Effective Applicative Structures

Andrea Asperti and Agata Ciabattoni

Dipartimento di Matematica
P.zza di Porta S.Donato 5, Bologna, Italy
{asperti,ciabatto}@cs.unibo.it

Abstract. An Effective Applicative Structure is a collection of partial functions over an arbitrary set M, indexed by elements of the same set, closed under composition, and containing projections, universal functions and functions S_n^m of the s-m-n theorem of Recursion Theory. The notion of EAS is developed as an abstract approach to computability, filling a notational gap between functional and combinatorial theories.

1 Introduction

Suppose having a set of indexed partial functions that is closed under composition, contains all projections and an interpreter, and satisfies the s-m-n theorem of Recursion Theory.
The question is: is it Turing complete?
The problem looks very natural: closure under composition and existence of projections are obvious properties of effective functions, while the s-m-n theorem and the existence of universal functions are basic results of the theory of effective computability. Many interesting results (such as Kleene's fixed point theorem) can be proved by the only use of the previous assumptions.
Surprisingly, it seems that the previous problem has never been really addressed in existing literature. Indeed, there are several similar "axiomatic" approaches to Recursion Theory, but all of them make stronger assumptions than ours. The most relevant (and closest) approach is the Basic Recursive Function Theory introduced by Wagner and Strong. The essential difference w.r.t. this system is that we drop constants and definition by cases (that is a quite restrictive and somewhat "integer-oriented" assumption).
As a matter of fact, constants and definition by cases are completely inessential to establish the (Turing) completeness of the system: the above assumptions (we call it an Effective Applicative Structure) are enough.
The result is obtained by proving that any EAS is actually a Partial Combinatory Algebra (PCA). By the Turing completeness of the latter (see [5]), we derive the Turing completeness of the former. Moreover, we have another interesting corollary, that was not evident a priori: it is easy to see that any PCA is an Effective Applicative Structure, so we have a *complete equivalence* between EAS and PCA.
In this way we obtain an alternative (and, in our opinion more natural and elegant) characterization of PCA's, which, in some sense, have always been the

"poor relations" of Combinatory Algebras: they lack the simplicity and formal elegance of the latters, while inheriting all their weak points: a pretty "operational" nature (that contrasts with the "denotational" idea of partiality), and a large arbitrarity in the choice of the basic combinators (S and K).

2 Effective Applicative Structures

An Effective Applicative Structure (EAS) is a family Φ^n of functions

$$\Phi^n : M \to \underbrace{(M^n \to M)}_{partial\ functions} \qquad n \in \mathcal{N}$$

over an arbitrary set M, which satisfies the following axioms[1]:

1. it is closed under composition, i.e.
   $\forall s, a_1, \ldots a_r \in M$ and $\forall r, i \in \mathcal{N}$, $\exists f \in M$ such that
   $$\Phi^r_s(\Phi^i_{a_1}(x_1, \cdots, x_i), \cdots \Phi^i_{a_r}(x_1, \cdots, x_i)) \simeq \Phi^i_f(x_1, \cdots, x_i)$$
   $\forall (x_1, \ldots, x_i) \in M^i$

2. it contains all projection functions I^n_i, i.e.
   $\forall i, n \in \mathcal{N}$, $\exists k \in M$ such that
   $$I^n_i \simeq \Phi^n_k$$

3. it contains the S^m_n functions of the s-m-n theorem, i.e.
   $\forall m, n \in \mathcal{N}$ $\exists j \in M$ such that Φ^{n+1}_j is total[2] and
   $$\Phi^m_{\Phi^{n+1}_j(i, x_1, \cdots, x_n)}(y_1, \cdots, y_m) \simeq \Phi^{m+n}_i(x_1, \cdots, x_n, y_1, \cdots, y_m)$$
   with $(x_1, \cdots, x_n) \in M^n$
   and $(y_1, \cdots, y_m) \in M^m$

4. it contains interpreters, i.e.
   $\forall r \in \mathcal{N}$, $\exists i \in M$ such that
   $$\Phi^{r+1}_i(x, y_1, \cdots, y_r) \simeq \Phi^r_x(y_1, \cdots, y_r)$$
   $\forall x \in M, \forall (y_1, \ldots, y_r) \in M^r$

In this paper we prove that any EAS is Turing complete, i.e. all recursive functions are representable in it.
An obvious example of EAS is the collection of (n-ary) recursive functions over naturals. However, we have very simple Effective Applicative Structures whose

