Towards an arithmetic for partial computable functionals

(outline of the talk)

Basil A. Karádais

August 12, 2013

1 Partiality, continuity, higher types

Consider the statement

$$\forall_{x\in\mathbb{R}} (x = 0 \lor x \neq 0) \ . \tag{1}$$

- It is unlikely that there is an algorithm f that *decides* (1), i.e., returns $\mathtt{tt}$ if $x = 0$ and $\mathtt{ff}$ if $x \neq 0$: on input $x = 0$ it would run forever (think of the decimal representation of x); it would only *semi-decide* it, i.e., it would be a *partial* algorithm.
- The algorithm as a mapping $f : \mathbb{R} \to \mathbb{B}$ is *discontinuous* at 0.
- Reals are Cauchy sequences of rationals; rationals are pairs of integers; integers are pairs of naturals; so

  $$f : (\mathbb{N} \to (\mathbb{N}\times\mathbb{N})\times(\mathbb{N}\times\mathbb{N}))\times(\mathbb{N}\to\mathbb{N}) \to \mathbb{B} \ . \tag{2}$$

  Algorithms like the above (and reals too) are certain higher-type functionals over $\mathbb{N}$ and $\mathbb{B}$.
- Domain theory provides solid mathematical grounds on which to construe algorithms as *partial continuous higher-type functionals*.
- Aiming at an implementation in a proof assistant, so at a *formal theory of higher-type computability*, we develop a *constructive* and *bottom-up* version of domain theory.

2 A bottom-up approach to higher-type computability through approximations

Three requirements for a theory of higher-type computation:

- *Principle of monotonicity*: if an algorithm terminates on some functional input f with output y, then it should still terminate with the same output y even if we gave more information on the input, namely some f' with $f \subseteq f'$.
- *Principle of finite support*: in order to compute some finite output an algorithm should only need finite information on the input.

- *Effectivity principle*: an algorithm should be approximated by a recursively enumerated set of *finite pieces of data*.

2.1 Approximations

Organize the *finite approximations* of objects of a given type as an *information system*.

- The *tokens* of information $a, b, c, \dots$ form a countable set: Tok.
- Finite collections of tokens U can be *consistent*: $U \in \mathsf{Con}$.
- A consistent set U may *entail* a token b: $U \vdash b$.
- Axioms for the approximations:

$$\begin{aligned} &\{a\} \in \mathsf{Con}\ , && (3)\\ &U \in \mathsf{Con} \wedge V \subseteq U \to V \in \mathsf{Con}\ , && (4)\\ &U \in \mathsf{Con} \wedge a \in U \to U \vdash a\ , && (5)\\ &U \vdash V \wedge V \vdash a \to U \vdash a\ , && (6)\\ &U \vdash a \to U \cup \{a\} \in \mathsf{Con}\ . && (7) \end{aligned}$$

- *Coherence*: consistency reduces to a binary predicate:

$$U \in \mathsf{Con} \leftrightarrow \forall_{a,b \in U} \{a,b\} \in \mathsf{Con}\ ; \tag{8}$$

  write $a \asymp b$ for $\{a,b\} \in \mathsf{Con}$.

- *Atomicity*: entailment reduces to a binary predicate:

$$U \vdash b \leftrightarrow \exists_{a \in U} \{a\} \vdash b\ ; \tag{9}$$

  write $U \vdash^A b$ for a neighborhood with an *atomic closure*.

If ρ and σ are coherent information systems, define their *function space* $\rho \to \sigma$.

- Function space tokens give information on the *graph* of a mapping. It is $\langle U, b\rangle \in \mathsf{Tok}_{\rho\to\sigma}$ if

$$U \in \mathsf{Con}_\rho \wedge b \in \mathsf{Tok}_\sigma\ .$$

- Consistency corresponds to single valuedness. It is $\langle U, b\rangle \asymp_{\rho\to\sigma} \langle U', b'\rangle$ if

$$U \asymp_\rho U' \to b \asymp_\sigma b'\ .$$

- If $W = \{\langle U_i, b_i\rangle \mid i < n\} \in \mathsf{Con}_{\rho\to\sigma}$ and $U \in \mathsf{Con}_\rho$, the *application* of W to U is

$$W \cdot U = \{b_i \mid U \vdash_\rho U_i\}\ .$$

- Entailment expresses informational economy. It is $W \vdash_{\rho\to\sigma} \langle U, b\rangle$ if

$$W \cdot U \vdash_\sigma b\ .$$

- **Fact.** If ρ and σ are coherent information systems, then $\rho \to \sigma$ is a coherent information system.
- **Fact.** If ρ and σ are atomic-coherent information systems, then $\rho \to \sigma$ is an atomic-coherent information system.

