Density formalized

Helmut Schwichtenberg (j.w.w. Basil Karádais, Iosif Petrakis)

Mathematisches Institut, LMU, München

Formalized Mathematics, Padova, 9. January 2013

Foundation of mathematics for computer-aided formalization

Desired features of such a foundation:

- minimalist
- two-level

points, ideals, abstract objects

$\uparrow\downarrow$

finite approximations

- accomodate constructive arguments, i.e., not restrict to the negative fragment.

- To accomodate constructive aspects use both $\exists_x A$ (strong $\exists$) and

$$\tilde{\exists}_x A \quad (\text{weak } \exists), \text{ defined by } \neg\forall_x\neg A \text{ (with } \neg A := A \to \bot).$$

- Similarly: $A \vee B$ (strong $\vee$) and

$$A \mathbin{\tilde{\vee}} B \quad (\text{weak } \vee), \text{ defined by } (A \to \bot) \to (B \to \bot) \to \bot.$$

- Classical logic then is a fragment, and we have

$$\vdash \exists_x A \to \tilde{\exists}_x A, \qquad \vdash A \vee B \to A \mathbin{\tilde{\vee}} B,$$

but not conversely; this is why $\tilde{\exists}, \tilde{\vee}$ are called "weak".

Kolmogorov 1932: "Zur Deutung der intuitionistischen Logik"

- Proposed to view a formula A as a computational problem, of type $\tau(A)$, the type of a potential solution or "realizer" of A.
- Example: $\forall_n \exists_{m>n} \mathrm{Prime}(m)$ has type $\mathbf{N} \to \mathbf{N}$.

The fact that nested implications may occur in A requires the concept of higher type computable functionals.

Fundamental property of computation:

evaluation must be finite.

- Principle of finite support. If $\mathcal{H}(\Phi)$ is defined with value n, then there is a finite approximation Φ_0 of Φ such that $\mathcal{H}(\Phi_0)$ is defined with value n.
- Monotonicity principle. If $\mathcal{H}(\Phi)$ is defined with value n and Φ' extends Φ, then also $\mathcal{H}(\Phi')$ is defined with value n.
- Effectivity principle. An object is computable just in case its set of finite approximations is (primitive) recursively enumerable (or equivalently, Σ^0_1-definable).

- Gödel (1958): "Über eine noch nicht benützte Erweiterung des finiten Standpunkts". Higher type term system T.
- Platek (1966): "Foundations of recursion theory".
- Scott (1969): LCF "Logic for Computable Functions". LCF's term language has arithmetic, booleans and recursion in higher types. LCF is based on classical logic.
- Plotkin (1977): Higher type term system PCF, with partiality.
- Martin-Löf (1984): constructive type theory. Formulas are types. Functionals are total.
- Proposal here: a constructive theory of computation in higher types, based on the Scott (1970) - Ershov (1977) model of partial continuous functionals.

points, ideals, abstract objects

$$\uparrow\downarrow$$

finite approximations

(Finitary) algebras (will be viewed as "non-flat Scott information systems").

- An algebra ι is given by its constructors.
- Examples:

$$0^{\mathbf{N}}, \mathrm{S}^{\mathbf{N}\to\mathbf{N}} \quad \text{for } \mathbf{N} \text{ (unary natural numbers)},$$
$$1^{\mathbf{P}}, S_0^{\mathbf{P}\to\mathbf{P}}, S_1^{\mathbf{P}\to\mathbf{P}} \quad \text{for } \mathbf{P} \text{ of (binary positive numbers)},$$
$$0^{\mathbf{D}} \text{ (axiom) and } \mathrm{C}^{\mathbf{D}\to\mathbf{D}\to\mathbf{D}} \text{ (rule) for } \mathbf{D} \text{ (derivations)}.$$

- Examples of "information tokens": $\mathrm{S}^n 0$ ($n \geq 0$), $\mathrm{S}^2 *$ (in **N**), $\mathrm{C}(\mathrm{C}0*)(\mathrm{C}*0)$ (in **D**) ($*$: special symbol; no information).
- An information token is total if it contains no $*$.
- In **D**: total token $\sim$ finite (well-founded) derivation.