[1] according to the standard notation of Recusion Theory, we write Φ^n_i instead of $\Phi^n(i)$.
[2] $\Phi^{n+1}_j = S^m_n$

domains is different from $\mathcal{N}$. An interesting case derives from λ-calculus: let $M \equiv \Lambda/\beta$ (i.e. the set of λ-terms modulo β-equality) and let

$$\phi^n_P(N_1, \cdots, N_n) = PN_1 \cdots N_n$$

then $\{\phi^n\}_{n \in M}$ is an EAS model.
In fact

1. $\{\phi^n\}$ satisfies closure under composition, i.e.
$\forall M_1, M_2, \cdots, M_l, \quad \forall n \in \mathcal{N}$
$M_i \in M \quad \forall i = 1 \cdots l \quad l \in \mathcal{N}$, $\exists F \in M$:

$$\phi_{M_1}(\phi_{M_2}(N_1, \cdots, N_n) \cdots \phi_{M_l}(N_1, \cdots, N_n)) = \phi_F(N_1, \cdots, N_n)$$

2. $\{\phi^n\}$ contains projections, i.e.

$$\phi^k_R(N_1, \cdots, N_k) = N_i$$

with $R \equiv \lambda x_1 \cdots x_k.x_i$
3. $\{\phi^n\}$ contains the S^m_n functions of the s-m-n theorem, i.e.
$\forall m, n \in \mathcal{N}\ \exists Q \in M$ such that

$$\phi^m_{\phi_Q(P,N_1,\cdots,N_n)}(M_1, \cdots, M_m) = \phi^{m+n}_P(N_1, \cdots, N_n, M_1, \cdots, M_m)$$

with $Q \equiv \lambda x_1, \cdots x_{n+1}.x_1x_2 \cdots x_{n+1}$
4. $\{\phi^n\}$ contains interpreters, i.e. $\forall n \in \mathcal{N} \exists U \in M$ such that

$$\phi_U(M, N_1, \cdots, N_r) = \phi_M(N_1 \cdots N_r)$$

with $U \equiv \lambda xy_1 \cdots y_r.xy_1 \cdots y_r$

Then $\{\phi^n\}_{n \in M}$ is an EAS model.

3 Other axiomatic approaches

The problem of defining an axiomatic approach to Recursion Theory has been previously investigated by several authors [20, 21, 19, 9, 7, 8, 14, 17].
In particular, the Basic Recursive Function Theory (BRFT) of [19] is closely related to EAS (let us recall that BRFT characterises the families of functions which form Uniformly Reflexive Structures [20, 21]).

Definition 1. *A BRFT (Basic Recursive Function Theory) is a structure (D, F, $(\phi^n)_{n \in \mathcal{N}}$) satisfying:*

- *D is an infinite set*

- *F is a collection of partial functions on D, such that:*

  1. *it is closed under composition*

2. *it contains all projection functions*
3. *it contains all constant functions on D*
4. *it contains the function for definition by cases*

$$f(x,a,b,c) = \begin{cases} b & \text{if } x = a \\ c & \text{otherwise} \end{cases}$$

5. *it contains the S^m_n function of the s-m-n theorem*
6. *it contains an interpreter*

Proposition 1. *All total recursive functions can be represented in any BRFT.*

Essentially, in EAS we drop constants and definition by cases. We shall prove that they are not essential for *representing* partial recursive functions.
A main consequence of having definition by cases is that application is *eventually* a partial operation. This can be proved by a simple diagonal argument. Take two distinct elements a and b and consider the function

$$f(x) = \begin{cases} a & \text{if } \phi_x(x) = b \\ b & \text{otherwise} \end{cases}$$

Since $f \in F$ there exists an index c such that $f = \phi_c$. Supposing $\phi_c(c)$ defined we would get a contradiction.
Another interesting structure is the ω-BRFT, that is a BRFT ($\mathcal{N}$, F, $(\phi^n)_{n\in\mathcal{N}}$) where $\mathcal{N}$ is the set of all natural numbers, and F *contains* the successor function. In this case we can prove something stronger, namely:

Proposition 2. *If ($\mathcal{N}$, F, $(\phi^n)_{n\in\mathcal{N}}$) is an ω-BRFT, then F* contains *all the recursive functions.*

Still we cannot prove that F contains *exactly* the partial recursive functions [7].

4 Turing completeness of PCA's

A *Partial Applicative Structure* is a pair $(A, \cdot)$, where A is an arbitrary set and $\cdot$ is a partial binary operation over A, called application.

