Logic for exact real arithmetic

Helmut Schwichtenberg
(j.w.w. Franziskus Wiesnet)

Mathematisches Institut, LMU, München

Dedicated to Vladimir P. Orevkov
on occasion of his 80th birthday

Proofs have two aspects:

1. they guarantee correctness, and
2. they may have computational content.

We address (2), and use a BHK-interpretation to extract programs from proofs. Features:

- The extract is a term in the underlying theory, hence we have a framework to formally prove its properties.
- Computational content in (co)inductive predicates only.
- From proofs in constructive analysis[1] we can extract programs operating on stream-represented real numbers.

[1] E. Bishop, Foundations of Constructive Analysis, 1967

Minimal logic, natural deduction

- Introduction and elimination rules for $\to$, $\forall$.
- Introduction and elimination axioms for (co)inductive predicates (e.g. $\exists$, $\vee$, $\wedge$).
- Proof terms with formulas as types, $\sim$ λ-terms with constants.
- Normalization is essential (eliminate use of lemmas, evaluate realizers).

Efficiency of normalization

- Needed to simplify terms in formulas (in interactive proofs with a proof assistant).
- Needed to evaluate realizing terms extracted from proofs.
- Superexponential for typed λ-terms[2].
- Analysis of efficiency for λ-terms with constants beautyfully done by Vladimir Orevkov[3].

[2] R. Statman, The typed λ–calculus is not elementary recursive, TCS 1979
[3] V. Orevkov, Lower bounds for increasing complexity of derivations after cut elimination, Zapiski 1979

Infinite data of base type

Consider the base type $\mathbb{L}$ of lists of signed digits $\bar{1}, 0, 1$. $\mathbb{L}$-objects can be total, cototal or partial (strict inclusions).

- A total object: $1 :: 0 :: 1 :: 0 :: []$
- A cototal object: $1 :: 0 :: 1 :: 0 :: 1 :: 0 :: \dots$

A partial object is the "deductive closure" of a finite "consistent" set of "tokens". For example, $1 :: * :: 1 :: *$ is a token, asserting that the 0th and 2nd element is 1.

Corecursion

${}^{\mathrm{co}}\mathcal{R}_{\mathbb{N}}^{\tau}$ of type $\tau \to (\tau \to \mathbb{U} + (\mathbb{N} + \tau)) \to \mathbb{N}$ is defined by

$$ {}^{\mathrm{co}}\mathcal{R}_{\mathbb{N}}^{\tau} x f = \begin{cases} 0 & \text{if } fx \equiv \mathrm{DummyL}^{\mathbb{U}+(\mathbb{N}+\tau)} \\ Sn & \text{if } fx \equiv \mathrm{Inr}(\mathrm{InL}^{\mathbb{N}\to\mathbb{N}+\tau} n) \\ S({}^{\mathrm{co}}\mathcal{R}_{\mathbb{N}}^{\tau} x' f) & \text{if } fx \equiv \mathrm{Inr}(\mathrm{InR}^{\tau\to\mathbb{N}+\tau} x'). \end{cases} $$

As a rule this is non-terminating, but still the constant ${}^{\mathrm{co}}\mathcal{R}_{\mathbb{N}}^{\tau}$ denotes a (partial) object in our model.

Formal neighborhoods

We use information systems[4] to represent the objects of our model.
Types are built from base types ι (free algebras) by $\tau \to \sigma$.

- Formal neighborhoods U are finite "consistent" sets of tokens.
- (U, a) is a token of type $\tau \to \sigma$.
- $\{(U_1, a_1), \dots, (U_n, a_n)\}$: formal neighborhood of type $\tau \to \sigma$.

Application of $\{(U_1, a_1), \dots, (U_n, a_n)\}$ to U:

$$\{\, a_i \mid U \vdash U_i \,\} \quad \text{where } \vdash \text{ means "entails".}$$

[4] K. Larsen and G. Winskel, Using information systems to solve recursive domain equations effectively, 1984

Computability and continuity

Partial continuous functional[5]: consistent "deductively closed" (possibly infinite) set of tokens. f is computable if this set is recursively enumerable. Continuity:

- Let f, x be infinite objects of types $\tau \to \sigma$, τ
- Let V be an approximation of $f(x)$.

