Type Theory with Abstract Reals under construction

Sewon Park

May 16, 2019

Motivation: Introduction to Type Theory

Type theory:

- Construction rules of types;
  - when A and B are types $A \to B$ is type
  - when A is a type and $a, b : A$ then $a =_A b$ is type
- Construction and elimination rules for elements
  - when $f : B$ assuming $x : A$ then $\lambda x.\ f : A \to B$
  - when $f : A \to B$ and $x : A$ then $f\ x : B$
  - when $x : A$ then there is a constant refl_A such that $\mathsf{refl}_A\ x : x =_A x$

Motivation: Introduction to Type Theory

Type theory:

- Product type $(p, q) : P \wedge Q$.
- Coproduct $\mathsf{inl}\, p, \mathsf{inr}\, q : P \vee Q$
- Dependent Function Type $\forall x : A.\ B(x)$
  - $\lambda x : A.\ t : \forall x : A.\ B(x)$ when $t : B(x)$ assuming $x : A$.
  - $p : \Pi_{x:\mathbb{N}} x =_{\mathbb{N}} x$ gives for any $x : \mathbb{N}$ a term $t : x =_{\mathbb{N}} x$.
- Dependent Pair $\exists x : A.\ B(x)$
  - $(x, t) : \exists x : A.\ B(x)$ where $x : A$ and $t : B(a)$
  - $(p, t) : \Sigma_{x:\mathbb{N}} x =_{\mathbb{N}} 42$ is a pair when $p := 42$, then $t : 42 =_{\mathbb{N}} 42$

Motivation: Introduction to Type Theory

Foundation of constructive mathematics:

- Reading certain types as propositions, term as proof, $\wedge$ as and, $\vee$ as or, $\forall$ as for all and $\exists$ to there exists.
- proving $P \vee Q$ requires to specify which one of the two holds
- proving $\exists x : A.\ B(x)$ requires to state $x : A$ then prove $t : B(x)$

internalizes BHK interpretation:

- proof is an object that can be manipulated: evaluated, transformed, etc.
- a proof of $t : \forall x : A.\ C(x)$ is already a lambda term in the language

Motivation: Mere Proposition

A proof (term) of $\forall x \exists y C(x, y)$ is $x \mapsto (y, t)$ where $t : C(x, y)$.
Proving $t : C(x, y)$ doesn't need to be constructive.

Example

$\mathsf{perm} : \forall X : \mathsf{list}\ A.\ \exists Y : \mathsf{list}\ A.\ \forall i : \mathbb{N}.\ \exists j : \mathbb{N}.\ Y[i] = X[j]$

Sometimes $\exists$ and $\vee$ are too strong!

- weak coproduct: $\forall P.\ P \tilde{\vee} \neg P$
- weak existence: $\neg \exists k.\ P \to \tilde{\exists} k.\ P$
- such weak type lives in prop .

Motivation: Reals in Type Theory

Construct it:

- $\mathsf{seq} := \mathbb{N} \to \mathbb{Q}$
- $\mathsf{Cauchy} := \lambda_{r:\mathsf{seq}}.\ \forall_{n,m:\mathbb{N}} |r\ n - r\ m| < 2^{-n-m}$
- $\mathbb{R} := \exists_{r:\mathsf{seq}} \mathsf{Cauchy}\ r$
- $x \simeq_{\mathbb{R}} y := \forall_{n:\mathbb{N}} |x\ n - y\ n| < 2^{-n}$ *(this is not $x =_{\mathbb{R}} y$)*

or have it as axiom:

- $\mathbb{R}$ is a type constant
- $0, 1 : \mathbb{R}$ are term constants
- $\mathsf{plus} : \mathbb{R} \to \mathbb{R} \to \mathbb{R}$ is a term constant
- $\mathsf{commutative}$ is a term constant of type $\forall_{x,y:\mathbb{R}}\ \mathsf{plus}\ x\ y = \mathsf{plus}\ y\ x$

Motivation: What is the correct set of axioms?

In Computable Analysis:

- $x > y$ is not decidable but is semi-decidable
- Given two semi-decidable predicates you can choose one nondeterministically: (iRRAM choose and ERC select)
- $\mathrm{sign}(x) := \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$ is not computable
- $\mathrm{softsign}(x, \varepsilon) := \begin{cases} 1 & \text{if } x > -\varepsilon \\ 0 & \text{if } x < \varepsilon \end{cases}$ is computable (essentially) multivalued function.

