Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science
Electronic Notes in Theoretical Computer Science 265 (2010) 437–451
www.elsevier.com/locate/entcs

Cartesian Monoids

Rick Statman[1,2]

Department of Mathematical Sciences
Carnegie Mellon University
Pittsburgh, PA 15213, USA

Abstract

The Cartesian monoid of partial piecewise shift operators is developed as a model for programming systems such as Backus' FP. Special attention is paid to those elements which are equationally implicitly definable.

Keywords: Cartesian monoids, arithmetical hierarchy.

1 Introduction

By the time he wrote his famous letter to Dedekind in 1877, Cantor probably knew that a set with more than one element is infinite if and only if it supports a surjective pairing function.

Here are six simple steps any mathematician following his nose might take:

- Begin with a *Cantor algebra*; a set together a with surjective pairing function.
- Add a monoid of functions to the Cantor algebra.
- Write down the axioms of a Cartesian monoid and construct the free model.
- Contemplate the "op" of the free model; the piecewise shift operators.
- Allow an undefined element; partial piecewise shift operators.
- Consider the algebraic elements.

Where are we? The hyperarithmetic hierarchy. We will elaborate below.

We are principally interested in the fundamental relationship between infinity and computability. In particular, we want to suggest that if we accept the notion of a completed infinity then certain computational structures are forced on us. The

[1] The author would like to thank referee 1 for several useful remarks and suggestions
[2] Email: statman@cs.cmu.edu

doi:10.1016/j.entcs.2010.08.026

inevitability of these structures is described by the 6 steps above. We pay attention to these structures because they follow the functional programming trend.

Functional programming and equational specification have their roots in the ideas of Herbrand, Gödel, Church, Curry and Kleene carried up to the present. The standard was articulated by Backus in his 1978 Turing award lecture:

> "... a small framework which accommodates a great variety of powerful features entirely as changeable parts."

Here is a very simple framework. We will see that it accommodates a great variety of powerful features as changeable parts.

2 The notion of Cartesian monoid

The notion of Cartesian monoid axiomatizes the idea of a monoid of functions on a set S supporting a surjective pairing function $S \times S \to S$ lifted pointwise to $(S \to S) \times (S \to S) \to (S \to S)$. So a *Cartesian monoid*

$$C = (M, *, I, L, R, \langle\ \rangle)$$

is a monoid $(M, *, I)$ together with elements $L, R \in M$ and a map $\langle\ \rangle : M \times M \to M$ satisfying

$$\begin{aligned} L * \langle F, G\rangle &= F && \text{(left projection)} \\ R * \langle F, G\rangle &= G && \text{(right projection)} \\ \langle L, R\rangle &= I && \text{(surjectivity)} \\ \langle F, G\rangle * H &= \langle F * H, G * H\rangle && \text{(pointwise lifting)} \end{aligned}$$

These axioms can be found among the first few of Backus' axioms for FP, but they have a much longer history. The author is not a historian and the reader would be well served to consider [8] page 118. Generally speaking the axioms of Cartesian monoids have been regarded as too weak to be of use in logical type theory. However, they have an interesting connection to word problems.

3 Background

In 1937 Church used the lambda calculus formulated as a monoid to prove the undecidability of the word problem for semigroups. This was pursued by Curry and Feys in their 1952 book on combinatory logic with the study of the B, I monoid. This monoid turns out to be the "positive part" of Thompson's group F, and to generate this group. The group F turns out to be a subgroup of another of Thompson's groups V "the group" of the free Cartesian monoid. The free Cartesian monoid is particularly important here because it is simple and embeds in every non-trivial Cartesian monoid. Indeed, in 1965 Richard Thompson discovered a family of peculiar groups in connection to his theorem characterizing the groups with solvable word problems. The group F was rediscovered by Peter Freyd and Alex Heller in 1993 in connection with *homotopy* retracts $x * x \sim x$ which do not split $y * z \sim I$

and $x \sim z * y$. It is clear that both collections of authors knew the monoid of piecewise shift operators. This monoid is isomorphic to the free Cartesian monoid. Lambek and Scott first defined the notion of a Cartesian monoid in 1980. Actually, they defined the notion of a Cartesian category leaving the Cartesian monoids as the degenerate ones with a single object. Clearly, both were aware of the existence of the free Cartesian monoid (on 0 generators). Since the axioms of a Cartesian monoid are given by identities we say it is an algebraic structure, but it is not of the familiar kind such as groups, rings and fields. The reader should consult [8] pp. 93-97 for some elementary properties. The homomorphism/quotient theory of monoids is very complicated and not like the case of groups. Nevertheless, Cartesian monoids are ubiquitous and can be amalgamated. $N \to N$ is a universal monoid into which every countable Cartesian monoid can be embedded by Cayley's argument. Cayley's argument can be summarized as follows. The Cartesian monoid as an algebra can be realized on a set S. The operation $\langle\ ,\ \rangle$ yields a surjective pairing function on that set. The monoid lifts to the set of functions $S \to S$ under composition by Cayley's trick of $x \mapsto$ left multiplication by x, and $\langle\ ,\ \rangle$ lifts pointwise.

Each element of the free Cartesian monoid on 0 generators is denoted by an expression in $I, L, R, \langle\ \rangle, *$ and each expression can be uniquely re-written in a normal form consisting of a binary tree whose nodes correspond to applications of $\langle\ \rangle$ with strings of L's and R's at its leaves (here I counts as the empty string) and no subexpressions of the form $\langle L * x, R * x\rangle$ (this is all modulo associativity; see [12]).