Then we can find approximations W of f and U of x such that

- $W(U)$ approximates $f(x)$, and
- $W(U) \vdash V$.

[5] D. Scott, Outline of a mathematical theory of computation, 1970, and Y. Ershov, Model C of partial continuous functionals, 1984

We inductively define a predicate I_0 on reals by the clauses

$$\forall_x(x = 0 \to x \in I_0), \quad \forall_{d \in \mathrm{Sd}} \forall_x \forall_{x' \in I_0} \Big(x = \frac{x' + d}{2} \to x \in I_0 \Big).$$

Then the induction (or least-fixed-point) axiom is

$$\forall_x(x{=}0 \to x \in P) \to \forall_{d \in \mathrm{Sd}} \forall_x \forall_{x' \in I_0 \cap P} \Big(x{=}\frac{x' + d}{2} \to x \in P \Big) \to I_0 \subseteq P.$$

Then ${}^{\mathrm{co}}I_0$ is given by the closure axiom

$$\forall_{x \in {}^{\mathrm{co}}I_0} \Big(x = 0 \vee \exists_{d \in \mathrm{Sd}} \exists_{x' \in {}^{\mathrm{co}}I_0} \Big(x = \frac{x' + d}{2} \Big) \Big)$$

and the coinduction (or greatest-fixed-point) axiom is

$$\forall_{x \in P} \Big(x = 0 \vee \exists_{d \in \mathrm{Sd}} \exists_{x' \in {}^{\mathrm{co}}I_0 \cup P} \Big(x = \frac{x' + d}{2} \Big) \Big) \to P \subseteq {}^{\mathrm{co}}I_0.$$

- Both I_0 and ${}^{\mathrm{co}}I_0$ are declared as "computationally relevant".
- The associated algebra is $\mathbb{L}$ (lists of signed digits).
- The first constructor $[]\colon \mathbb{L}$ is a witness for the first clause, and the second :: of type $\mathbb{D} \to \mathbb{L} \to \mathbb{L}$ a witness for the second.

Computational content of the axioms:

- Clauses: constructors
- Induction axiom: recursion operator $\mathcal{R}_{\mathbb{L}}^{\tau}$
- Closure axiom: destructor $\mathcal{D}_{\mathbb{L}}$
- Coinduction axiom: corecursion operator ${}^{\mathrm{co}}\mathcal{R}_{\mathbb{L}}^{\tau}$

Since 0 as real number is represented by the stream of 0's, we can simplify I_0 by removing the nullary clause, and obtain I and ${}^{\mathrm{co}}I$. We only need ${}^{\mathrm{co}}I$, coinductively defined by the closure axiom

$$\forall_{x \in {}^{\mathrm{co}}I} \exists_{d \in \mathrm{Sd}} \exists_{x' \in {}^{\mathrm{co}}I} \left(x = \frac{x' + d}{2} \right).$$

Therefore, the coinduction axiom is

$$\forall_{x \in P} \exists_{d \in \mathrm{Sd}} \exists_{x' \in {}^{\mathrm{co}}I \cup P} \left(x = \frac{x' + d}{2} \right) \to P \subseteq {}^{\mathrm{co}}I.$$

The associated data type is the algebra $\mathbb{S}$ (of streams of signed digits) given by a single binary constructor of type $\mathbb{D} \to \mathbb{S} \to \mathbb{S}$.

