Program extraction in constructive analysis

Helmut Schwichtenberg

Mathematisches Institut, LMU, München

Tsukuba University, 19. June 2009

Algebras and function spaces

- Parametrized free algebras. Examples: Binary numbers (constructors 1, S_0, S_1), lists.
- "Lazy" base types; function spaces via limits of finite approximations (Scott's information systems).
- Computable functionals are recursively enumerable limits.
- Variables range over the Scott-Ershov partial continuous functionals.
- Constructors are injective and have disjoint ranges.

Computable functionals

can be conveniently defined by "computation rules" (a form of pattern matching). Examples:

$$\begin{cases} \mathcal{R}(0, r, s) =_\tau r, \\ \mathcal{R}(\mathrm{S}n, r, s) =_\tau s(n, \mathcal{R}(n, r, s)) \end{cases}$$

or the fixed point operator

$$Y_\tau w^{\tau \to \tau} =_\tau w(Y_\tau w).$$

Denotational and operational semantics

- Define terms from (simply) typed variables and constants by (lambda) abstraction and application.
- The approach via information systems allows a direct definition of denotational semantics.
- Operational semantics (β-conversion plus computation rules) is "adequate": closed terms denoting "total" objects evaluate to numerals.

Minimal logic

- The only (basic) logical connectives are $\to$, $\forall$.
- Proofs have two aspects:
  (i) They guarantee correctness.
  (ii) They may have computational content.
- Computational content only enters a proof via inductively (or coinductively) defined predicates.

Natural deduction: assumption variables u^A. Rules for $\to$:

derivation	proof term
$$\dfrac{\begin{array}{c}[u\colon A] \\ \mid M \\ B\end{array}}{A \to B}\to^+ u$$	$(\lambda_{u^A} M^B)^{A\to B}$
$$\dfrac{\begin{array}{c}\mid M \\ A \to B\end{array} \qquad \begin{array}{c}\mid N \\ A\end{array}}{B}\to^-$$	$(M^{A\to B} N^A)^B$

Natural deduction: rules for $\forall$

derivation	proof term
$$\dfrac{\begin{array}{c} \mid M \\ A \end{array}}{\forall_x A}\,\forall^+ x \quad \text{(var. cond.)}$$	$(\lambda_x M^A)^{\forall_x A}$ (var. cond.)
$$\dfrac{\begin{array}{c} \mid M \\ \forall_x A(x) \end{array} \qquad r}{A(r)}\,\forall^-$$	$(M^{\forall_x A(x)} r)^{A(r)}$

Inductive definitions

- Example: Totality, defined by the clauses

$$T0, \qquad \forall_n (Tn \to T(\mathrm{S}n)).$$

- Elimination (or least fixed point) scheme

$$\forall_n (Tn \to A(0) \to \forall_n (Tn \to A(n) \to A(\mathrm{S}n)) \to A(n)),$$

i.e., the induction scheme for (total) natural numbers.

Example: Leibniz equality

- is defined by the clause $\forall_x \mathrm{Eq}_\rho(x^\rho, x^\rho)$. Elimination scheme:

$$\forall_{x,y}(\mathrm{Eq}(x,y) \to \forall_x C(x,x) \to C(x,y)).$$

- With $C(x,y) := A(x) \to A(y)$ this implies

$$\forall_{x,y}(\mathrm{Eq}(x,y) \to A(x) \to A(y)) \qquad \text{(compatibility of } \mathrm{Eq}).$$

  Hence symmetry and transitivity of Eq.

Equalities

Notice that we have at least three different equalities:

- Leibniz equality Eq.
- Decidable equality $=_{\mathbb{N}} : \mathbb{N} \to \mathbb{N} \to \mathbb{B}$. The boolean term $n =_{\mathbb{N}} m$ is turned into a formula by writing

$$\mathrm{Eq}_{\mathbb{B}}(n =_{\mathbb{N}} m, \mathtt{tt}).$$

- Equality of reals: a defined equivalence relation.

Example: $\exists$

- $\exists_x A$ is a nullary inductively defined predicate, with parameter $\{ x \mid A \}$.
- Clause:
$$\forall_x (A \to \exists_x A).$$
- Elimination scheme:
$$\exists_x A \to \forall_x (A \to B) \to B \qquad (x \text{ not free in } B).$$
- Similarly for $\wedge$, $\vee$.

Ex-Falso-Quodlibet

need not be assumed, but can be proved.

$$\mathbf{F} \to A, \text{ with } \mathbf{F} := \mathrm{Eq}(\mathrm{ff}, \mathrm{tt}) \text{ ("falsity")}.$$

The proof is in 2 steps. (i) $\mathbf{F} \to \mathrm{Eq}(x^\rho, y^\rho)$, since from $\mathrm{Eq}(\mathrm{ff}, \mathrm{tt})$ by compatibility

$$\mathrm{Eq}\,\underbrace{[\textbf{if } \mathrm{tt} \textbf{ then } x \textbf{ else } y]}_{x}\,\underbrace{[\textbf{if } \mathrm{ff} \textbf{ then } x \textbf{ else } y]}_{y}.$$

(ii) Induction on (the sim. definition of) predicates and formulas.