Notation

– Instead of $a \cdot b$ we write ab; moreover, we conventionally suppose that application is left associative.
– $ab\downarrow$ means "ab is defined"
  $ab\uparrow$ means "ab is not defined"
– If t_1, t_2 are applicative expressions, $t_1 \simeq t_2$ abbreviates $t_1\downarrow \vee\, t_2\downarrow \Rightarrow t_1 = t_2$.

Definition 2. *A* Partial Combinatory Algebra *(PCA) is a structure $\mathcal{A} = (A, S, K, \cdot)$, where $(A, \cdot)$ is a partial applicative structure and S and K are two distinguished elements of A that satisfy the following conditions, for all $a, b, c \in A$:*

1. $Ka\downarrow$, $Sa\downarrow$, $Sab\downarrow$
2. $Kab \simeq a$
3. $Sabc \simeq ac(bc)$

Proposition 3. *In any PCA*

1. $ac(bc)\downarrow \Rightarrow Sabc\downarrow$;
2. $a\downarrow \wedge Kab\uparrow \Rightarrow b\uparrow$.

Let Q be a PCA, and let us fix some numeral system in Q, that is an interpretation (numeral) $\overline{m}$ for each integer m .

Definition 3. *Let f be a partial function from $\mathcal{N}$ to $\mathcal{N}$. The element e of a PCA Q is said to* numerically represent f *if:*

$$Q \models e\overline{m} = \overline{k} \iff k = f(m).$$

Theorem 1. *All partial recursive functions are numerically representable in any PCA Q.*

Proof. See Beeson [5].

We shall discuss Beeson's proof in some detail in the following, since it seems to contain some (minor) problems. First of all, it is rather indirect. Beeson uses two auxiliary theories PCA$^+$ and EON which are extensions of PCA with pairing, numbers, definition by cases (PCA$^+$), and induction scheme (EON). Then he resorts to a subtle model-theoretic argument, involving Kripke models, to overcome the difficulty of interpreting the predicate $N(x)$ of PCA$^+$ (representing integers) inside the PCA (it cannot be interpreted as the set of numerals, since it is not defined by a formula in PCA). Actually, it is not difficult to provide a more direct argumentation. However, the main problem with Beeson's proof is of a different nature. All along the proof, he makes a large (and in our opinion somewhat arbitrary) use of a λ-notation. His encoding of λ-abstraction into combinators is based on the following definition:

Definition 4. *1.* $\lambda^\circ x.x \equiv SKK$
2. $\lambda^\circ x.M \equiv KM$ *if M is a variable $y \neq x$, S or K.*
3. $\lambda^\circ x.(MN) \equiv S(\lambda^\circ x.M)(\lambda^\circ x.N)$

A major consequence of this definition is that for any combinatorial expression M, $\lambda^\circ x.M\downarrow$. This property is heavily used by Beeson, since it avoids several annoying problems. However, $\lambda^\circ x.M$ has a different and well known problem, completely ignored in [5]: it does not satisfy the substitution lemma (see remark 2.16 in [10]). That is, in general,

$$(\lambda^\circ x.M)[N/y] \neq \lambda^\circ x.(M[N/y])$$

As a consequence, Beeson's free and easy use of the λ-notation does not seem to be completely justified. Consider for instance the recursion theorem 2.7 (see [5] p.103), reported below with its proof:

Theorem 2. *There is a term* **R** *such that PCA proves:*

$$\mathbf{R}f\downarrow \;\wedge\, [g \;=\; \mathbf{R}f \;\Rightarrow\; \forall x(gx \;\simeq\; fgx)]$$

Proof. Take **R** to be $\lambda^\circ f.tt$ where $t = \lambda^\circ yx.f(yy)x$. Then $g \simeq tt \simeq \lambda^\circ x.f(tt)x$, so $gx \simeq fgx$.

Actually, $g \simeq tt \simeq (\lambda^\circ x.f(yy)x)[t/y] \not\simeq \lambda^\circ x.f(tt)x$. On the other side, the fact that $\mathbf{R}f\downarrow$ is a consequence of $\lambda^\circ x.f(yy)x\downarrow$, that is *essentially* based on the definition of λ°.
These are indeed minor details, and they can be fixed in several simple ways. Our sketch of solution in the following is essentially used as a pretext to present a few results on representability of λ-terms in partial combinatory algebras.

4.1 Representability of λ-terms in PCA

Let us consider the "usual" encoding function $[\quad]$ from λ-calculus to Combinatory Logic.

