Partial Combinatory Algebras – Between realizability and oracle computations

Jaap van Oosten

Department of Mathematics
Utrecht University

Tulips Seminar, March 16, 2023

Computability Theory

A function $f : U \to \mathbb{N}$ for $U \subseteq \mathbb{N}$ is called a *partial function* on $\mathbb{N}$; note that a partial function may be total (in case $U = \mathbb{N}$).
Such a function is *computable* if there is a Turing machine T such that for all $n \in \mathbb{N}$ we have:

- if $n \in U$ then T, with input n, reaches a halting state and outputs $f(n)$;
- if T, with input n, reaches a halting state then $n \in U$.

Think of a Turing machine as a program in a very primitive computer language.

We can enumerate all Turing machines: $T_1, T_2, \ldots$. To every T_i corresponds a partial function ϕ_i as before; the domain of ϕ_i is the set

$$\{n \in \mathbb{N} \mid T_i \text{ reaches a halting state with input } n\}$$

We may write nm for $\phi_n(m)$. This is not associative; when we write $n_1 n_2 \cdots n_k$ we mean $(\cdots((n_1 n_2)n_3)\cdots)n_k$.
Since the ϕ_i are *partial* functions, such expressions need not denote anything. We write: $nm\downarrow$ to indicate that m is in the domain of ϕ_n.

Subsets of $\mathbb{N}^k$ are, in computability theory, often called 'problems'; the 'problem' is to decide by an algorithm whether or not a given k-tuple of natural numbers is an element of that set. The algorithm (which we identify with a Turing machine) is then a 'solution' of the problem. One of the oldest such problems was the *Halting Problem* (Turing): the set

$$H \;=\; \{(n,m) \mid m \text{ is in the domain of } \phi_n\}$$

And Turing proved:
Theorem The halting problem is unsolvable (i.e., has no solution). One also says that H is an *undecidable set*.

The theory of computability aims to classify subsets of $\mathbb{N}^k$ in terms of 'difficulty to calculate'. An important tool is the notion of *Turing reducibility*: for subsets A, B of $\mathbb{N}^k$, the notation $A \leq_T B$ (A is Turing reducible to B) if a Turing machine can decide the question '$n \in A$?' provided it has access to answers to '$m \in B$?' (for example, by consulting a database for B). Turing said the machine may 'consult an oracle'.

Examples of similar structures:
$\mathcal{K}_2$ ("Kleene's second model") is the set $\mathbb{N}^{\mathbb{N}}$ of functions from $\mathbb{N}$ to $\mathbb{N}$. We assume a coding of sequences $\langle a_0, \ldots a_{n-1} \rangle$. For functions α, β, we let $\alpha\beta\downarrow$ if and only if for each natural number n there is some k such that

$$\alpha(\langle n, \beta(0), \ldots, \beta(k-1) \rangle) > 0$$

and we let $\alpha\beta(n) = \alpha(\langle n, \beta(0), \ldots, \beta(k-1) \rangle) - 1$ for the least such k.
Note that we may view the definition of $\alpha\beta$ as an 'oracle consultation': α consults the oracle β a number of times, until it decides it has enough information.
We may restrict the structure $\mathcal{K}_2$ to the computable functions: we write $\mathcal{K}_2^{\mathrm{rec}}$.

Examples of similar structures (continued)

A *total* structure of this kind was defined by Dana Scott: let $\mathcal{S}$ be the powerset of $\mathbb{N}$. We assume bijections:

$$\begin{aligned} \langle\cdot,\cdot\rangle : &\quad \mathbb{N}^2 \to \mathbb{N} \\ e_{-} : &\quad \mathbb{N} \to \mathcal{P}_{\text{fin}}(\mathbb{N}) \end{aligned}$$

Let $AB = \{y |$ for some $n, e_n \subseteq B$ and $\langle n, y\rangle \in A\}$ The functions ϕ_A are continuous when $\mathcal{S}$ is given the *Scott topology*: identify $\mathcal{S}$ with the set of all functions $\mathbb{N} \to \{0,1\}$; give $\{0,1\}$ the Sierpinski topology (with $\{1\}$ the one nontrivial open set) and $\mathcal{S}$ the product topology).