For **D** (derivations):

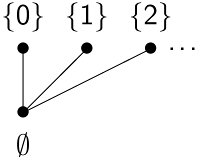
- $\{C0*, C*0\}$ is consistent, written $C0* \uparrow C*0$.
- $\{C0*, C*0\} \vdash C00$ (entails).
- Ideals: consistent and deductively closed sets of tokens.

Examples of ideals:

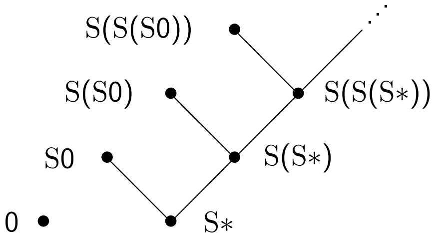
- $\{C0*, C**\}$.
- $\{C00, C0*, C*0, C**\}$.
- The deductive closure of a finite (well-founded) derivation.
- $\{C**, C(C**)*, C*(C**), C(C**)(C**), \dots\}$ (cototal).
- Locally correct, but possibly non well-founded derivations (Mints 1978).

Flat or non flat algebras?

- Flat:



- Non flat:



Non flat!

- Continuous maps $f\colon |\mathbf{N}| \to |\mathbf{N}|$ (see below) are monotone: $x \subseteq y \to fx \subseteq fy$.
- Easy: every constructor gives rise to a continuous function.
- Want: constructors have disjoint ranges and are injective (cf. the Peano axioms: $\mathrm{S}x \neq 0$ and $\mathrm{S}x = \mathrm{S}y \to x = y$).
- This holds for non-flat algebras, but not for flat ones. There constructors must be strict (i.e., $\mathrm{C}\vec{x}\emptyset\vec{y} = \emptyset$), hence

$$\begin{array}{lll} \text{in } \mathbf{P}\text{:} & \mathrm{S}_1\emptyset = \emptyset = \mathrm{S}_2\emptyset & \text{(overlapping ranges),} \\ \text{in } \mathbf{D}\text{:} & \mathrm{C}\emptyset\{0\} = \emptyset = \mathrm{C}\{0\}\emptyset & \text{(not injective).} \end{array}$$

The Scott-Ershov model of partial continuous functionals.

- Let $\mathbf{A} = (A, \mathrm{Con}_A, \vdash_A)$, $\mathbf{B} = (B, \mathrm{Con}_B, \vdash_B)$ be information systems (Scott). Function space: $\mathbf{A} \to \mathbf{B} := (C, \mathrm{Con}, \vdash)$, with

$$C := \mathrm{Con}_A \times B,$$

$$\{(U_i, b_i)\}_{i \in I} \in \mathrm{Con} := \forall_{J \subseteq I}(\bigcup_{j \in J} U_j \in \mathrm{Con}_A \to \{b_j\}_{j \in J} \in \mathrm{Con}_B),$$

$$\{(U_i, b_i)\}_{i \in I} \vdash (U, b) := (\{\, b_i \mid U \vdash_A U_i \,\} \vdash_B b).$$

- Partial continuous functionals of type ρ: the ideals in $\mathbf{C}_\rho$.

$$\mathbf{C}_\iota := (\mathrm{Tok}_\iota, \mathrm{Con}_\iota, \vdash_\iota), \qquad \mathbf{C}_{\rho \to \sigma} := \mathbf{C}_\rho \to \mathbf{C}_\sigma.$$

- $f \in |\mathbf{C}_\rho|$: limit of formal neighborhoods $U \in \mathrm{Con}_{\rho \to \sigma}$.
- $f \in |\mathbf{C}_\rho|$ computable: r.e. limit.

Why formalization?