- Case Is. Let K_0 be the nullary clause $A_1 \to \cdots \to A_n \to It$. By IH: $\mathbf{F} \to A_i$. Hence It. From $\mathbf{F}$ we obtain $\mathrm{Eq}(s, t)$, by (i). Hence Is by compatibility.
- The cases $A \to B$, $\forall_x A$ are easy.

Embedding classical arithmetic

- Let $\neg A := (A \to \mathbf{F})$, and

$$\tilde{\exists}_x A := \neg \forall_x \neg A, \qquad A \mathbin{\tilde{\vee}} B := (\neg A \to \neg B \to \mathbf{F}).$$

- Consider a total boolean term $r^{\mathbf{B}}$ as representing a decidable predicate: $\mathrm{Eq}(r, \mathtt{tt})$.
- Prove $\forall_{p \in T}(\neg\neg \mathrm{Eq}(p, \mathtt{tt}) \to \mathrm{Eq}(p, \mathtt{tt}))$ by boolean induction.
- Lift this via $\to, \forall$ using

$$\begin{aligned} &\vdash (\neg\neg B \to B) \to \neg\neg(A \to B) \to A \to B, \\ &\vdash (\neg\neg A \to A) \to \neg\neg \forall_x A \to \forall_x A. \end{aligned}$$

- For formulas A built from $\mathrm{Eq}(\cdot, \mathtt{tt})$ by $\to, \forall_{x \in T}$ prove stability

$$\forall_{\vec{x} \in T}(\neg\neg A \to A) \qquad (\mathrm{FV}(A) \text{ among } \vec{x}).$$

Reals

A real number x is a pair $((a_n)_{n \in \mathbb{N}}, \alpha)$ with $a_n \in \mathbb{Q}$ and $\alpha \colon \mathbb{N} \to \mathbb{N}$ such that $(a_n)_n$ is a Cauchy sequence with modulus α, that is

$$\forall_{k,n,m}(\alpha(k) \leq n, m \to |a_n - a_m| \leq 2^{-k}),$$

and α is weakly increasing.

Two reals $x := ((a_n)_n, \alpha)$, $y := ((b_n)_n, \beta)$ are equivalent (written $x = y$), if

$$\forall_k(|a_{\alpha(k+1)} - b_{\beta(k+1)}| \leq 2^{-k}).$$

Nonnegative and positive reals

A real $x := ((a_n)_n, \alpha)$ is nonnegative (written $x \in \mathbb{R}^{0+}$) if

$$\forall_k (-2^{-k} \leq a_{\alpha(k)}).$$

It is k-positive (written $x \in_k \mathbb{R}^+$) if

$$2^{-k} \leq a_{\alpha(k+1)}.$$

$x \in \mathbb{R}^{0+}$ and $x \in_k \mathbb{R}^+$ are compatible with equivalence.

Can define $x \mapsto k_x$ such that $a_n \leq 2^{k_x}$ for all n.
However, $x \mapsto k_x$ is not compatible with equivalence.

Arithmetical functions

Given $x := ((a_n)_n, \alpha)$ and $y := ((b_n)_n, \beta)$, define

z	c_n	$\gamma(k)$
$x + y$	$a_n + b_n$	$\max(\alpha(k+1), \beta(k+1))$
$-x$	$-a_n$	$\alpha(k)$
$\lvert x\rvert$	$\lvert a_n\rvert$	$\alpha(k)$
$x \cdot y$	$a_n \cdot b_n$	$\max(\alpha(k+1+k_{\lvert y\rvert}), \beta(k+1+k_{\lvert x\rvert}))$
$\frac{1}{x}$ for $\lvert x\rvert \in_l \mathbb{R}^+$	$\begin{cases} \frac{1}{a_n} & \text{if } a_n \neq 0 \\ 0 & \text{if } a_n = 0 \end{cases}$	$\alpha(2(l+1)+k)$

Comparison of reals

Write $x \leq y$ for $y - x \in \mathbb{R}^{0+}$ and $x < y$ for $y - x \in \mathbb{R}^{+}$.

$$x \leq y \leftrightarrow \forall_k \exists_p \forall_{n \geq p} (a_n \leq b_n + 2^{-k}),$$

$$x < y \leftrightarrow \exists_{k,q} \forall_{n \geq q} (a_n + 2^{-k} \leq b_n).$$

Write $x <_{k,q} y$ (or simply $x <_k y$ if q is not needed) when we want to call these witnesses. Notice:

$$x \leq y \leftrightarrow y \not< x.$$

Continuous functions

A continuous function $f: I \to \mathbb{R}$ on a compact interval I with rational end points is given by

- an approximating map $h_f: (I \cap \mathbb{Q}) \times \mathbb{N} \to \mathbb{Q}$,
- a (uniform) modulus map $\alpha_f: \mathbb{N} \to \mathbb{N}$ such that $(h_f(c,n))_n$ is a real with modulus α_f, and
- a (uniform) modulus of continuity $\omega_f: \mathbb{N} \to \mathbb{N}$ satisfying

$$|a-b| \le 2^{-\omega_f(k)+1} \to |h_f(a,n) - h_f(b,n)| \le 2^{-k}$$

for $n \ge \alpha_f(k)$. α_f, ω_f required to be weakly increasing.