There is a common axiomatics underlying these structures; we speak of *Partial Combinatory Algebras* (PCAs).
The final abstract definition is due to Solomon Feferman (1975), but the basic idea (restricting to total computations) was laid out by M. Schönfinkel around 1920.
Peter Johnstone therefore calls PCAs “Schönfinkel algebras”.
Which prompts the following short biographical intermezzo:

ПЕТРА ✱ ИМПЕРАТО

Moses Ilyich (or is it *Isayevich*?) *Schönfinkel* is one of the more mysterious figures in the history of logic. He was born in 1889 (or was it 1887?) in Ukraina. He worked from 1914 (!) to 1924 under Hilbert in Göttingen, during which period one paper appeared: *Über die Bausteine der mathematischen Logik* in *Mathematische Annalen* 92, 1924. However, this paper appears to have been written by someone else, who took notes during lectures by Schönfinkel.
A second paper, coauthored by Bernays, appeared in 1927; by this time, however, Schönfinkel was already in a mental hospital in Moscow.
He died in 1942 in Moscow; his papers were used for firewood by his neighbours.
Stephen Wolfram, who has a voluminous piece about Schönfinkel on his web page, also relates that his mother was from a family called "Lurie"; and the Lurie's were business partners of father Schönfinkel.

A Partial Combinatory Algebra is a set A with a partial binary operation $(a, b) \mapsto ab$ and special elements k and s, which satisfy:

$$\begin{gathered} \mathsf{k}x\downarrow \\ (\mathsf{k}a)b = a \\ (\mathsf{s}a)b\downarrow \end{gathered}$$

and: if $ac(bc)\downarrow$ then $\mathsf{s}abc\downarrow$ and

$$\mathsf{s}abc = (ac)(bc)$$

.

The letter k stands for "Konstante Funktion"; the letter s is mysteriously called "Verschmelzungsfunktion" (blending function). The original (Schönfinkel's) aim: to provide an alternative foundation of mathematics in which not sets, but *functions* are the primitive notion.

We use the following conventions for brackets and other notations: a statement $t = s$ implies that t, s and all their subterms are defined.
We write $t \preceq s$ to mean: if $s\downarrow$ then $t = s$. We write $t \simeq s$ to mean $t \preceq s$ and $s \preceq t$.
Examples: $\mathsf{s}abc \preceq ac(bc)$; $\mathsf{k}(bx) \simeq bx$.

Basic facts about PCAs

Let A be a PCA.

We consider expressions obtained from variables $(x, y, z, u, v, \ldots)$, elements of A $(a, b, c, \ldots)$, and the juxtaposition operation: e.g., x, a, $x(ab)y$, $xayb$.

For any such expression t in variables $x_0, \ldots, x_n$ there is an element $\Lambda x_0 \cdots x_n.t$ with the following properties: for each tuple $a_0, \ldots, a_n$ from A we have

- $(\Lambda x_0 \cdots x_n.t)a_0 \cdots a_{n-1} \downarrow$
- $(\Lambda x_0 \cdots x_n.t)a_0 \cdots a_n \preceq t(a_0, \ldots, a_n)$

For example: for $\Lambda x.x$ one can take skk: $\mathsf{skk}a = \mathsf{k}a(\mathsf{k}a) = a$.

Let $\mathsf{p} = \Lambda xyz.zxy$ so $\mathsf{p}ab = \Lambda z.zab$; let $\mathsf{p}_0 = \Lambda v.v\mathsf{k}$ and let $\mathsf{p}_1 = \Lambda v.v(\Lambda wu.u)$. Then $\mathsf{p}_0(\mathsf{p}ab) = a$ and $\mathsf{p}_1(\mathsf{p}ab) = b$ so p is an *ordered pair* operator, with *unpairings* p_0 and p_1.

There are also *Booleans* t and f and a *definition by cases* term C satisfying $C\mathsf{t}ab = a$ and $C\mathsf{f}ab = b$.