2.2 Objects (numbers, functions, functionals) as ideals

Recover the *objects* $x \subseteq \mathsf{Tok}$ of the type as *ideals*. Write $x \in \mathsf{Ide}$.

- An object is *consistent* ("well-defined"):

$$U \subseteq x \to U \in \mathsf{Con} \ .$$

- An object is *deductively closed* ("informationally complete"):

$$U \subseteq x \wedge U \vdash b \to b \in x \ .$$

Endow the set of objects with the *Scott topology*.

- A set $\mathscr{U} \subseteq \mathsf{Ide}$ is *open* if it is upwards closed (monotonicity principle):

$$x \in \mathscr{U} \wedge x \subseteq y \to y \in \mathscr{U} \ ,$$

  and features finite support:

$$x \in \mathscr{U} \to \underset{U \subseteq x}{\exists} \overline{U} \in \mathscr{U} \ ,$$

  where $\overline{U} := \{b \mid U \vdash b\}$.

- The collection of the *cones of ideals* $\nabla U := \{x \in \mathsf{Ide} \mid U \subseteq x\}$ over consistent sets $U \in \mathsf{Con}$,

$$\{\nabla U \mid U \in \mathsf{Con}\} \ ,$$

  is a *basis for the Scott topology*.

- T_0-separation, but cartesian closure.

- **Fact.** A mapping $f : \mathsf{Ide}_\rho \to \mathsf{Ide}_\sigma$ is *Scott-continuous* when it is monotone

$$x \subseteq y \to f(x) \subseteq f(y) \ ,$$

  and it satisfies the principle of finite support:

$$b \in f(x) \to \underset{U \subseteq x}{\exists} b \in f(U) \ .$$

  Fact. $\mathsf{Ide}_\rho \to \mathsf{Ide}_\sigma \cong \mathsf{Ide}_{\rho \to \sigma}$.

2.3 Concrete types

A toy *type system*.

- Base types are *algebras* $\mathbb{A}$ inductively generated by constructors $C_1, \ldots, C_K$ of respective arities $r_1, \ldots r_K$:

$$a_1, \ldots, a_r \in \mathbb{A} \to C a_1 \cdots a_r \in \mathbb{A} \ .$$

- *Naturals* $\mathbb{N}$ are given by the constructors 0, S, of respective arities 0, 1.

- *Booleans* $\mathbb{B}$ are given by the constructors $\mathtt{tt}$, $\mathtt{ff}$, of respective arities 0, 0.

- *Binary trees* (or *derivations*) $\mathbb{D}$ are given by the constructors 0, B, of respective arities 0, 2.
- Endow every algebra with *partiality*: either by adding a *pseudotoken* $*$ (*flat types*), or by adding a *pseudoconstructor* $*$ of arity 0 (*non-flat types*).
- $\mathbb{B}$, $\mathbb{N}$, $\mathbb{D}$ are types; if ρ, σ are types, then $\rho \to \sigma$ is a type.

Every type is interpreted as a *coherent information system*.

- The tokens of $\mathbb{D}$ are the elements of the algebra (generated together with the pseudoconstructor).
- Consistency:

$$\begin{aligned}&a \asymp_{\mathbb{D}} * \wedge * \asymp_{\mathbb{D}} a \ ,\\ &0 \asymp_{\mathbb{D}} 0 \ ,\\ &a \asymp_{\mathbb{D}} a' \wedge b \asymp_{\mathbb{D}} b' \to Bab \asymp_{\mathbb{D}} Ba'b' \ .\end{aligned}$$

- Entailment:

$$\begin{aligned}&U \vdash_{\mathbb{D}} * \ ,\\ &\{0,\ldots,0\} \vdash_{\mathbb{D}} 0 \ ,\\ &\{a_1,\ldots,a_m\} \vdash_{\mathbb{D}} a \wedge \{b_1,\ldots,b_m\} \vdash_{\mathbb{D}} b \to \{Ba_1b_1,\ldots,Ba_mb_m\} \vdash_{\mathbb{D}} Bab \ ,\\ &U \setminus \{*\} \vdash b \to U \vdash b \ .\end{aligned}$$

- If ρ, σ are types interpreted as coherent information systems, then $\rho \to \sigma$ is interpreted as their function space.
- *Non-flat base types* increase complexity of the arguments but allow for more flexibility and nice properties.
- For base types over $\mathbb{N}$ and $\mathbb{B}$ (and other *non-superunary algebras*) we may exclusively use *atomic-coherent information systems*, but in general, like with $\mathbb{D}$, just coherent ones, due to the *Coquand counterexample*:

$$\{B0*, B*0\} \vdash B00 \ \wedge \ \{B0*, B*0\} \nvdash^{A} B00 \ ; \tag{10}$$

3 Contributions

Two major questions in higher-type computability theory:

- **Density**: *In the presence of partiality, can we recover the* total objects *of a given type?*

  Many important consequences: one of them, choice principle for total functionals (i.e., the axiom of choice is provable).