Motivation: What is the correct set of axioms?

Consider the three order axioms:

- Totality : $\forall x : \mathbb{R}.\ x > 0 \vee x = 0 \vee x < 0$
- Wtotality : $\forall x : \mathbb{R}.\ x > 0\ \tilde{\vee}\ x = 0\ \tilde{\vee}\ x < 0$
- softsign : $\forall_{x\ \varepsilon:\mathbb{R}}\ \varepsilon > 0 \to (x > -\varepsilon) \vee (x < \varepsilon)$

Fix x and $\varepsilon > 0$. softsign $x\ \varepsilon$ is either

- $(0, t)$ where $t : x > -\varepsilon$
- $(1, t)$ where $t : x < \varepsilon$

Let x' that coincides to $x = x'$. Then, softsign $x\ \varepsilon =$ softsign $x'\ \varepsilon$. softsign is reduced to single-valued function.

Exact Real Type Theory: axioms

Axiom (multivalue functor)

- *Given a type* T, $\mathsf{mv}\,T$ *is a type.*
- *Given an element* $t : T$, *then* $\iota\, t : \mathsf{mv}\,T$
- *Given a function* $f : A \to \mathsf{mv}\,T$, *then* $\mathsf{lift}\ f : \mathsf{mv}\,A \to \mathsf{mv}\,B$ *such that*

  $f\ a = (\mathsf{lift}\ f)(\iota\ a)$ *for all* $a : A$.
- *There is isomorphism:* $\mathsf{iso} : \mathsf{mv}\,(\mathsf{mv}\,T) \to \mathsf{mv}\,T$ *such that* $a = \mathsf{iso}\,(\iota\ a)$ *for all* $a : \mathsf{mv}\,T$.

Exact Real Type Theory: axioms

Axiom (Sierpinski type)

- S *is a type such that* $\downarrow, \uparrow : \mathsf{S}$
- *Define* $s\downarrow := s = \downarrow : \mathsf{prop}$
- *There is an element:*

  $\mathsf{select} : \forall_{s_1,s_2:\mathsf{S}}\, s_1\downarrow \vee s_2\downarrow \to \mathsf{mv}\,(s_1\downarrow \vee s_2\downarrow)$

Definition

$\mathsf{semi}\ P := \exists_{s:\mathsf{S}}\, P \leftrightarrow s\downarrow$

Lemma

$\mathsf{choose} : \forall_{P,Q:\mathsf{prop}}\, \mathsf{semi}\, P \to \mathsf{semi}\, Q \to P\ \tilde{\vee}\ Q \to \mathsf{mv}\,(P \vee Q)$

Exact Real Type Theory: axioms

Axiom (Reals R)

- *Classical properties assumed as* **prop** *; e.g.,*
  $\mathsf{WTotality} : \forall_{x,y:\mathsf{R}}.\ x > y \,\tilde{\vee}\, x = y \,\tilde{\vee}\, y > x$
- *Semi decidability of order assumed:* $\forall_{x,y:\mathsf{R}}\ \mathsf{semi}\,(x > y)$
- *Construction via limit:*
  $\mathsf{limit} : \forall_{P:\mathsf{R}\to\mathsf{prop}}\ \tilde{\exists}!_{z:\mathsf{R}}\ P\ z \to (\forall_{n:\mathbb{N}}.\ \mathsf{mv}\,(\exists_{z:\mathsf{R}}\ \tilde{\exists}_{u:\mathsf{R}}\ P\ u \wedge |u - z| < 2^{-n})) \to \exists_{u:\mathsf{R}} P\ u$
- *Continuity of real functions:*
  $\mathsf{cont} : \forall_{f:\mathsf{R}\to\mathsf{R}} \forall_{x,y:\mathsf{R}} x < y \to \exists_{m,M:\mathsf{R}} \forall_{z:\mathsf{R}} x < z < y \to m < f\ z\ < M$

Lemma

for all $x, y, \varepsilon : \mathsf{R}$, *if* $\varepsilon > 0$ *we have* $\mathsf{mv}\,(x > y - \varepsilon \vee y > x - \varepsilon)$