Some Computability theory in a PCA A
There is a copy of $\mathbb{N}$ in A: $\{\bar{n} \mid n \in \mathbb{N}\}$, the *Curry numerals*.
For every k-ary partial recursive function ϕ there is an element a_ϕ of A simulating ϕ: for all $n_1, \ldots, n_k \in \mathbb{N}$,

$$a_\phi \bar{n}_1 \cdots \bar{n}_k \preceq \overline{\phi(n_1, \ldots, n_k)}$$

We can manipulate finite sequences $\langle a_0, \ldots, a_{k-1}\rangle$ of elements of A. For example we have for suitable $c, d \in A$:

$$c\bar{i}\langle a_0, \ldots, a_{k-1}\rangle = a_i$$
$$d\langle a_0, \ldots, a_{k-1}\rangle = \bar{k}$$

Some Computability theory in a PCA A (continued) We have a *recursion theorem* in every PCA A: there are elements y, z satisfying, for each $f \in A$:

i) $\mathrm{y}f \preceq f(\mathrm{y}f)$

ii) $\mathrm{z}f\downarrow$

iii) $\mathrm{z}fx \preceq f(\mathrm{z}f)x$ for all $x \in A$.

Theorem. Let A be a PCA. For every computable function F on the natural numbers, there is an element ϕ of A satisfying $\phi\overline{n} \simeq \overline{F(n)}$ (here $\overline{n}$ is the Curry numeral corresponding to the natural number n).

Applicative morphisms of PCAs

Let A, B be PCAs. An *applicative morphism* $A \to B$ is a total relation γ (we think of γ as a function from A to the set of nonempty subsets of B, so (A, γ) is an assembly over B) for which there is an element $\mathsf{r} \in B$ which satisfies:

For each pair a, a' of elements of A and $b \in \gamma(a), b' \in \gamma(a')$, if $aa' \downarrow$ in A then $\mathsf{r}bb' \downarrow$ in B, and $\mathsf{r}bb' \in \gamma(aa')$.

The element r *realizes* the morphism γ. Composition of morphisms is composition of total relations.

We think of γ as a *simulation* in B of computations in A; the element r is a machine that translates code for an A-program into code for a B-program.

Examples of applicative morphisms
$\delta_1 : \mathcal{K}_1 \to A$: $\delta_1(n) = \{\bar{n}\}$ is the essentially unique applicative morphism $\mathcal{K}_1 \to A$ (up to a suitable notion of isomorphism of applicative morphisms)
$\delta_2 : \mathcal{K}_2^{\mathrm{rec}} \to \mathcal{K}_1$: $\delta_2(\phi) = \{e \in \mathbb{N} \,|\, \phi = \varphi_e\}$. Think of what a realizer of this morphism does; how it simulates the action of $\mathcal{K}_2^{\mathrm{rec}}$ in $\mathcal{K}_1$!
There are interesting applicative morphisms between $\mathcal{K}_2$ and $\mathcal{S}$ in both directions.

Computations in PCAs with an oracle

Let $\gamma : A \to B$ be an applicative morphism. A partial function $f : A \rightharpoonup A$ is *representable* w.r.t. γ if there is an element $b \in B$ satisfying: for each $a \in A$, if $f(a)\downarrow$ then $b\gamma(a) \subseteq \gamma(f(a))$.
Theorem (vO 2006): Given PCA A and partial function f on A, there is a PCA $A[f]$ which is universal with the property that there is a decidable applicative morphism $\iota_f : A \to A[f]$ w.r.t which f is representable: if $\gamma : A \to B$ is decidable and f is representable w.r.t. γ, then γ factors uniquely through ι_f:

$$\begin{tikzcd}
A \arrow[r, "\iota_f"] \arrow[dr, "\gamma"'] & A[f] \arrow[d] \\
 & B
\end{tikzcd}$$

Applying this construction to $\mathcal{K}_1$ gives us the PCA of "computations with oracle f".