Computational content of the axioms:

- Closure axiom: destructor $\mathcal{D}_{\mathbb{S}}$ of type $\mathbb{S} \to \mathbb{D} \times \mathbb{S}$, defined by

$$\mathcal{D}_{\mathbb{S}}(d :: u) = \langle d, u \rangle.$$

- Coinduction axiom: corecursion operator ${}^{\mathrm{co}}\mathcal{R}^{\tau}_{\mathbb{S}}$ of type $\tau \to (\tau \to \mathbb{D} \times (\mathbb{S} + \tau)) \to \mathbb{S}$:

$${}^{\mathrm{co}}\mathcal{R}^{\tau}_{\mathbb{S}} x f = \begin{cases} d :: u & \text{if } fx = \langle d, \mathrm{InL}^{\mathbb{S} \to \mathbb{S}+\tau} u \rangle \\ d :: {}^{\mathrm{co}}\mathcal{R}^{\tau}_{\mathbb{S}} x' f & \text{if } fx = \langle d, \mathrm{InR}^{\tau \to \mathbb{S}+\tau} x' \rangle. \end{cases}$$

Soundness theorem

Let M be an $\mathbf{r}$-free derivation of a formula A from assumptions $u_i \colon C_i$ ($i < n$). Then we can derive

$$\begin{cases} \mathrm{et}(M) \ \mathbf{r} \ A & \text{if } A \text{ is c.r.} \\ A & \text{if } A \text{ is n.c.} \end{cases}$$

from assumptions

$$\begin{cases} z_{u_i} \ \mathbf{r} \ C_i & \text{if } C_i \text{ is c.r.} \\ C_i & \text{if } C_i \text{ is n.c.} \end{cases}$$

The proof needs invariance axioms:

- Constructively to state A means[6] the same as to say that A has a realizer.
- This statement $A \leftrightarrow \exists_x (x \mathbf{r} A)$ was called "to assert is to realize" by Feferman[7].
- For **r**-free c.r. formulas A we require the invariance axioms

$$\forall_z (z \mathbf{r} A \to A).$$
$$A \to \exists_z (z \mathbf{r} A).$$

[6] A.N. Kolmogorov, Zur Deutung der intuitionistischen Logik, Math. Zeitschr., 1932

[7] S. Feferman, Constructive theories of functions and classes, 1979

Proof of the soundness theorem

We only consider the cases using invariance axioms.

Case $(\lambda_{u^A} M^B)^{A \to B}$ with B n.c. and A c.r. We need a derivation of $A \to B$. By IH we have a derivation of B from $z\ \mathbf{r}\ A$. Required derivation of B from A:

$$\dfrac{\dfrac{A \to \exists_z (z\ \mathbf{r}\ A) \qquad A}{\exists_z (z\ \mathbf{r}\ A)} \qquad \begin{matrix}[z\ \mathbf{r}\ A] \\ \mid \mathrm{IH} \\ B\end{matrix}}{B}\ \exists^-$$

Case $(M^{A \to B} N^A)^B$ with B n.c. and A c.r. We need a derivation of B. By IH we have derivations of $A \to B$ and of $\mathrm{et}(N)\ \mathbf{r}\ A$. We obtain the required derivation from

$$\dfrac{\dfrac{\forall_z (z\ \mathbf{r}\ A \to A) \qquad \mathrm{et}(N)}{\mathrm{et}(N)\ \mathbf{r}\ A \to A} \qquad \begin{matrix}\mid \mathrm{IH} \\ \mathrm{et}(N)\ \mathbf{r}\ A\end{matrix}}{A}$$

and the derivation of $A \to B$.

Extracted term $\mathrm{et}(M)$ of a derivation M^A with A c.r.

$$
\begin{aligned}
\mathrm{et}(u^A) &:= z_u^{\tau(A)} \quad (z_u^{\tau(A)} \text{ uniquely associated to } u^A),\\
\mathrm{et}((\lambda_{u^A} M^B)^{A\to B}) &:= \begin{cases} \lambda_{z_u}^{\tau(A)}\mathrm{et}(M) & \text{if } A \text{ is c.r.}\\ \mathrm{et}(M) & \text{if } A \text{ is n.c.}\end{cases}\\
\mathrm{et}((M^{A\to B} N^A)^B) &:= \begin{cases} \mathrm{et}(M)\mathrm{et}(N) & \text{if } A \text{ is c.r.}\\ \mathrm{et}(M) & \text{if } A \text{ is n.c.}\end{cases}\\
\mathrm{et}((\lambda_x M^A)^{\forall_x A}) &:= \mathrm{et}(M),\\
\mathrm{et}((M^{\forall_x A(x)} t)^{A(t)}) &:= \mathrm{et}(M).
\end{aligned}
$$