- Correctness, precision, completeness. Likely to become the future standard in mathematics.
- Computer support: data banks, help in (interactive) proving.
- Computational content: realizability interpretation, soundness, extraction.

Terms (of higher type)

- T^+ (common extension of Gödel's T and Plotkin's PCF). Partial functionals allowed.
- Constants are given by their defining equations. Examples are Y (fixed point operator), $\mathcal{R}$ (structural recursion), ${}^{\mathrm{co}}\mathcal{R}$ (corecursion).
- An (external) semantics: for every closed term $\lambda_{\vec{x}}M$ of type $\vec{\rho} \to \sigma$ inductively define a set $[\![\lambda_{\vec{x}}M]\!]$ of tokens of type $\vec{\rho} \to \sigma$. $[\![\lambda_{\vec{x}}M]\!]$ is an ideal.

Definition ($a \in [\![\lambda_{\vec{x}} M]\!]$)

Case $\lambda_{\vec{x},y,\vec{z}} M$ with $\vec{x}$ free in M, but not y.

$$\frac{(\vec{U}, \vec{W}, a) \in [\![\lambda_{\vec{x},\vec{z}} M]\!]}{(\vec{U}, V, \vec{W}, a) \in [\![\lambda_{\vec{x},y,\vec{z}} M]\!]}(K).$$

Case $\lambda_{\vec{x}} M$ with $\vec{x}$ the free variables in M.

$$\frac{U \vdash a}{(U, a) \in [\![\lambda_x x]\!]}(V), \quad \frac{(\vec{U}, V, a) \in [\![\lambda_{\vec{x}} M]\!] \quad (\vec{U}, V) \subseteq [\![\lambda_{\vec{x}} N]\!]}{(\vec{U}, a) \in [\![\lambda_{\vec{x}}(MN)]\!]}(A).$$

For every constructor C and defined constant D:

$$\frac{\vec{U} \vdash \vec{a^*}}{(\vec{U}, \mathrm{C}\vec{a^*}) \in [\![\mathrm{C}]\!]}(\mathrm{C}), \quad \frac{(\vec{V}, a) \in [\![\lambda_{\vec{x}} M]\!] \quad \vec{U} \vdash \vec{P}(\vec{V})}{(\vec{U}, a) \in [\![D]\!]}(D),$$

with one rule (D) for every defining equation $D\vec{P}(\vec{x}\,) = M$.

TCF (theory of computable functionals), a variant of HA^ω with variables ranging over arbitrary partial continuous functionals.

- Terms from T^+. Constants for (partial) computable functionals, defined by equations.
- Inductively (and coinductively) defined predicates. Totality for ground types inductively defined.
- Induction := elimination (or least-fixed-point) axiom for a totality predicate. (Coinduction := greatest-fixed-point axiom for a coinductively defined predicate.)
- Minimal logic: $\to, \forall$ only. $=$ (Leibniz), $\exists$, $\vee$, $\wedge$ (Martin-Löf) inductively defined.
- $\bot := (\mathrm{False} = \mathrm{True})$. Ex-falso-quodlibet: $\bot \to A$ provable.

An extension TCF^+ of TCF, with variables

x, y, f	for partial continuous functionals,
a, b, c	for tokens,
U, V, W	for formal neighborhoods.

(All variables are typed).

- Internal semantics: we now can inductively define predicates $P_{\vec{x},M}a$ to mean $a \in [\![\lambda_{\vec{x}} M]\!]$.
- Based on these we can now formally prove in TCF^+ that (for example) $[\![Y]\!]f = \bigcup_n f^n\emptyset$, or more precisely

$$a \in [\![Y]\!]f \leftrightarrow \exists_n (a \in f^n\emptyset)$$

Proof sketch. Recall the defining equation

$$Yf = f(Yf).$$

It suffices to prove

For every $n > 0$, there is a derivation of $(U, a) \in [\![Y]\!]$ with D-height n if and only if $U^n\emptyset \vdash a$.