Exact Real Type Theory: theories

Define $C(x, y, z) := x \geq y \to x = z \wedge y > x \to y = z$

Lemma (Maximum)

for all x, y *we have* $\exists_{z:\mathsf{R}} C\ x\ y\ z$

Lemma (Weak IVT)

for all $f : \mathsf{R} \to \mathsf{R}$, *if* $f\ 0 < 0 < f\ 1$, *we have* $\tilde{\exists}_{z:\mathsf{R}}\ 0 < z < 1 \wedge f\ z = 0$

Lemma (IVT)

for all $f : \mathsf{R} \to \mathsf{R}$, *if* $f\ 0 < 0 < f\ 1$ *and* $\tilde{\exists}!_{z:\mathsf{R}}\ f\ z = 0$ *we have* $\exists_{z:\mathsf{R}}\ 0 < z < 1 \wedge f\ z = 0$

Exact Real Lambda Calculus

Simply Typed Lambda Calculus:

1. Base Types: τ_0
2. Types: $\tau := \tau_0 \mid \tau_1 \to \tau_2 \mid \tau_1 \times \tau_2$
3. Terms: $t := x$ (variable) $\mid \lambda\, x : \tau.\, t \mid t_1\ t_2 \mid (t_1, t_2) \mid \mathsf{fst}\ \ t \mid \mathsf{snd}\ \ t$
4. Context: $\Gamma := (x_1 : \tau_1) :: (x_2 : \tau_2) :: \ldots :: (x_n : \tau_n)$
5. Type Judgement:
   1. $\Gamma \vdash x : \tau$ if and only if $(x, \tau) \in \Gamma$
   2. $\Gamma \vdash \lambda x : \tau_1 . t : \tau_1 \to \tau_2$ if and only of $(x, \tau) :: \Gamma \vdash t : \tau_2$
   3. $\Gamma \vdash t_1\ t_2 : \tau_2$ if and only of $\Gamma \vdash t_1 : \tau_1 \to \tau_2$ and $\Gamma \vdash t_2 : \tau_1$
   4. $\Gamma \vdash (t_1, t_2) : \tau_1 \times \tau_2$ if and only of $\Gamma \vdash t_1 : \tau_1$ and $\Gamma \vdash t_2 : \tau_2$
   5. $\Gamma \vdash \mathsf{fst}\ t : \tau_1$ if and only of $\Gamma \vdash t : \tau_1 \times \tau_2$
   6. $\Gamma \vdash \mathsf{snd}\ t : \tau_2$ if and only of $\Gamma \vdash t : \tau_1 \times \tau_2$
6. Reduction:
   1. $(\lambda\, x : \tau.\, t_1)\ t_2 \to t_1[x \mapsto t_2]$
   2. $\mathsf{fst}\ (t_1, t_2) \to t_1 \quad \mathsf{snd}\ (t_1, t_2) \to t_2$

Exact Real Lambda Calculus

Gödel's System T:

1. Base Types: $\tau_0 := \mathsf{nat}$
2. Types: $\tau := \tau_0 \mid \tau_1 \to \tau_2 \mid \tau_1 \times \tau_2$
3. Terms: $t := x \mid \lambda\, x : \tau.\, t \mid t_1\ t_2 \mid c$ (constants)
4. Constants: $c :=$

   $0 : \mathsf{nat}$

   $s : \mathsf{nat} \to \mathsf{nat}$

   $\mathsf{primerec}_\tau : \tau \to (\mathsf{nat} \to \tau \to \tau) \to \mathsf{nat} \to \tau)$
5. Reduction:

   $\mathsf{primerec}_\tau\ a\ f\ (s\ n) \to f\ n\ (\mathsf{primerec}_\tau\ a\ f\ n)$

   $\mathsf{primerec}_\tau\ a\ f\ 0 \to a$

$+_{\mathbb{N}}, *_{\mathbb{N}} : \mathsf{nat} \to \mathsf{nat} \to \mathsf{nat}$ and $>_{\mathbb{N}}, =_{\mathbb{N}}: \mathsf{nat} \to \mathsf{nat} \to \mathsf{bool}$ are defined by recursion.