Note, that this construction gives us a notion of "Turing reducibility in A": if f and g are partial functions on A, then $f \leq_T g$ if and only if f is representable w.r.t. $\iota_g : A \to A[g]$. Equivalently: for every decidable applicative morphism $A \xrightarrow{\gamma} B$ we have: if g is representable w.r.t. γ, then so is f.

An extension of the "oracle" result (Faber/vO 2016)
Given a PCA A, we can define what we call an "effective operation of type 2" in A, and we have, for any partial function $F : A^A \rightharpoonup A$ a similar universal solution for "forcing F to be an effective operation": a decidable applicative morphism $\iota_F : A \to A[F]$ with the expected universal property.
We have the following result (which should not come unexpected): For the Kleene functional E ($E(f) = 0$ if and only if $\exists n f(n) = 0$) we have: a function $\mathbb{N} \to \mathbb{N}$ is representable w.r.t. $\mathcal{K}_1[E]$ if and only if the function f is hyperarithmetical.
This opens up the possibility of "realizability with hyperarithmetical functions"; this is a sheaf subtopos of the effective topos in which there is a model of Peano Arithmetic (with classical logic!). Such a model cannot exist in the effective topos.

In a recent paper, Jetze Zoethout takes this one step further. He explains why a straightforward extension to "third-order functionals" is not to be expected; however, employing a "lax" version of PCAs (the equations hold "up to inequality") one can obtain, for such a PCA A and third-order Φ, a PCA $A[\Phi]$ enjoying a weaker universal property.

Realizability

I restrict myself to the PCA $\mathcal{K}_1$, although a lot of it generalizes to arbitrary PCAs.

For any set X, we have a preorder structure on the set $\mathcal{P}(\mathbb{N})^X$ of functions from X to the powerset of $\mathbb{N}$:

$\phi \leq \psi$ iff there is e such that for all $x \in X$ and all $n \in \phi(x)$, $en\downarrow$ and $en \in \psi(x)$.

We call this the *realizability preorder*. It is a *Heyting prealgebra*: we have

$$\begin{array}{rcl} \phi \wedge \psi(x) & = & \{\langle n,m\rangle \mid n \in \phi(x), m \in \psi(x)\} \\ \phi \to \psi(x) & = & \{e \mid \forall n \in \phi(x) en \in \psi(x)\} \end{array}$$

and we have similar definitions of $\vee$, $\top$, $\bot$, $\exists$, $\forall$. So we have a realizability interpretation of first-order intuitionistic logic.

A function $\beta : \mathcal{P}(\mathbb{N}) \to \mathcal{P}(\mathbb{N})$ is monotone if the statement

$$\forall p, q \in \mathcal{P}(\mathbb{N})(p \to q) \to (\beta(p) \to \beta(q))$$

is true under the realizability interpretation.
The function β is a *Lawvere-Tierney topology* if moreover it satisfies:

$$\begin{array}{l} \forall p(p \to \beta(p)) \\ \forall p, q(\beta(p) \wedge \beta(q) \to \beta(p \wedge q)) \\ \forall p(\beta(\beta(p)) \to \beta(p)) \end{array}$$

Again, in the realizability interpretation. Note: if we define $\neg p$ to be $p \to \emptyset$, so

$$\neg\neg(p) = \begin{cases} \mathbb{N} \text{ if } p \neq \emptyset \\ \emptyset \text{ otherwise} \end{cases}$$

then $\neg\neg$ is an example of a Lawvere-Tierney topology.

But, the structure of Lawvere-Tierney topologies is way more complicated!
Hyland proved: there is an embedding of the Turing degrees into the set of Lawvere-Tierney topologies.
Together with Sori Lee, I investigated further structure.
Very nice recent work has been done by Takeyuki Kihara.