Definition 5. $[\quad] : \Lambda \to CL$ *is defined as follows:*

1. $[x] \equiv x$
2. $[MN] \equiv [M]\,[N]$
3. $[\lambda x.M] \equiv \lambda^* x.[M]$
   where
   (a) $\lambda^* x.x \equiv SKK$
   (b) $\lambda^* x.M \equiv KM \quad x \notin FV(M)$
   (c) $\lambda^* x.(MN) \equiv S(\lambda^* x.M)(\lambda^* x.N)$

In this way a λ-term M is translated into an applicative expression $[M]$ built up by means of variables, S and K. The problem is to understand when $[M]$ is defined in any PCA, i.e. if $[M]\downarrow$. Note that, in this case, the problem is not only due to rule 2., but also to case (b) of the definition of $\lambda^* x.M$. So, in general, it is not true that $\lambda^* x.M\downarrow$.
Given the translation above, there exists a well known correspondence between weak reduction in the λ-calculus and weak reduction in Combinatory Logic.
Let us recall that the weak reduction strategy $\overset{w}{\to}$ in the λ-calculus is defined by the following rules:

$$(\lambda x.M)N \overset{w}{\to} M[N/x]$$

$$\frac{(M \overset{w}{\to} M_1)}{(MN \overset{w}{\to} M_1 N)}$$

However, in the case of PCA's, the above correspondence must be used very carefully, due to the problem of partiality.

Theorem 3. *Suppose* $P\downarrow$. *Then, for any* M

$$(\lambda^* x.M)\,P \;\simeq M[P/x]$$

Proof. Easy consequence of proposition 3.

Theorem 3 should be correctly understood. In particular it does not imply that for all λ-terms M, N if $M \overset{w}{\rightarrow} N$ and $[N]\downarrow$ then $[M]\downarrow$. A counterexample is easily found. Barendregt (see theorem 1.3 in [4]) proved that there are PCA's in which $[\delta\delta]\uparrow$ (where $\delta \equiv \lambda x.xx$). Take now $M = \lambda x.y\ (\delta\delta)$. Then $M \overset{w}{\rightarrow} y$ and $[y]\downarrow$, but obviously $[M]\uparrow$.

Theorem 4. *Let M be a closed λ-term in normal form. Then $[M]\downarrow$.*

Proof. Easy induction on the structure of M.

Definition 6. *An* applicative *λ-term is any λ-term of the kind $(M\ N)$.*

Proposition 4. *Let $M = \lambda x.P$. Then $[M] = \lambda^* x.[P]\uparrow$ if and only if there exists an* applicative *subterm $Q = (Q_1 Q_2)$ of M (of P) such that*

1. *$x \notin FV(Q)$*
2. *$[Q]\uparrow$*
3. *$[Q]$ is a subexpression of $[M]$*

Proof. By induction on the structure of P.

- Let P be a variable. Then the statement is trivially true, since $\lambda^* x.[P]\downarrow$.
- Let $P = (N_1\ N_2)$. There are two subcases. If $x \notin \mathrm{FV}(P)$ take $Q = P$. Otherwise, $[M] = \lambda^* x.[P] \simeq S\ (\lambda^* x.[N_1])\ (\lambda^* x.[N_2])$, and the statement follows by induction hypothesis.
- Let $P = \lambda y.N$. By induction hypothesis $\lambda y.N\uparrow$ iff there exists an applicative subterm $Q = (Q_1 Q_2)$ of N with the properties above. If $x \notin \mathrm{FV}(Q)$ then Q does the job. Conversely, if $x \in \mathrm{FV}(Q)$ then the application $Q = (Q_1 Q_2)$ is eventually splitted by λ^* into the defined term $S\ (\lambda^* x.[Q_1])\ (\lambda^* x.[Q_2])$

Corollary 1. *Let M be a* closed *λ-term. Then $[M]\uparrow$ if and only if there exists a* closed *applicative subterm Q of M such that $[Q]\uparrow$ and $[Q]$ is a subexpression of $[M]$.*

An interesting consequence of the previous corollary is the following:

Theorem 5. *Let M be a closed strongly normalizable λ-term. Then $[M]\downarrow$.*

Proof. By induction on the length l of the longest normalizing derivation.