Every derivation of $(U, a) \in [\![Y]\!]$ must have the form

$$\dfrac{\dfrac{\dfrac{W \vdash (V, a)}{(W, V, a) \in [\![\lambda_f f]\!]} \qquad \dfrac{\dfrac{(U_i, a_i) \in [\![Y]\!]}{(W, U_i, a_i) \in [\![\lambda_f Y]\!]} \quad \dfrac{W \vdash (V_{ij}, a_{ij})}{(W, V_{ij}, a_{ij}) \in [\![\lambda_f f]\!]}}{(W, a_i) \in [\![\lambda_f (Yf)]\!]}}{(W, a) \in [\![\lambda_f (f(Yf))]\!]}}{(U, a) \in [\![Y]\!]}\,(Y), \text{ assuming } U \vdash W$$

with $V := \{\, a_i \mid i \in I \,\}$, $U_i := \{\, (V_{ij}, a_{ij}) \mid j \in I_i \,\}$.
"$\to$": by induction on the D-height. "$\leftarrow$": by induction on n.

Computational content of proofs. Two alternatives.

1. Might be seen directly and expressed by a term M in T^+. Then one needs to prove that M realizes A (soundness).
2. Alternative (works always): extract computational content from a proof of A. Soundness proof can be machine generated automatically.

Example: density theorem. The first alternative will be used.

Inductive definition of $(\vec{U}, a) \in [\![\lambda_{\vec{x}} M]\!]$ not necessary here: since no defined constants occur, $[\![\lambda_{\vec{x}} M]\!]$ of type $\vec{\rho} \to \sigma$ can be defined from a "Σ-formula" $(\vec{U}, a) \in [\![\lambda_{\vec{x}} M]\!]$.

TCF^+: formal language, axioms.

- Need coding of types ρ, tokens a, formal neighborhoods U.
- U as $\{ a_i \mid i < n \}$, finite enumerated set (a_i prim. rec.).
- Δ-formula: equation $t = 0$ with t prim. rec. term.
- Fix $W = \{ (U_i, b_i) \mid i < n \}$, $z := \{ a \mid C(a) \}$ (C Δ-formula).

$$Wz := \{ b_i \mid \forall_{a \in U_i} C(a) \} \quad \text{(application of } W \text{ to } z\text{)}$$

can be written as a finite enumerated set.

- (Typed) term: built from variables and constructors:

$$M, N ::= x^\rho \mid \mathrm{C}^\rho \mid (\lambda_{x^\rho} M^\sigma)^{\rho\to\sigma} \mid (M^{\rho\to\sigma} N^\rho)^\sigma.$$

- Σ-formula: $t = 0$ (Δ-formula), $a \in x$; $\wedge$, $\vee$, $\exists$, $\forall_{i<n}$.
- An ideal is a consistent deductively closed set of tokens.

$$I_\rho x := \forall_{a,b\in x}(a \uparrow b) \wedge \forall_{U\subseteq x}\forall_a(U \vdash a \to a \in x).$$

- Σ-comprehension. Let $C(a, \vec{y}\,)$ be a Σ-formula.

$$\begin{aligned} I_{\vec\rho}\vec{y} &\to \forall_{a,b}(C(a,\vec{y}\,) \to C(b,\vec{y}\,) \to a \uparrow b) \\ &\to \forall_{U,b}(\forall_{a\in U} C(a,\vec{y}\,) \to U \vdash b \to C(b,\vec{y}\,)) \\ &\to \exists_x \forall_a (a \in x \leftrightarrow C(a,\vec{y}\,)). \end{aligned}$$

Assume that no defined constant D occurs in M. Then for $\lambda_{\vec{x}} M$ of type $\vec{\rho} \to \sigma$ we can define $(\vec{U}, a) \in [\![\lambda_{\vec{x}} M]\!]$ as a Σ-formula.