Consider a c.r. inductively defined predicate. The extracted terms for its axioms are:

- Clauses: constructors
- Induction axiom: recursion operator $\mathcal{R}^{\tau}$
- Closure axiom: destructor $\mathcal{D}$
- Coinduction axiom: corecursion operator ${}^{\mathrm{co}}\mathcal{R}^{\tau}$

For the induction axiom $(I^{\mathrm{nc}})^{-}$ of a "one-clause-nc" inductive predicate with a c.r. competitor predicate the extracted term is the identity.

Realizers

Example. I_0.

- By another inductive predicate $I_0^{\mathbf{r}}$ of arity $(\mathbb{R}, \mathbb{L})$ we can express that a list u witnesses ("realizes") that the real x is in I_0.
- We write $u \mathbf{r} I_0 x$ (u is a realizer of $x \in I_0$) for $(x, u) \in I_0^{\mathbf{r}}$.
- The predicate $I_0^{\mathbf{r}}$ is n.c. (since we already have a realizer u).
- $I_0^{\mathbf{r}}$ is inductively defined by the two clauses

$$(0, []) \in I_0^{\mathbf{r}}, \quad \forall_{d \in \mathrm{Sd}} \forall_{(x,u) \in I_0^{\mathbf{r}}} \left(\left(\frac{x+d}{2}, s_d :: u \right) \in I_0^{\mathbf{r}} \right)$$

  and the induction axiom

$$(0, []) \in Q \to \forall_{d \in \mathrm{Sd}} \forall_{(x,u) \in I_0^{\mathbf{r}} \cap Q} \left(\left(\frac{x+d}{2}, s_d :: u \right) \in Q \right) \to I_0^{\mathbf{r}} \subseteq Q.$$

  s_d is the signed digit corresponding to the formula $d \in \mathrm{Sd}$.
- Similarly we coinductively define the n.c. predicate $({}^{\mathrm{co}}I_0)^{\mathbf{r}}$.

Application: division of reals in $[-1,1]$

Idea[8]: three representations of $\frac{x}{y}$:

$$\frac{x}{y} = \frac{1+\frac{x_1}{y}}{2} = \frac{0+\frac{x_0}{y}}{2} = \frac{-1+\frac{x_{-1}}{y}}{2}$$

where

$$x_1 = 4\frac{x+\frac{-y}{2}}{2}, \quad x_0 = 2x, \quad x_{-1} = 4\frac{x+\frac{y}{2}}{2}.$$

- Depending on x choose one of these representations.
- This gives the first digit.
- Result: corecursive definition of $\frac{x}{y}$.

[8] A. Ciaffaglione and P.D. Gianantonio, A certified, corecursive implementation of exact real numbers. TCS 2006

Define ${}^{\mathrm{co}}I$ coinductively by the closure axiom

$$\forall_{x \in {}^{\mathrm{co}}I} \exists_{d \in \mathrm{Sd}} \exists_{x' \in {}^{\mathrm{co}}I} \left(x = \frac{x' + d}{2} \right).$$

Theorem (CoIDiv)

For x, y in ${}^{\mathrm{co}}I$ with $\frac{1}{4} \leq y$ and $|x| \leq y$ we have $\frac{x}{y}$ in ${}^{\mathrm{co}}I$.