- $l = 0$. In this case M is in normal form, and $[M]\downarrow$ by theorem 4.
- By contradiction. Suppose that $[M]\uparrow$. Then by corollary 1 there exists a *closed* applicative subterm Q of M such that $[Q]\uparrow$ and $[Q]$ is a subexpression of $[M]$. Since Q is a closed applicative term,

  $$Q = (\lambda x.P)\, Q_1\, Q_2\, \ldots\, Q_n$$

  and

  $$(\lambda x.P)\, Q_1\, Q_2\, \ldots\, Q_n \overset{w}{\rightarrow} P[Q_1/x]\, Q_2\, \ldots\, Q_n$$

By induction hypothesis $[P[Q_1/x]Q_2\ \dots\ Q_n]\downarrow$, and also $[Q_1]\downarrow$. Then, by theorem 3

$$[(\lambda x.P)\, Q_1\, Q_2 \dots Q_n] \simeq [P[Q_1/x]Q_2\ \dots\ Q_n]$$

that would imply

$$[Q] = [(\lambda x.P)\, Q_1\, Q_2 \dots Q_n]\downarrow$$

Corollary 2. *Let M and M_1 be closed strongly normalizable λ-terms. Then $M \overset{w}{=} M_1 \Rightarrow [M] = [M_1]$*

Let us now come back to the problem of the fixpoint in PCA's.
The term **R** of theorem 2 does not work with λ^*, since we cannot prove that $tt \simeq \lambda^* x.f(tt)x \simeq (\lambda^* x.f(yy)x)[t/y]$ is defined. The problem is due to the auto-application of y inside t, that is not splitted any more by $\lambda^* x$. To force this splitting it is enough to add a "dummy" occurrence of x inside one argument of the application.
Let us define

$$R^* \equiv \lambda f.tt$$

with

$$t = \lambda^* yx.f((Kyx)y)x$$

Then we have

$$g \ \simeq\ tt \ \simeq\ \lambda^* x.f((Ktx)t)x$$

Now, $\lambda^* x.f((Ktx)t)x$ is forcedly defined by corollary 1, since it does not contain any closed applicative subterm Q, such that $[Q]\uparrow$. Indeed the only candidate would be Kt, but t is obviously defined, and so is Kt.
Moreover,

$$\forall x\ \ gx \ \simeq\ ttx \ \simeq\ (\lambda^* x.f((Ktx)t)x)x \simeq f((Ktx)t)x) \simeq f(tt)x \simeq fgx$$

4.2 The numeral system

We can now safely approach the problem of the Turing completeness of PCA's using the comfortable notation of λ-calculus. We shall be rather sketchy here, both for lack of space, and because Beeson's proof, apart some abuse of notation, is essentially correct.
The first problem is to find a numeral system that works well with the weak reduction strategy. Church numerals are not adequate to our purposes; for instance $Succ\ \underline{n}$ does not weakly reduces to $\underline{n+1}$.

Remark 1. It looks that Church numerals could be used by considering a sligtly weaker notion of represantability, namely:

e weakly represents f in a PCA Q if: $Q \models e\overline{m}xy \simeq \overline{k}xy \iff k = f(m)$

A clear advantage of Church numerals is that we could apply theorem 5 to prove that, as far as we consider primitive recursive functions, we do not have problems with partiality. Indeed, all primitve recursive terms are typable in system F, and thus are strongly normalizing. Minimalization would then be an easy step. Unfortunately, with the numeral system below, we have no obvious way of applying theorem 5, and we must proceede to a case by case analysis.

A numeral system that properly works with weak reductions is the following (see Chap.6, Sec.2 in [3]):

Definition 7. *Let* $Pair \equiv \lambda xyz.zxy$*,* $T \equiv \lambda xy.x$*,* $F \equiv \lambda xy.y$*. Then*

- $\underline{0} = I = \lambda x.x$
- $\underline{n+1} = Pair\ F\ \underline{n} = [F, \underline{n}]$

Constants and projections are defined in the obvious way, and the system is naturally closed under composition. Let us see some examples of numerical functions:

- $Succ\ \equiv\ \lambda n.[F, n]$
- $Pred\ \equiv\ \lambda n.nF$
- $Test_0\ \equiv\ \lambda n.nT$

Turing completeness of PCA's can then be proved following the same line of [3] (Chap.6, Sec.3), with the obvious care due to partiality. Representability of basic functions is trivial, and composition does not give any trouble.
A more problematic case is already provided by the primitive recursion schema ($\boldsymbol{n}$ is a vector of variables):

$$f(0, \boldsymbol{n}) = g(\boldsymbol{n})$$

$$f(x+1, \boldsymbol{n}) = h(f(x, \boldsymbol{n}), x, \boldsymbol{n})$$

As a matter of fact, we do not have iterators in this numeral system, and we are forced to use a fixpoint operator. As we have seen, computing a fixpoint is not particularly problematic, but we must care about the correct definition of the functional. In this case, Barendregt's functional derived from the recursive equation

$$fx\boldsymbol{y} = test_0\ x\ (g\boldsymbol{y})\ (h(f(Pred\,x)\boldsymbol{y})x\boldsymbol{y})$$

would not work. Indeed

$$f0\boldsymbol{y} = T\ (g\boldsymbol{y})\ (h(f(Pred\,0)\boldsymbol{y})x\boldsymbol{y})$$

This term is defined if and only if both arguments of T are defined (and we have no way to prove the second one is). A simple solution is to take

$$fx\boldsymbol{y} = (test_0\ x\ (\lambda x\boldsymbol{y}.g\boldsymbol{y})\ (\lambda x\boldsymbol{y}.h(f(Pred\,x)\boldsymbol{y})x\boldsymbol{y})x\boldsymbol{y}$$

A similar care is due when defining the operator μ of minimalization.

