right) \right)_{i<k} \to P\, \vec{x} \right)$$

where P is a so-called competitor predicate

Elimination rules of I^{nc}.

$(I^{nc})^+_i$ is same as I^+_i.

Competitor predicate of $(I^{nc})^-$ must be n.c.

$$\forall_{\vec{x}}\Big(I^{nc}\vec{x} \to \big(\forall_{x_i}\big((\mathrm{Bi}_\nu(I^{nc} \cap P^{nc}))_{\nu<n_i} \to P^{nc}\vec{t}_i\big)\big)_{i<k} \to P^{nc}\vec{x}\Big)$$

Exceptionally, for $I^{nc} = \mu^{nc}_X\big(\forall^{nc}_{\vec{x}}((A_\nu)_{\nu<n_0} \to^{nc} X\vec{t}\,)\big)$,

we call I^{nc} one clause non-computational idpc,

and <u>allow</u> the competitor to be computational.

Proofs and derivations

We extend natural deduction by adding

- axioms C^A as proof constants.
- introduction rule $(\to^{nc})^+$
- elimination rule $(\to^{nc})^-$
- introduction rule $(\forall^{nc})^+$
- elimination rule $(\forall^{nc})^-$.

- Elimination rules are same as $\to^-$, $\forall^-$.
- We put additional constraints to $\to^+$, $\forall^+$ to formulate $(\to^{nc})^+$, $(\forall^{nc})^+$

Proof terms

derivation	proof term
$u : A$	u^A
$\dfrac{\begin{array}{c}[u:A] \\ \vert M \\ B\end{array}}{A \to B} \to^+$	$(\lambda u^A M^B)^{A \to B}$
$\dfrac{\vert M \quad \vert N}{A \to B \quad A \quad B} \to^-$	$(M^{A \to B} N^A)^B$

Proof terms (cont.)

derivation	proof term
$\dfrac{\begin{array}{c}\vdots M \\ A(x)\end{array}}{\forall_x A(x)}\,\forall^+$	$(\lambda x\, M^{A(x)})^{\forall_x A(x)}$
$\dfrac{\begin{array}{cc}\vdots M & \\ \forall_x A(x) & r\end{array}}{A(r)}\,\forall^-$	$(M^{\forall_x A(x)}\, r)^{A(r)}$
$C : A$	C^A

- for $\to^{nc}$, $\forall^{nc}$, we just use the same syntax of $\to, \forall$ but with "nc"