Exact Real Lambda Calculus

Exact Real Lambda Calculus:

1. Base Types: $\tau_0 := \mathsf{real} \mid \mathsf{nat} \mid \mathsf{bool} \mid \mathsf{unit}$
2. Types: $\tau := \tau_0 \mid \tau_1 \to \tau_2 \mid \tau_1 \times \tau_2$
3. Terms: $t := x \mid \lambda\, x : \tau.\, t \mid t_1\ t_2 \mid c$ (constants)
4. Constants: $c :=$

$+_{\mathbb{R}}, *_{\mathbb{R}} : \mathsf{real} \to \mathsf{real} \to \mathsf{real}.\ \ -, / : \mathsf{real} \to \mathsf{real}.$
$>_{\mathbb{R}}: \mathsf{real} \to \mathsf{real} \to \mathsf{bool}. \quad //\ \text{infix}$
$\mathsf{inj}\ : \mathsf{nat} \to \mathsf{real}$
$\mathsf{select}\ : \mathsf{unit} \to \mathsf{unit} \to \mathsf{bool}$
$\mathsf{m}\ : (\mathsf{real} \to \mathsf{real}) \to \mathsf{real} \times \mathsf{real} \to \mathsf{real} \times \mathsf{real}$
$\downarrow: \tau \to \mathsf{unit} \quad //\ \text{postfix}$
$\mathsf{lim}\ : (\mathsf{nat} \to \mathsf{real}) \to \mathsf{real}$

Set-theoretic semantics

Meaning of a well typed ERLC term $a : A$.

- Semantics of types are:
  - $[\![\mathsf{real}]\!] := \mathcal{P}\mathbb{R}$ $[\![\mathsf{nat}]\!] := \mathcal{P}\mathbb{N}$ $[\![\mathsf{bool}]\!] := \mathcal{P}\mathbf{2}$ $[\![\mathsf{unit}]\!] := \mathcal{P}\{*\}$
  - $[\![A \to B]\!] := \mathcal{P}([\![A]\!] \to [\![B]\!])$ $[\![A \times B]\!] := \mathcal{P}([\![A]\!] \times [\![B]\!])$

  where $\mathcal{P}A :=$ finite nonempty subset of $A_\perp$.
- Semantics of terms:
  - $[\![\lambda\, x : \tau.\, t]\!] := \{y \mapsto [\![t]\!][x \mapsto y]\}$
  - $[\![t_1\ t_2]\!] := \cup_{f \in [\![t_1]\!]} \{f\ [\![t_2]\!]\}$

  It remains to define semantics of constants.

Set-theoretic semantics

$$[\![/_{\mathbb{R}}]\!]\ \{y_1, \cdots, y_m\} := \cup_i \text{if } y_i \neq 0 \text{ or } \bot \text{ then } \{1/y_i\} \text{ else } \{\bot\}$$

$$[\![\mathsf{inj}\ ]\!]\ \{n_1, \cdots, n_m\} = \{n_1, \cdots, n_m\} \subseteq \mathbb{R}$$

$$[\![\downarrow]\!]\{e_1, \cdots, e_n\} := \cup_i \text{if } e_i \neq \bot \text{ then } \{*\} \text{ else } \{\bot\}$$

$$[\![\mathsf{select}\ ]\!]\ \{v_1, \cdots, v_n\}\ \{w_1, \cdots, w_m\} := \cup_{i,j} \begin{cases} \{tt\} & \text{if } v_i = * \\ \{ff\} & \text{if } w_i = * \\ \{\bot\} & \text{if } v_i = w_j = * \end{cases}$$

$$\{a_1, \cdots, a_n\}[\![\downarrow]\!] := \cup_i \text{if } a_i \neq \bot \text{ then } \{*\} \text{ else } \{\bot\}$$

$$[\![\mathsf{lim}\ ]\!]\ f :=$$

$$\begin{cases} \{z | \forall n\ y.\ y \in f\ n \rightarrow |z - y| < 2^{-n}\} & \text{if such } z \text{ exists uniquely,} \\ \{\bot\} & \text{otherwise.} \end{cases}$$