4.3 On separability of numerals

One could wonder if the above numeral system is "well defined" in any PCA, in the sense that $\underline{m} \overset{w}{\neq} \underline{n} \iff [\underline{m}] \neq [\underline{n}]$

In fact, from corollary 2 it follows that if $\underline{m} \overset{w}{\neq} \underline{n} \Rightarrow [\underline{m}] \neq [\underline{n}]$

Let us prove the converse.

Definition 8. *For CL-terms, the class of terms in strong normal form (strong nf) is inductively defined as follows:*

- *all non-redex atoms (i.e. variables or constants other than S and K) are in strong nf.*
- *if $X_1 \cdots X_n$ are in strong nf, and a is any non-redex atom, then $aX_1 \cdots X_n$ is in strong nf.*
- *if X is in strong nf, then so is $\lambda^\eta x.X$ which is defined (see def. 2.14 in [10]) by induction on M:*
  - $\lambda^\eta x.M \equiv KM$ *if* $x \notin FV(M)$
  - $\lambda^\eta x.x \equiv I$
  - $\lambda^\eta x.Ux \equiv U$
  - $\lambda^\eta x.UV \equiv S(\lambda^\eta x.U)(\lambda^\eta x.V)$

Actually, it is easy to prove that the translation [ ] of definition 7 over numerals give terms in strong normal form. Then $[\underline{m}] \neq [\underline{n}] \Rightarrow \underline{m} \overset{w}{\neq} \underline{n}$ follows by (the Combinatory Logic reformulation of) Böm's separability theorem (see [6, 11]): let M and N combinators, either in strong normal form; if $M \neq N$, then there exists $n \geq 0$ and combinators $L_1 \cdots L_n$ such that $\forall X, Y$

$$ML_1 \cdots L_n XY = X$$

$$NL_1 \cdots L_n XY = Y$$

5 Equivalence of EAS and PCA

In this section we shall prove that any Partial Combinatory Algebra provides a model of Effective Applicative Structure and, conversely, for any Effective Applicative Structure its domain can be naturally equipped with a PCA-structure.

5.1 From PCA to EAS

Let $\mathcal{M} = (M, S, K, \cdot)$ be a PCA. Let us define a family of (partial) functions $\phi^n : M \to (M^n \to M) \quad n \in \mathcal{N}$ as follows:

$$\phi^n_p(b_1, \cdots, b_n) \simeq pb_1 \cdots b_n$$

(we shall often omit the superscript n when it clear from the context.)

We have that

1. $\{\phi^n\}$ satisfies closure under composition
in fact
$\forall M_1, M_2, \cdots, M_l, \quad \forall n \in \mathcal{N}$
$M_i \in M \quad \forall i = 1 \cdots l \quad l \in \mathcal{N}$, $\exists F \in M$ such that

$$\phi_{M_1}(\phi_{M_2}(N_1, \cdots, N_n) \cdots \phi_{M_l}(N_1, \cdots, N_n)) = \phi_F(N_1, \cdots, N_n)$$

with $F \equiv \lambda^* x_1 \cdots x_n . M_1(M_2 x_1 \cdots x_n) \cdots (M_l x_1 \cdots x_n)$
2. $\{\phi^n\}$ contains projection functions, i.e.

$$\phi^k_R(N_1, \cdots, N_k) = N_i$$

with $R \equiv \lambda x_1 \cdots x_k . x_i$

3. $\{\phi^n\}$ contains the S^m_n function of the s-m-n theorem, i.e.
$\forall m, n \in \mathcal{N}$ $\exists Q \in M$ such that $\phi_Q(P, N_1 \cdots N_n)$ is defined and

$$\phi^m_{\phi_Q(P, N_1, \cdots, N_n)}(M_1, \cdots, M_m) = \phi^{m+n}_P(N_1, \cdots, N_n, M_1, \cdots, M_m)$$

with $Q \equiv \lambda x_1, \cdots x_{n+1} y_1 \cdots y_m . x_1 x_2 \cdots x_{n+1} y_1 \cdots y_m$
4. $\{\phi^n\}$ contains interpreters, i.e. $\forall n \in \mathcal{N} \quad \exists U \in M$ such that

$$\phi_U(M, N_1, \cdots, N_n) = \phi_M(N_1, \cdots, N_n)$$

with $U \equiv \lambda x y_1 \cdots y_n . x y_1 \cdots y_n$

Then $\phi^n : M \to (M^n \to M) \quad n \in \mathcal{N}$ is an Effective Applicative Structure.