As a warming up, consider *partial multifunctions* on $\mathbb{N}$: these are partial, multi-valued functions $f :\subseteq \mathbb{N} \rightrightarrows \mathbb{N}$.
We have a notion of *Turing reducibility* between these, which can be described in terms of a game between players Merlin and Arthur. Given two partial multifunctions f, g, the game $G(f, g)$ proceeds as follows:
Merlin starts by calling a number x_0. At the n-th round, Arthur has a choice: either he responds $y_n = \langle 0, u_n \rangle$, or he responds $y_n = \langle 1, v_n \rangle$ in which case the game is over.

$$\begin{array}{lcccccccc} \text{Merlin} & x_0 & & x_1 & & x_2 & & \cdots \\ \text{Arthur} & & y_0 & & y_1 & & y_2 & \cdots \end{array}$$

The *rule of the game is*: Merlin's first bid, x_0, must be an element of the domain of definition of f. If Arthur's n-th move is $\langle 0, u_n \rangle$ then u_n must be in the domain of g, and Merlin's response x_{n+1} must be an element of $g(u_n)$. If Arthur terminates the game with $\langle 1, v_n \rangle$, then we must have $v_n \in f(x_0)$.
Arthur *wins the game* if he can successfully terminate in the sense above, or Merlin is not able to comply with the rule.
Definition. The partial multifunction f is Turing reducible to g, $f \leq_T g$, if Arthur has a winning strategy for the game G(f,g).
Examples: if f is the empty function then $f \leq_T g$ for all g: Merlin cannot make a legal opening move. If g is such that for some n, $g(n) = \emptyset$, then Arthur wins by putting $y_0 = \langle 0, n \rangle$, and Merlin is unable to respond.

Now we consider what we call *bilayer functions*: partial multifunctions $f :\subseteq \mathbb{N} \times \Lambda \rightrightarrows \mathbb{N}$ for an arbitrary set Λ. The elements of Λ are thought of as "secret inputs". The *public domain* of f, $\mathrm{dom}_{\mathrm{pub}}(f)$ is the set of those numbers n for which there exists $c \in \Lambda$ such that $(n, c) \in \mathrm{dom}(f)$.
Given two bilayer functions f, g, we have a game $G(f, g)$ between 3 players: Merlin, Arthur and Nimue. Merlin and Nimue are creatures from the underworld and have access to secret information. Arthur is human and sees only the public part. However, Nimue is a benign nymph who helps Arthur in his quest. Merlin starts by calling (x_0, c_0) (Arthur sees only x_0). At te n-th round, Arthur plays either $\langle 0, u_n \rangle$ or $\langle 1, u_n \rangle$ in which latter case the game is over.
Nimue, at the n-th round (and the game is not over, so Arthur's move was $\langle 0, u_n \rangle$), plays $z_n \in \Lambda$ (again, not observable by Arthur). At his $n + 1$-st round, Merlin responds by playing x_{n+1}.

Merlin	(x_0, c_0)			x_1			
Arthur		y_0			y_1		$\cdots$
Nimue			z_0			z_1	

Rules: Merlin's first bid, (x_0, c_0) must be in $\mathrm{dom}(f)$.
If Arthur responds $y_n = \langle 0, u_n \rangle$ then we must have $u_n \in \mathrm{dom}_{\mathrm{pub}}(g)$, so for some $z \in \Lambda$, $(u_n, z) \in \mathrm{dom}(g)$.
If Arthur responds $\langle 1, u_n \rangle$ then $u_n \in f(x_0, c_0)$.
If Arthur has played $\langle 0, u_n \rangle$ then Nimue chooses $z_n \in \Lambda$ satisfying $(u_n, z_n) \in \mathrm{dom}(g)$.
On his $n + 1$-st round, Merlin picks $x_{n+1} \in g(u_n, z_n)$.
Arthur and Nimue *win the game* if either Merlin is unable to comply with the rule, or Arthur terminates with $u_n \in f(x_0, c_0)$.

Definition. For two bilayer functions f, g: $f \leq_T g$ (f is Turing reducible to g) if Arthur/Nimue have a winning strategy for the game $G(f, g)$.
We have a preorder of bilayer functions and Turing reducibility.
Theorem (Kihara) The preorder of bilayer functions and Turing reducibility is equivalent to the preorder of Lawvere-Tierney topologies.

Thanks