Definition ($a \in M$ as Σ-formula)

$$(a \in M) := \exists_{\vec{U} \subseteq \vec{x}}((\vec{U}, a) \in [\![\lambda_{\vec{x}} M]\!]).$$

with $\vec{x}$ the free variables of M.

- One can prove that every closed term M denotes an ideal, i.e.,

$$a, b \in M \to a \uparrow b, \qquad U \subseteq M \to U \vdash b \to b \in M.$$

- $(M = N) := \forall_a (a \in M \leftrightarrow a \in N)$ (extensional equality).
- $G_\rho x$ (x is a total ideal) is defined by induction on ρ:

$$\begin{aligned} G_\iota x \quad &:= I_\rho x \wedge x \text{ contains a total token } a, \\ G_{\rho\to\sigma} f \quad &:= I_{\rho\to\sigma} f \wedge \forall_x (G_\rho x \to \underbrace{\exists_y (y = fx \wedge G_\sigma y)}_{G_\sigma(fx)}). \end{aligned}$$

Lemma (Extension)

If $G_\rho f$, $I_\rho g$ and $f \subseteq g$, then $G_\rho g$.

Lemma (Continuity of application)

$b \in fx \leftrightarrow \exists_{U \subseteq x} ((U, b) \in f)$.

Definition (Extensional equality $=^t_\rho$ on total ideals)

$$(x =^t_\iota y) := (x = y),$$
$$(f =^t_{\rho\to\sigma} g) := \forall_{x\in G_\rho}(fx =^t_\sigma gx).$$

Theorem (Ershov, Longo & Moggi)

$$\forall_{x,y\in G_\rho}\forall_{f\in G_{\rho\to\sigma}}(x =^t_\rho y \to fx =^t_\sigma fy).$$

Proof. Uses a characterization of $=^t_\rho$:

$$\forall_{f,g\in G_\rho}(f =^t_\rho g \leftrightarrow G_\rho(f \cap g)).$$

The total functionals are dense (w.r.t. the Scott topology) in the space of all partial continuous functionals of type ρ.

$$\forall_{U \in \mathrm{Con}_\rho} \exists_{x \in G_\rho} (U \subseteq x).$$

One can explicitly define a realizer via Δ-formulas:

Theorem (Density; Kreisel, Ershov, U. Berger)

For every type $\rho = \rho_1 \to \ldots \to \rho_p \to \iota$ *we have* Δ*-formulas* TExt_ρ *and* Sep^i_ρ $(i = 1, \ldots, p)$ *such that the following can be proved in* TCF^+*. For any given* $U, V \in \mathrm{Con}_\rho$

(a) $U \subseteq \{ a \mid \mathrm{TExt}_\rho(U, a) \} \in G_\rho$ *and*

(b) $U \not\uparrow_\rho V \to \vec{z}_{U,V} \in G \wedge U\vec{z}_{U,V} \not\uparrow_\iota V\vec{z}_{U,V}$,

where $\vec{z}_{U,V} = z_{U,V,1}, \ldots, z_{U,V,p}$ *and* $z_{U,V,i} = \{ a \mid \mathrm{Sep}^i_\rho(U, V, a) \}$.

Proof. By induction on ρ.

Conclusion

- Basic semantical concept: partial continuous functionals.
- Ideal (or point): a consistent deductively closed set of tokens.
- TCF: Theory of computable functionals.
- TCF^+: Refinement, with $\forall_a$ and $\forall_U$, in addition to $\forall_x$.
- Formalization of the density theorem in TCF^+.

References

- U. Berger, K. Miyamoto, H.S. and M. Seisenberger, Minlog - A tool for program extraction supporting algebras and coalgebras. Calco-Tools 2011.
- S. Huber, B. Karádais and H.S., Towards a formal theory of computability. 2010.
- K. Miyamoto and H.S., Program extraction in exact real arithmetic. To appear in MSCS.
- H.S. and S.S. Wainer, Proofs and Computations. Perspectives in Logic, ASL & Cambridge UP, 2012.