5.2 From EAS to PCA

Given an Effective Applicative Structure

$$\Phi^n : M \to (M^n \to M) \quad n \in \mathcal{N}$$

we use the interpreter of condition 4. to define a partial binary operation $\cdot$: $M \times M \to M$:

$$a \cdot b \equiv \Phi^2_u(a, b) \simeq \Phi_a(b)$$

(u is the index of the interpreter with 2 arguments.)
S and K are defined as follows.

– **Existence of K**
We prove the existence of an index K in M such that, for all $i, j \in M$

$$(\Phi_k(i))(j) \simeq i$$

In fact

$$i \simeq I^2_1(i, j)$$

from hypotesis 2. of EAS:

$$\simeq \Phi^2_l(i,j)$$

$$\simeq \Phi^1_{S^1_1(l,i)}(j)$$

from hp. 3. of EAS :

$$\simeq \Phi^1_{\Phi^2_m(l,i)}(j)$$

$$\simeq \Phi^1_{\Phi^1_{S^1_1(m,l)}(i)}(j)$$

Since S^m_n is total, setting $K \equiv S^1_1(m,l)$, we have:

$$\simeq \Phi^1_{\Phi^1_K(i)}(j)$$

that, for definition of application is:

$$\simeq (K(i)) \cdot (j)$$

– **Existence of S**
We prove the existence of an index $S \in M$ such that $\forall a, b, c \in M$

$$((\Phi_S(a))(b))(c) \simeq (ac)(bc)$$

In fact

$$(ac)(bc)$$

from definition of application

$$\simeq \Phi^2_z(\Phi^2_z(a,c), \Phi^2_z(b,c))$$

that is

$$\simeq \Phi^2_z(\Phi^2_z(I^3_1(a,b,c), I^3_3(a,b,c)), \Phi^2_z(I^3_2(a,b,c), I^3_3(a,b,c)))$$

from hp. 1. and 2. of EAS

$$\simeq \Phi^3_h(a,b,c)$$

$$\simeq \Phi^1_{S^2_1(h,a,b)}(c)$$

from hp. 3. of EAS

$$\simeq \Phi^1_{\Phi^3_j(h,a,b)}(c)$$

$$\simeq \Phi^1_{\Phi^1_{S^2_1(j,h,a)}(b)}(c)$$

from hp. 3. of EAS :

$$\simeq \Phi^1_{\Phi^1_{\Phi^3_j(j,h,a)}(b)}(c)$$

$$\simeq \Phi^1_{\Phi^1_{\Phi^1_{S^2_1(j,j,h)}(a)}(b)}(c)$$

since S^m_n is total, setting $S \equiv S^2_1(j,j,h)$:

$$\simeq \Phi^1_{\Phi^1_{\Phi^1_S(a)}(b)}(c)$$

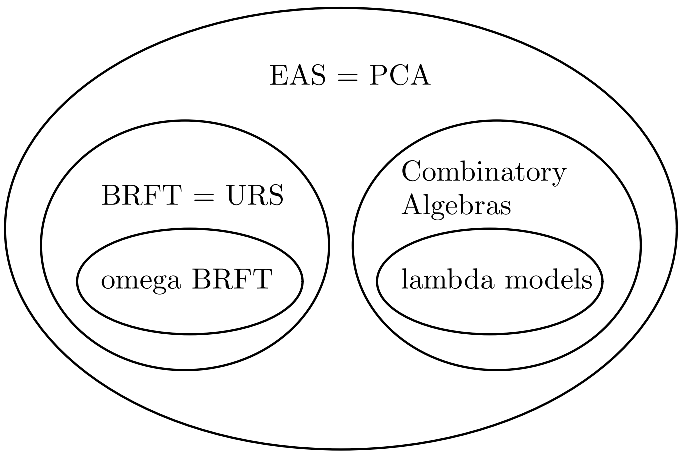
that for definition of application is:

$$(((S(a))b)c)$$

that is the definition of S.
- $Kx\downarrow, Sx\downarrow$ and $(S(x))y\downarrow$ for all $x, y \in M$ since S^m_n is a total function.
- It is easy to prove that if $K = S$ then $|M| = 1$.

