

Introduction to abstract computability - Tutorial at MFPS 24

Pieter J. W. Hofstra
 Dept. of Mathematics and Statistics, University of Ottawa,
 Ottawa, K1N 6N5, Ontario, Canada
 email : phofstra@uottawa.ca

May 21, 2008

1 Basic Theory

Definition 1.1 (Restriction category). A *restriction category* is a category \mathbf{C} endowed with a combinator $\bar{(-)}$, sending $f : A \rightarrow B$ to $\bar{f} : A \rightarrow A$, such that the following axioms are satisfied.

R.1	$f\bar{f} = f$	
R.2	$\overline{f\bar{g}} = \bar{g}\bar{f}$	whenever $dom(f) = dom(g)$
R.3	$\overline{g\bar{f}} = \bar{g}\bar{f}$	whenever $dom(f) = dom(g)$
R.4	$\overline{\bar{g}f} = \bar{f}g\bar{f}$	whenever $cod(f) = dom(g)$

Enrichment. Restriction categories are locally ordered: given parallel maps $f, g : A \rightarrow B$, say that

$$f \leq g \Leftrightarrow f = g\bar{f}.$$

Partial Products. An object 1 in a restriction category \mathbf{C} is said to be a *restriction terminal object* if for each object A there is a unique total map $A \xrightarrow{!_A} 1$, such that $!_1 = 1$; moreover, this family of maps must satisfy the condition that for each $A \xrightarrow{f} B$ we have $!_B f = !_A \bar{f}$, as in the diagram below.

$$\begin{array}{ccc} A & & \\ \downarrow f & \searrow !_A & \\ B & \xrightarrow{!_B} & 1 \end{array}$$

A *partial product* of two objects A, B is an object $A \times B$ equipped with total projections $A \times B \xrightarrow{\pi_A} A$ and $A \times B \xrightarrow{\pi_B} B$, such that for each C and each pair of maps $C \xrightarrow{f} A, C \xrightarrow{g} B$, there is a unique map $\langle f, g \rangle : C \rightarrow A \times B$ with the properties that $\pi_A \langle f, g \rangle \leq f, \pi_B \langle f, g \rangle \leq g$ and $\langle f, g \rangle = \bar{f}\bar{g}$.

$$\begin{array}{ccccc} & & C & & \\ & \swarrow f & \downarrow \langle f, g \rangle & \searrow g & \\ A & \xleftarrow{\pi_A} & A \times B & \xrightarrow{\pi_B} & B \end{array}$$

Definition 1.2 (Cartesian restriction categories and functors).

- (i) A restriction category is called *cartesian* if it has all binary partial products as well as a restriction terminal object.

- (ii) A functor is a *restriction functor* F if $F(\bar{f}) = \overline{F(f)}$ for all maps f .
- (iii) A restriction functor F is a *cartesian restriction functor* (or simply a *cartesian functor*) when the canonical comparison morphisms $\langle F\pi_A, F\pi_B \rangle : F(A \times B) \rightarrow FA \times FB$ and $!_{F1} : F1 \rightarrow 1$ are isomorphisms.

Example: *global sections*: given any cartesian restriction category \mathbf{C} , $\mathbf{C} \xrightarrow{\Gamma} \mathbf{Par}$ is defined by

$$\Gamma(C) = \mathbf{Tot}(C)[1, C],$$

the collection of total points of C .

Turing Categories.

Definition 1.3 (Turing category). Let \mathbf{C} be a cartesian restriction category.

- (i) Given a morphism $\tau_{X,Y} : A \times X \rightarrow Y$, a morphism $f : Z \times X \rightarrow Y$ is said to *admit a $\tau_{X,Y}$ -index* when there exists a total map $h : Z \rightarrow A$ for which the following diagram is commutative:

$$\begin{array}{ccc} A \times X & \xrightarrow{\tau_{X,Y}} & Y \\ \uparrow & \nearrow f & \\ h \times X & & \\ \downarrow & & \\ Z \times X & & \end{array}$$

In this case, h is called a $\tau_{X,Y}$ -index for f .

- (ii) A morphism $\tau_{X,Y}$ is called a *universal application* when every $f : Z \times X \rightarrow Y$ admits a $\tau_{X,Y}$ -index. We will also simply say that $\tau_{X,Y}$ is *universal*.
- (iii) A *Turing object* in \mathbf{C} is an object A such that for each $X, Y \in \mathbf{C}$, there is a universal application $\tau_{X,Y} : A \times X \rightarrow Y$.
- (iv) The category \mathbf{C} is called a *Turing category* if it possesses a Turing object.

Remark 1.4.

