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

$$\forall x \in \mathbb{R} \ (x = 0 \lor x \neq 0).$$

(1)

- It is unlikely that there is an algorithm $f$ that decides (1), i.e., returns $\text{tt}$ if $x = 0$ and $\text{ff}$ 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

$$f : (\mathbb{N} \to (\mathbb{N} \times \mathbb{N}) \times (\mathbb{N} \times \mathbb{N})) \times (\mathbb{N} \to \mathbb{N}) \to \mathbb{B}.$$ (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 \(a, b, c, \ldots\) form a countable set: \(\text{Tok}\).
- Finite collections of tokens \(U\) can be consistent: \(U \in \text{Con}\).
- A consistent set \(U\) may entail a token \(b\): \(U \vdash b\).
- Axioms for the approximations:
  
  \[
  \begin{align*}
  \{a\} & \in \text{Con} , \\
  U \in \text{Con} \land V \subseteq U & \to V \in \text{Con} , \\
  U \in \text{Con} \land a \in U & \to U \vdash a , \\
  U \vdash V \land V \vdash a & \to U \vdash a , \\
  U \vdash a & \to U \cup \{a\} \in \text{Con} .
  \end{align*}
  \]

- Coherence: consistency reduces to a binary predicate:
  
  \[
  U \in \text{Con} \iff \forall a, b \in U \{a, b\} \in \text{Con} ;
  \]

write \(a \simeq b\) for \(\{a, b\} \in \text{Con}\).

- Atomicity: entailment reduces to a binary predicate:
  
  \[
  U \vdash b \iff \exists a \in U \{a\} \vdash b ;
  \]

write \(U \vdash^A b\) for a neighborhood with an atomic closure.

If \(\rho\) and \(\sigma\) are coherent information systems, define their function space \(\rho \to \sigma\).

- Function space tokens give information on the graph of a mapping. It is \(\langle U, b \rangle \in \text{Tok}_{\rho \to \sigma}\) if
  
  \[
  U \in \text{Con}_\rho \land b \in \text{Tok}_\sigma .
  \]

- Consistency corresponds to single valuedness. It is \(\langle U, b \rangle \simeq_\rho \to_\sigma \langle U', b' \rangle\) if
  
  \[
  U \simeq_\rho U' \to b \simeq_\sigma b' .
  \]

- If \(W = \{\langle U_i, b_i \rangle \mid i < n \} \in \text{Con}_{\rho \to \sigma}\) and \(U \in \text{Con}_\rho\), 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 \(\rho\) and \(\sigma\) are coherent information systems, then \(\rho \to \sigma\) is a coherent information system.

- Fact. If \(\rho\) and \(\sigma\) 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 $x \subseteq \text{Tok}$ of the type as ideals. Write $x \in \text{Ide}.$

- An object is consistent (“well-defined”):
  
  $$U \subseteq x \rightarrow U \in \text{Con}. $$

- An object is deductively closed (“informationally complete”):
  
  $$U \subseteq x \wedge U \vdash b \rightarrow b \in x. $$

Endow the set of objects with the Scott topology.

- A set $U \subseteq \text{Ide}$ is open if it is upwards closed (monotonicity principle):
  
  $$x \in U \wedge x \subseteq y \rightarrow y \in U, $$

  and features finite support:

  $$x \in U \rightarrow \exists \ U \subseteq x \ U \in U, $$

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

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

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

  is a basis for the Scott topology.

- $T_0$-separation, but cartesian closure.

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

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

  and it satisfies the principle of finite support:

  $$b \in f(x) \rightarrow \exists \ U \subseteq x \ b \in f(U). $$

  **Fact.** $\text{Ide}_\rho \rightarrow \text{Ide}_\sigma \equiv \text{Ide}_{\rho \rightarrow \sigma}.$

2.3 Concrete types

A toy type system.

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

  $$a_1, \ldots, a_r \in A \rightarrow C_{a_1} \cdots a_r \in A. $$

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

- Booleans $\mathbb{B}$ are given by the constructors $\texttt{t}, \texttt{f},$ 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 $\rho$, $\sigma$ are types, then $\rho \rightarrow \sigma$ 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:**
  
  \[
  a \vdash_{\mathbb{D}} * \land * \vdash_{\mathbb{D}} a ,
  \]
  
  \[
  0 \vdash_{\mathbb{D}} 0 ,
  \]
  
  \[
  a \vdash_{\mathbb{D}} a' \land b \vdash_{\mathbb{D}} b' \rightarrow Bab \vdash_{\mathbb{D}} Ba'b' .
  \]

- **Entailment:**
  
  \[
  U \vdash_{\mathbb{D}} *,
  \]
  
  \[
  \{0, \ldots, 0\} \vdash_{\mathbb{B}} 0 ,
  \]
  
  \[
  \{a_1, \ldots, a_m\} \vdash_{\mathbb{D}} a \land \{b_1, \ldots, b_m\} \vdash_{\mathbb{D}} b \rightarrow \{Ba_1 b_1, \ldots, Ba_m b_m\} \vdash_{\mathbb{D}} Bab ,
  \]
  
  \[
  U \setminus \{*\} \vdash b \rightarrow U \vdash b .
  \]

- If $\rho$, $\sigma$ are types interpreted as coherent information systems, then $\rho \rightarrow \sigma$ 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:
  
  \[
  \{B0*, B*0\} \vdash B00 \land \{B0*, B*0\} \not\vdash^A B00 ;
  \]

\section{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.

• **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 $N \to N$ by certain schemes (initial functions, composition, primitive recursion, $\mu$-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 $\ast$’s. A total object at a base type $\Delta$ 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 $G_\rho$ for the totals at $\rho$.

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

• A type is dense if

$$\forall U \in \text{Con} \exists x \in G \ U \subseteq x,$$

that is, the set $G$ is dense with respect to the Scott topology: $U \subseteq x$ means $x \in \bigvee U$, so $G \cap U \neq \emptyset$, for all $U$’s.

• All of the previous proofs are by mutual induction: if $\rho$ is dense and $\sigma$ is separating, then $\rho \to \sigma$ is separating; if $\rho$ is separating and $\sigma$ 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 $D$?

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

\[ U \sim V \rightarrow U \vdash V \lor V \vdash U, \]

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 $D$ 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.

\[ \{B0*, B+0\} \vdash B00 \iff B \begin{bmatrix} 0 & * \\ * & 0 \end{bmatrix} \vdash B \begin{bmatrix} 0 \\ 0 \end{bmatrix} \iff B \begin{bmatrix} 0 & * \\ * & 0 \end{bmatrix} \vdash A \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \]

• 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 an equientailing token.

For example, $\{B0*, B+0\} \sim \{B00\}$. 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.

The domain-theoretic counter-part of (implicitly) atomic information systems?

---

Thanks

To Apostolos, Brent, Dirk, Rhea, Sifis.