6 Conclusions

In this paper we proved that any set of indexed partial functions closed under composition, containing projections and universal function, and satisfying the s-m-n theorem of Recursion Theory is eventually Turing complete (i.e. all partial recursive functions are representable in it). The theory of such systems, that we called Effective Applicative Structures, is a subtheory of the Basic Recursive Function Theory; in particular, we drop constants and definition by cases that are inessential for completeness.
The fact that all partial recursive functions are representable inside any EAS, is proved by establishing an equivalence between EAS's and Partial Combinatory Algebras. In this way, we also fill an evident notational gap between the "combinatorial" and the "functional" approach to computability. By providing a common framework we help in understanding the two opposite developments of these approaches: towards "totality" or "definition by cases". Recall that a BRFT *cannot be* a *total* Combinatory Algebra since application is eventually a partial operation (see section 3).



Do their close relation with Recursion Theory, EAS's seem to provide a more natural and friendly framework than Partial Combinatory Algebras. In general, it looks interesting to reprhase open problems and results from PCA's to EAS's. For instance, we have recently proved that Klop's sufficent conditions for completability of PCA's [12] are equivalent to the existence of an *injective* S_n^m function (or to the Padding Lemma) [1].
Other promising directions for future work are that of investigating the relations between EAS's and the categorical notion of principal morphism (see for instance [2]), and more generally, to compare our structures with other systems based on partial self-application, such as the partial λ-calculus [13] or related theories [18].

References

1. A.Asperti, A.Ciabattoni. *On Completability of Partial Combinatory Algebras.* Draft.
2. A.Asperti, G.Longo. *Categories, types and structures.* MIT Press. 1991.
3. H.Barendregt. *The lambda calculus* North-Holland. 1984.
4. H.Barendregt. *Normed uniformly reflexive structures.* In *λ calculus and Computer Science Theory.* C.Bohm ed., LNCS 37, pp.272-286, Springer Verlag. 1975.
5. M.J.Beeson. *Foundation of constructive mathematics.* Springer-Verlag. 1985.
6. C.Böhm, M.Dezani-Ciancaglini. *A discrimination algorithm inside λ-β-calculus.* Theoret.Comp.Sci. vol.8, pp.271-291. 1979.
7. R.E.Byerly. *Mathematical aspects of recursive function theory.* In *Harvey Friedman's research on the foundations of mathematics*, L.A.Harrington ed., pp.339-352, Elsevier Science. 1985.
8. J.E.Fenstad. *On axiomatizing recursion theory.* In *Generalized recursion theory*, North-Holland, pp.385-404. 1974.
9. H.Friedman. *Axiomatic recursive function theory.* In *Logic Colloquium '69* , R.Gandy e M.Yates eds., pp.113-141, North-Holland. 1969.

10. J.R.Hindley, J.P.Seldin. *Introduction to combinators and λ-calculus* London Math. Soc. 1990.
11. J.R.Hindley. *The discrimination theorem holds for combinatory weak reduction.* Theor.Comput.Sci. vol.8. 1979. p.393-394
12. J.W.Klop. *Extending partial combinatory algebras.* Bulletin of the European Association for Theoret. Comp.Sci., vol.16, pp.472-482. 1982.
13. E.Moggi. *The partial λ-calculus.* Ph.D. Thesis, University of Edinburgh, 1988.
14. Y.Moschovakis. *Axioms for computation theories.* In *Logic Colloquium '69* , R.Gandy e M.Yates eds., pp.199-255, North-Holland. 1969.
15. P.Odifreddi. *Classical recursion theory.* North-Holland. 1989.
16. H.Rogers. *Theory of recursive functions and effective computability.* McGraw Hill. 1967.
17. A.Schlüter. *A Theory of rules for enumerated classes of functions.* Archive for Mathematical Logic. To appear.
18. T.Strahm. *Partial Applicative Theories and explicit substitutions.* Journal of Logic and Computation. To appear.
19. H.Strong. *Algebraically generalized recursive function theory.* IBM J. of Research and Developement, vol.12, pp.465-475. 1968.
20. E.Wagner. *Uniformly reflexive structures: an axiomatic approach to computability.* Information Sciences vol.1, pp.343-362. 1969.
21. E.Wagner. *Uniformly reflexive structures: on the nature of gödelizations and relative computability.* Trans. Amer. Math. Soc., vol.144, pp.1-41. 1969.