- We may think of the universal applications as weak evaluations. Thus, a Turing category is cartesian closed in a weak sense.
- Turing structure is not unique and is not part of the data.
- Any Turing object is universal; in particular, it admits an internal pairing operation.
- Turing structure may be generated by specifying a universal object and a universal self application map $A \times A \xrightarrow{\bullet} A$.
- Recursion theoretically, the axioms may be interpreted as enforcing the universality theorem and the parameter theorem.
- The class of Turing categories is stable under idempotent splitting.

The key example is the category $\mathbf{Comp}(\mathbb{N})$: the objects are the finite powers of the natural numbers $1, \mathbb{N}, \mathbb{N}^2, \dots$. Arrows are (tuples of) partial recursive functions.

Coherence. Suppose we have two Turing objects A, B , with specified Turing morphisms $\bullet_A = \tau_{A,A}$ and $\bullet_B = \sigma_{B,B}$. Choose a retraction $(m, r) : A \triangleleft B$.

Lemma 1.5. *If (A, \bullet_A) and (B, \bullet_B) are both Turing objects, then there are codes $u : 1 \rightarrow A$ and $v : 1 \rightarrow B$, and morphisms $m : A \rightarrow B$ and $n : B \rightarrow A$ such that the following diagrams commute:*

$$\begin{array}{ccc} A \times A & \xrightarrow{\bullet_A} & A \\ \langle u, m, m \rangle \downarrow & & \downarrow m \\ B \times B \times B & \xrightarrow{\bullet_B} & B \end{array} \qquad \begin{array}{ccc} B \times B & \xrightarrow{\bullet_B} & B \\ \langle v, n, n \rangle \downarrow & & \downarrow n \\ A \times A \times A & \xrightarrow{\bullet_A} & A \end{array}$$

Equationally, this means

$$u \bullet m(x) \bullet m(y) = m(x \bullet y), \quad v \bullet n(x) \bullet n(y) = n(x \bullet y).$$

Padding. Given a Turing morphism $A \times A \xrightarrow{\bullet} A$ define \bullet' as

$$A \times A \xrightarrow{r \times A} (A \times A) \times A \xrightarrow{\pi_0 \times A} A \times A \xrightarrow{\bullet} A.$$

Here, we have chosen an embedding-projection pair $(m, r) : A \times A \triangleleft A$. Now \bullet' is again a Turing morphism and has the property that every index is a section. (Recursion theoretically: the S-m-n maps are injective.)

2 A bit of recursion theory

Universality, Enumeration, Parameterization. These are built into the setting.

M-completeness. For domains e on X and e' on Y we say that $e \leq_m e'$ (in words: e many-one reduces to e') when there is a total map $f : X \rightarrow Y$ such that $e = f^*(e')$, i.e. e is the pullback of e' along f . A domain is said to be m-complete if every other domain m-reduces to it.

Proposition 2.1. *The domain of a Turing morphism $A \times A \xrightarrow{\bullet} A$ is M-complete.*

We may define a family of iterated application morphisms $\bullet^{(n)} : A \times A^n \rightarrow A$. These are defined inductively: $\bullet^{(1)} = \bullet$, and if we have defined $\bullet^{(n)}$, then let $\bullet^{(n+1)}$ be the composite

$$A \times A \times A^n \xrightarrow{\bullet \times 1} A \times A^n \xrightarrow{\bullet^{(n)}} A.$$

We define $\bullet^{(0)}$ to be the composite

$$A \xrightarrow{\Delta} A \times A \xrightarrow{\bullet} A.$$

Proposition 2.2. *Each of the derived morphisms $\bullet^{(n)}$ has M-complete domain.*

Corollary 2.3. *The halting set is M-complete.*

Undecidability. We need non-trivial partiality. Say that a domain e is decidable if it is a complemented element in the lattice of domains.

Theorem 2.4. *In a Turing category with zero, the halting problem is undecidable.*

Creativity and Effective Non-Recursiveness. Classically, both of these notions coincide with M-completeness. In a Turing category with zero, effective non-recursiveness coincides with M-completeness but creativity is strictly weaker.

Representations of r.e. sets. In order to obtain various equivalent characterizations of recursive sets and r.e. sets we need even more structure (coproducts, ranges, choice, ...)

3 Partial Combinatory Algebras

Definition 3.1 (Applicative system). An *applicative system* $\mathbb{A} = (A, \bullet)$ in \mathbf{C} consists of an object A together with a morphism $A \times A \xrightarrow{\bullet} A$, called *application*.

Definition 3.2 (Computable morphism, absolute version). Let $\mathbb{A} = (A, \bullet)$ be an applicative system.