Program Extraction

- Given $\Gamma_{\mathbb{R}} \vdash t : T$ in ERTT, extract computational content into a term $\langle t \rangle$ in ERLC.
- $\langle P : \mathsf{prop} \rangle := \mathsf{unit} \mid \langle p : P \rangle := *$
- $\langle T : \mathsf{type}_{i>0} \rangle := \mathbf{abort} \mid \langle t : T \rangle := \mathbf{abort}$
- $\langle T_1 \times T_2 : \mathsf{type}_0 \rangle := \langle T_1 \rangle \times \langle T_2 \rangle \mid \langle (t_1, t_2) \rangle := (\langle t_1 \rangle, \langle t_2 \rangle)$
- $\langle A \to B : \mathsf{type}_0 \rangle := \langle A \rangle \to \langle B \rangle \mid \langle \lambda x : A.\, t : A \to B \rangle := \lambda y : \langle A \rangle . \langle t \rangle [x \mapsto y]$
- $\langle t_1\ t_2 \rangle := \langle t_1 \rangle\ \langle t_2 \rangle$

Program Extraction

- Multivalued functor:
  - $\langle \mathsf{mv}\ T \rangle := \langle T \rangle$
  - $\langle \iota\ t \rangle := \langle t \rangle$
  - $\langle \mathsf{lift}\ f \rangle := \langle f \rangle$
  - $\langle \mathsf{iso}\ T \rangle := \lambda\ x : [\![T]\!].\ x$
- Sierpinski type:
  - $\langle \mathsf{S}\ \rangle := \mathsf{unit}$
  - $\langle \downarrow \rangle := *$
  - $\langle \mathsf{select}\ \rangle := \mathsf{select}$
- Real type:
  - $\langle \mathsf{R} \rangle := \mathsf{real}$
  - $\langle \mathsf{semi}\ (x > y) \rangle := \langle x \rangle > \langle y \rangle$
  - $\langle \mathsf{limit} \rangle := \mathsf{limit}$

$\forall\exists$-predicate

- $t : \forall_{x:\mathbb{R}} P(x) \to \mathsf{mv}\ (\exists_{y:\mathbb{R}} C(x,y))$
- Let P and C be in prop .
- Extraction:

$$
\begin{aligned}
&\langle \forall_{x:\mathsf{R}} P(x) \to \mathsf{mv}\ (\exists_{y:\mathbb{R}} C(x,y)) \rangle \\
&= \langle \mathsf{R} \rangle \to \langle P(x) \rangle \to \langle \mathsf{mv}\ (\exists_{y:\mathbb{R}} C(x,y)) \rangle \\
&= \mathsf{real} \to \mathsf{unit} \to \langle \exists_{y:\mathbb{R}} C(x,y) \rangle \\
&= \mathsf{real} \to \mathsf{unit} \to \mathsf{real} \times \mathsf{unit} \\
&\simeq \mathsf{real} \to \mathsf{real}
\end{aligned}
$$

Ongoing Work I: operational semantics for ERLC

- (A, δ_A) is represented space when A is a set with $\delta_A : \mathbb{B} \rightharpoonup A$ partial surjective.
- An infinite string β *realizes* a when $\delta_A\ \beta = a$.
- A computable (infinite) string function $\tau : \mathbb{B} \rightharpoonup \mathbb{B}$ realizes a set–valued function $A \to P(A)$ when
  $\forall a \in A.\ \forall \beta \in \delta_A^{-1}\ \{a\}.\ \delta_B\ (\tau\ \beta) \in f\ a$.
- Continuous string function $\mathbb{B} \rightharpoonup \mathbb{B}$ can be embedded in $\mathbb{B}$.
- Assigning represented space to each type in ERLC and string function to each term in ERLC gives computable interpretation (program).
- Show that all ERLC's set-theoretic semantics are realizable.

Ongoing Work II: set-theoretic semantic for ERTT

- Given a proposition ERTT P, define its set-theoretical semantics $\langle P \rangle_{\mathsf{prop}}$ such that $\langle P(x) \rangle \subseteq \mathbb{R}$.
- For any $\forall\exists$-predicate $t : \forall_{x:\mathbb{R}} P(x) \to \mathsf{mv}\ (\exists_{y:\mathbb{R}} C(x,y))$ prove $\forall_{x \in \mathbb{R}} \langle P \rangle_{\mathsf{prop}}\ x \to \bot \notin [\![\langle t \rangle]\!]\ \{x\} \wedge [\![\langle t \rangle]\!]\ \{x\} \subseteq \langle Q \rangle_{\mathsf{prop}}\ x$
- Hence the program extraction is sound

Ongoing and Future Work III

- Implementation of both ERTT and ERLC
- Define Categorical semantics for both ERTT and ERLC