  Kleene 1959, Kreisel 1959: before domain theory.

  Berger 1993: density ("total objects are dense in the partial ones") in abstract domain theory.

  Schwichtenberg 1996: density for flat systems.

Schwichtenberg 2006: density for non-flat systems over $\mathbb{N}$ and $\mathbb{B}$.

Huber 2010: density for non-flat systems.

Huber–K.–Schwichtenberg 2010: formalization of density for non-flat systems.

- **Definability**: *Given an object at some type as a recursively enumerable set of tokens (i.e., with an algorithm listing its elements), what basic constructs do we need to have in the formal language in order to express it?*

  Of the same importance in higher-type computability as the characterization of recursive functions of type $\mathbb{N} \to \mathbb{N}$ by certain schemes (initial functions, composition, primitive recursion, μ-recursion).

  Plotkin 1977: definability for a theory over $\mathbb{N}$ and $\mathbb{B}$, without approximations; need least fixed point functionals and two "parallel operations".

  Schwichtenberg 1999: definability for flat systems over $\mathbb{N}$ and $\mathbb{B}$; Plotkin's extra terms suffice.

3.1 Density in coherent systems

- A *total token* is a token with no $*$'s. A *total object* at a base type $\mathbb{A}$ is an ideal that contains a total token. At type $\rho \to \sigma$, a total object is one that gives total values to total arguments. Write G_ρ for the totals at ρ.

- A type is *separating* if inconsistent neighborhoods in the type can be separated by *total objects* of appropriate type.

- A type is *dense* if

$$\forall_{U \in \mathsf{Con}} \exists_{x \in G} U \subseteq x ,$$

  that is, the set G is *dense with respect to the Scott topology*: $U \subseteq x$ means $x \in \nabla U$, so $G \cap \overline{U} \neq \varnothing$, for all U's.

- All of the previous proofs are by *mutual induction*: if ρ is dense and σ is separating, then $\rho \to \sigma$ is separating; if ρ is separating and σ is dense, then $\rho \to \sigma$ is dense; so all types all simultaneously separating and dense.

  Elegant argument, but complicated implementation.

- Call a type *finitely separating* if inconsistent neighborhoods in the type can be separated by *neighborhoods* of appropriate type.

  Result 1.1. *Every type is finitely separating (no density required).*

  Result 1.2. *Every type is dense (use Result 1.1 as a lemma).*

  In this way we obtain a "linear" proof of density.

3.2 Definability in atomic-coherent systems

- **Result 2.** *To capture all computable functionals over $\mathbb{N}$ and $\mathbb{B}$, we need one more "parallel operation" other than Plotkin's.*

- What about more general base types like $\mathbb{D}$?

  In the proof of Result 1 we made heavy and crucial use of the *comparability property*

  $$U \asymp V \to U \vdash V \vee V \vdash U \ ,$$

  a converse of the "propagation of consistency" axiom (7).

  Result 3. *A coherent information system induced by an algebra has the comparability property if and only if the algebra has at most unary constructors.*

  For base types like $\mathbb{D}$ we need a better understanding of non-atomic systems.

3.3 Implicit atomicity in non-atomic coherent systems

- Counter-observation to the Coquand counterexample (10): there is some hidden atomicity even in non-atomic systems.

  $$\left\{B0*, B*0\right\} \vdash B00 \Leftrightarrow B\begin{bmatrix} 0 & * \\ * & 0 \end{bmatrix} \vdash B\begin{bmatrix} 0 \\ 0 \end{bmatrix} \Leftrightarrow \begin{bmatrix} 0 & * \\ * & 0 \end{bmatrix} \vdash^{A} \begin{bmatrix} 0 \\ 0 \end{bmatrix} .$$

- Elaboration of the notion of (not necessarily atomic) entailment for algebras and redefinition in terms of entailment on appropriate *matrix systems which are atomic*.

  Result 4. *In a coherent information system induced by an algebra, for every neighborhood there is a equientailing token.*

  For example, $\left\{B0*, B*0\right\} \sim \left\{B00\right\}$. Does this hold for higher types?

- Call a type *implicitly atomic* if for every neighborhood there is an equivalent one whose closure is atomic, in the sense of (9).

  Result 5. *Every type is implicitly atomic.*

4 Outlook

- Use of implicit atomicity to simplify arguments and obtain nicer results in the general case of types over any kinds of algebra.

- Similarly to separation, can one prove density also with a finite witness? Can one retain the linear argumentation?

- What more is needed in order to establish definability for types over general algebras?

- **Result 6.** *Coherent information systems correspond to "coherent" domains.*

  What is the domain-theoretic counter-part of (implicitly) atomic information systems?

Thanks

To Apostolos, Brent, Dirk, Rhea, Sifis.