- (i) For $n \geq 0$, a morphism $f : A^n \rightarrow A$ is said to be \mathbb{A} -*computable* (or *computable* whenever \mathbb{A} is clear from the context) when there exists a total point $p : 1 \rightarrow A$ such that

$$\begin{array}{ccc} A \times A^n & \xrightarrow{\bullet} & A \\ p \times 1 \uparrow & \nearrow f & \\ A^n & & \end{array}$$

is commutative. (We suppress the isomorphism $A^n \cong 1 \times A^n$). Moreover, if $n > 1$, it is required that the morphism

$$1 \times A^{n-1} \xrightarrow{p \times 1} A \times A^{n-1} \xrightarrow{\bullet^{n-1}} A$$

is total. In this situation, we call p a *code* for the morphism f .

- (ii) More generally, a morphism $A^n \rightarrow A^m$ is computable when all components are; finally, we say that a morphism $A^n \rightarrow 1$ is computable if its domain is (as a map $A^n \rightarrow A^n$).

Lets the \mathbb{A} -polynomial category be the smallest cartesian restriction subcategory of \mathbf{C} on the objects $1, A, A^2, \dots$ which contains all total points of A as well as the application map. Morphisms in this subcategory are called \mathbb{A} -*polynomial morphisms*.

Definition 3.3 (Combinatory Completeness, Partial Combinatory Algebra). An applicative system is *combinatory complete* when every polynomial morphism is computable. A *Partial Combinatory Algebra* (PCA) is a combinatory complete applicative system.

Theorem 3.4 (Categorical characterization of completeness). *For an applicative system $\mathbb{A} = (A, \bullet)$ in a category \mathbf{C} , the following are equivalent:*

- (i) \mathbb{A} is combinatory complete
- (ii) The \mathbb{A} -computable morphisms form a cartesian restriction category over \mathbf{C} .

Theorem 3.5 (Structure theorem for Turing categories).

1. When $\mathbb{A} = (A, \bullet)$ is a PCA in \mathbf{D} , then $\text{Comp}(\mathbb{A})$ is a Turing category with Turing object (A, \bullet) .
2. If \mathbf{D} is a Turing category and (A, \bullet) is a Turing object, then $\mathbb{A} = (A, \bullet)$ is a PCA in \mathbf{D} and $\mathbf{D} \simeq \mathcal{K}_E(\text{Comp}(\mathbb{A}, \mathbf{D}))$ for some class E of idempotents.

4 Simulations

Definition 4.1 (Simulation of PCAs). Let $\mathbb{A} = (A, \bullet_A), \mathbb{B} = (B, \bullet_B)$ be PCAs and let $\phi : A \rightarrow B$ be a morphism on the underlying objects.

- (i) Given an \mathbb{A} -computable morphism $f : A^n \rightarrow A$, we say that f is ϕ -computable if there exists a \mathbb{B} -computable morphism $h : B^n \rightarrow B$ such that the following diagram commutes:

$$\begin{array}{ccc} A^n & \xrightarrow{f} & A \\ \phi^n \downarrow & & \downarrow \phi \\ B^n & \xrightarrow{h} & B \end{array}$$

In the above diagram, we say that h (strictly) simulates f .

- (ii) The map $\phi : A \rightarrow B$ is a (strict) simulation when every \mathbb{A} -computable f is ϕ -computable.

Lemma 4.2. For PCAs $\mathbb{A} = (A, \bullet_A), \mathbb{B} = (B, \bullet_B)$ and a morphism $\phi : A \rightarrow B$ the following are equivalent:

- (i) ϕ is a simulation;
(ii) there exists a code $u : 1 \rightarrow B$ with the property that

$$\begin{array}{ccc} A^2 & \xrightarrow{\bullet} & A \\ u \times \phi \times \phi \downarrow & & \downarrow \phi \\ B^3 & \xrightarrow{\bullet} & B \end{array}$$

commutes.

We denote by $\mathfrak{PC}\mathfrak{A}(\mathcal{C})$ the category of PCAs and simulations in the category \mathcal{C} ; this category is preorder-enriched. For two parallel simulations $\phi, \psi : \mathbb{A} \rightarrow \mathbb{B}$, we define $\phi \vdash \psi$ if and only if there exists a code w such that the following diagram commutes:

$$\begin{array}{ccc} A & & \\ \langle w, \phi \rangle \downarrow & \searrow \psi & \\ B \times B & \xrightarrow{\bullet} & B \end{array}$$

In particular, we can consider adjunctions and equivalences in this category.

Theorem 4.3. Let \mathcal{C} be a split category and let \mathbb{A} and \mathbb{B} be PCAs in \mathcal{C} . Then \mathbb{A} and \mathbb{B} are equivalent as PCAs if and only if the Turing categories $\text{Comp}(\mathbb{A})$ and $\text{Comp}(\mathbb{B})$ are Morita-equivalent as categories.