Proof by coinduction. Computational content:

$$\mathrm{Div}(u, v) :=$$

$$\begin{cases} \mathrm{SdR} :: \mathrm{Div}(\mathrm{AuxR}(u, v), v) & \text{if } u = 1\tilde{u} \vee u = 01\tilde{u} \vee u = 001\tilde{u}, \\ \mathrm{SdM} :: \mathrm{Div}(\mathrm{Double}(u), v) & \text{if } u = 000\tilde{u}, \\ \mathrm{SdL} :: \mathrm{Div}(\mathrm{AuxL}(u, v), v) & \text{if } u = \bar{1}\tilde{u} \vee u = 0\bar{1}\tilde{u} \vee u = 00\bar{1}\tilde{u}. \end{cases}$$

Look-ahead: 3 digits.

Lemma

${}^{\mathrm{co}}I$ is closed under shifting a real $x \leq 0$ ($x \geq 0$) by $+1$ (-1).

Computational content:

$$
\begin{aligned}
\mathrm{add1}(\mathrm{SdR}{::}u) &:= [\mathrm{SdR}, \mathrm{SdR}, \dots], & \mathrm{sub1}(\mathrm{SdR}{::}u) &:= \mathrm{SdL}{::}u, \\
\mathrm{add1}(\mathrm{SdM}{::}u) &:= \mathrm{SdR}{::}\mathrm{add1}(u), & \mathrm{sub1}(\mathrm{SdM}{::}u) &:= \mathrm{SdL}{::}\mathrm{sub1}(u), \\
\mathrm{add1}(\mathrm{SdL}{::}u) &:= \mathrm{SdR}{::}u & \mathrm{sub1}(\mathrm{SdL}{::}u) &:= [\mathrm{SdL}, \mathrm{SdL}, \dots].
\end{aligned}
$$

Extracted term of the $+1$ part:

```
[u](CoRec ai=>ai)u
 ([u0][case (DesYprod u0)
     (s pair u1 -> [case s
       (SdR -> SdR pair InL cCoIOne)
       (SdM -> SdR pair InR u1)
       (SdL -> SdR pair InL u1)])])
```

Translation into Haskell

Recall

$$\mathrm{Div}(u,v) :=$$

$$\begin{cases} \mathrm{SdR} :: \mathrm{Div}(\mathrm{AuxR}(u,v),v) & \text{if } u = 1\tilde{u} \vee u = 01\tilde{u} \vee u = 001\tilde{u}, \\ \mathrm{SdM} :: \mathrm{Div}(\mathrm{Double}(u),v) & \text{if } u = 000\tilde{u}, \\ \mathrm{SdL} :: \mathrm{Div}(\mathrm{AuxL}(u,v),v) & \text{if } u = \bar{1}\tilde{u} \vee u = 0\bar{1}\tilde{u} \vee u = 00\bar{1}\tilde{u}. \end{cases}$$

Tests (in `ghci` with time measuring by `:set +s`). Return the first n digits of the result of dividing $\frac{1001}{3001}$ by $\frac{10001}{20001}$

number of digits	runtime in seconds
10	0.01
25	0.05
50	0.14
75	0.26
100	0.46

Formal soundness proof

```
(add-sound "CoIDiv")
;; ok, CoIDivSound has been added as a new theorem:

;; allnc x,y,u^(
;;  CoIMR x u^ ->
;;  allnc u^0(
;;   CoIMR y u^0 ->
;;   (1#4)<<=y -> abs x<<=y ->
;;   CoIMR(x*RealUDiv y 3)(cCoIDiv u^ u^0)))

;; with computation rule

;; cCoIDiv eqd([u,u0]cCoIDivAux u0 u)
```

The generated formal soundness proof can be machine checked.

Conclusion

- TCF as a variant of HA^ω. Differences
  - based on a model (Shoenfield: "classical axiom system")
  - partial continuous functionals, contain corecursion operators
  - inductive and coinductive predicates.
- Realizability, invariance axioms, formal soundness proof.
- Application[9]: division algorithm for stream represented reals extracted from a formalized proof (in Minlog[10]) on ordinary reals.

[9] H.S. and F. Wiesnet, LMCS 17, April 2021

[10] `http://minlog-system.de`, file `examples/analysis/sddiv.scm`