Notice: h_f, α_f, ω_f are of type level 1 only.

Application of a continuous function to a real

Given a continuous function f (by h_f, α_f, ω_f) and a real $x := ((a_n)_n, \alpha)$, application $f(x)$ is defined to be

$$(h_f(a_n, n))_n$$

with modulus $k \mapsto \max(\alpha_f(k+2), \alpha(\omega_f(k+1)-1))$.

One proves easily

$$x = y \to f(x) = f(y),$$
$$|x - y| \leq 2^{-\omega_f(k)} \to |f(x) - f(y)| \leq 2^{-k}.$$

Intermediate value theorem

Let $a < b$ be rationals. If $f : [a, b] \to \mathbb{R}$ is continuous with $f(a) \leq 0 \leq f(b)$, and with a uniform lower bound on its slope, then we can find $x \in [a, b]$ such that $f(x) = 0$.

Proof sketch.

1. Approximate Splitting Principle. Let x, y, z be given with $x < y$. Then $z \leq y$ or $x \leq z$.
2. IVTAux. Assume $a \leq c < d \leq b$, say $2^{-n} < d - c$, and $f(c) \leq 0 \leq f(d)$. Construct c_1, d_1 with $d_1 - c_1 = \frac{2}{3}(d - c)$, such that $a \leq c \leq c_1 < d_1 \leq d \leq b$ and $f(c_1) \leq 0 \leq f(d_1)$.
3. IVTcds. Iterate the step $c, d \mapsto c_1, d_1$ in IVTAux.

Let $x = (c_n)_n$ and $y = (d_n)_n$ with the obvious modulus. As f is continuous, $f(x) = 0 = f(y)$ for the real number $x = y$. $\square$

Inverse functions

Theorem

Let $f\colon [a,b] \to \mathbb{R}$ be continuous with a uniform lower bound on its slope. Let $f(a) \le a' < b' \le f(b)$. We can find a continuous $g\colon [a',b'] \to \mathbb{R}$ such that $f(g(y)) = y$ for every $y \in [a',b']$ and $g(f(x)) = x$ for every $x \in [a,b]$ such that $a' \le f(x) \le b'$.

Proof sketch.

Let $f(a) \le a' < b' \le f(b)$. Construct a continuous $g\colon [a',b'] \to \mathbb{R}$ by the Intermediate Value Theorem. $\square$

Example: squaring $f: [1,2] \to [1,4]$

Given by

- the approximating map $h_f(a,n) := a^2$,
- the uniform Cauchy modulus $\alpha_f(k) := 0$, and
- the modulus $k \mapsto k+3$ of uniform continuity.

A lower bound on its slope is $l := -1$, because for all $c, d \in [1,2]$

$$2^{-k} \leq d - c \to c^2 <_{k-1} d^2.$$

Then $h_g(u,n) := c_n^{(u)}$, as constructed in the IVT for $x^2 - u$, iterating IVTAux. The Cauchy modulus α_g is such that $(2/3)^n \leq 2^{-k+3}$ for $n \geq \alpha_g(k)$, and the modulus of uniform continuity is $\omega_g(k) := k+2$.

Formalization, program extraction

Many details. Important: representation of data. Here: direct approach, by explicitely building the required number systems (natural numbers in binary, rationals, reals as Cauchy sequences of rationals with a modulus, continuous functions in the sense of the type-1 representation described above, etc.)

Method of program extraction based on modified realizability (Kleene, Kreisel, Troelstra).

Results of demo

- Given: formalized proof of `"InvApprox"`.
- `inv-approx-eterm` defined, after animating the theorems.
- Squaring function `sq` defined on $[1, 2]$ by `ContConstr`.
- Term `inv-sq-approx` defined as `inv-approx-eterm` applied to `sq` and some bounds.
- `inv-sq-approx` applied to 3 (argument, to be inverted) and 20 (error bound: number of binary digits) normalized.

Related work

Russell O'Connor (PhD Thesis, Nijmegen 2009) builds on Coq; he uses a slightly different version of $\mathbb{R}$. Here:

- No need for dependent types, universes, "strength".
- Minimal logic for $\to, \forall$ plus inductive definitions suffice.
- But: partial functionals need to be first class citizens.

References

- E. Bishop. Foundations of Constructive Analysis. McGraw-Hill, 1967.
- H.S., Realizability interpretation of proofs in constructive analysis. Theory of Computing Systems, 2008.
- R. O'Connor, Incompleteness & Completeness. Formalizing Logic and Analysis in Type Theory. PhD Thesis, Nijmegen 2009.