4 TOPS and POPS

The "op" class of Cartesian monoids comes from taking sums instead of (Cartesian) products. These are the piecewise shift operators defined on (open subsets of) Cantor Space. Here we construe Cantor space as the product of $\{0, 1\}$, endowed with the discrete topology, along the natural numbers. The properties of Cantor space are very well known; in particular, it is a totally disconnected compact Hausdorff space. Basic open neighborhoods are represented by binary sequences u so that $f \in u \iff$ there exists g s.t. $f = u\hat{}g$. Here we add a point at infinity, denoted @, to Cantor space to get projective Cantor space. A map F : projective Cantor space $\to$ projective Cantor space is said to be a *piecewise shift operator* if whenever $f \in$ Cantor space and $F(f) \neq @$ there exist u, v finite binary sequences and a $g \in$ Cantor space such that for all $h \in$ Cantor space

(i) $f = u\hat{}g$

(ii) $F(f) = v\hat{}g$

(iii) $F(u\hat{}h) = v\hat{}h$

and of course $F(@) = @$. The set of piecewise shift operators form a Cartesian monoid by

$$\begin{aligned} I(f) &= f \\ L(f) &= 0\hat{}f \qquad (0 \text{ shift}) \\ R(f) &= 1\hat{}f \qquad (1 \text{ shift}) \\ \langle F, G\rangle(0\hat{}f) &= F(f) \qquad (\text{definition by cases}) \\ \langle F, G\rangle(1\hat{}f) &= G(f) \qquad (\text{definition by cases}) \\ F * G(f) &= G(F(f)). \end{aligned}$$

If F is a piecewise shift operator let

$$\text{dom}(F) = \{f \in \text{ Cantor space} \mid F(f) \neq @\}.$$

This makes $\text{dom}(F)$ an open subset of Cantor space and $F|_{\text{dom}(F)}$ an open continuous mapping of $\text{dom}(F)$ into Cantor space. When $\text{dom}(F) =$ Cantor space we say that F is *total*. So we have two Cartesian monoids.

- Total Operators of Piecewise Shift (TOPS)
- Partial Operators of Piecewise Shift (POPS)

Since Cantor Space is compact (König's lemma) each member of TOPS is determined by a finite binary tree with binary sequences at its leaves; it follows that

Theorem 4.1 *TOPS is isomorphic to the free Cartesian monoid on* 0 *generators.*

5 POPS

Members of the POPS monoid can be thought of as infinite binary trees with finite strings of L's and R's at their leaves (here the empty string is I). For example, the operator constantly @, also denoted @, is the complete binary tree with no leaves ($\text{dom}(@) = \emptyset$; @ corresponds to the totally undefined partial map Cantor space $\to$ Cantor space). Among the piecewise shift operators are the members of the Thompson/Freyd-Heller group F generated by the operators A_n defined by

$$\begin{aligned} n\hat{}0\hat{}0\hat{}a &\mapsto n\hat{}0\hat{}0\hat{}0\hat{}a \\ n\hat{}0\hat{}1\hat{}a &\mapsto n\hat{}0\hat{}0\hat{}1\hat{}a \\ n\hat{}1\hat{}0\hat{}a &\mapsto n\hat{}0\hat{}1\hat{}a \\ n\hat{}1\hat{}1\hat{}a &\mapsto n\hat{}1\hat{}a, \end{aligned}$$

where we have identified the number n with the sequence of n 1's. These operators satisfy the interesting identity

$$A_n * A_m = A_{m+1} * A_n$$

when $m > n$. The members U of POPS for which there is a solution to $U * x = I$ are the (left) *units* of POPS. They are all members of TOPS and form a monoid

(not Cartesian, since $\langle I, I\rangle$ is not a unit).

If C is a Cartesian monoid then the polynomial Cartesian monoid

$$C[x_1, \ldots, x_n]$$

is defined in the obvious way. Similarly for products, sums and all the normal universal algebraic constructions. Among the univariate polynomials are the linear ones (without nested indeterminates).

(i) If $F \in C$ then F is linear (constant).

(ii) If $F, G \in C$ then $F * x * G$ is linear.

(iii) If $p(x)$ and $q(x)$ are linear then so is $\langle p(x), q(x)\rangle$.

If D is an extension of C then the element F of D is said to be *algebraic* over C if it is the unique solution to a Cartesian monoid polynomial equation

$$p(x) = q(x)$$

with parameters from C. If an element F_j of D is one coordinate of the unique solution $(F_1, \ldots, F_n)$ to a Cartesian monoid polynomial equation $p(x_1, \ldots, x_n) = q(x_1, \ldots, x_n)$ with parameters from C then, by pairing, F_j lies in the sub-Cartesian monoid generated by an algebraic element. We loosely call F_j algebraic as well. More generally, the definition of algebraic is equivalent to saying that there is a triple of univariate polynomials $p(y), q(y), r(y)$ such that $p(y) = q(y)$ has a unique solution G and $F = r(G)$. Clearly multiple equations can be combined into one by pairing.

For an example consider the system E:

$$x = y * z$$

$$y = \langle R, y * L\rangle$$

$$z = \langle z * R, L\rangle.$$

This has a unique solution in POPS

$$y = \langle R, \langle R * L, \langle \ldots \langle R * L * \_ * L * L, \langle \ldots\rangle\rangle \ldots\rangle\rangle\rangle$$

where y is undefined on the sequence constantly 1,

$$z = \langle\langle\langle \ldots \langle\langle \ldots\rangle, L * R * \_ * R * R\rangle \ldots\rangle, L * R\rangle, L\rangle$$

where z is undefined on the sequence constantly 0, and

$$x = \langle L, \langle L * R, \langle \ldots \langle L * R * \_ * R * R, \langle \ldots\rangle\rangle \ldots\rangle\rangle\rangle.$$

where x is like y (hint to the reader; to read these complex expressions first pair the explicit parentheses $\langle\ ,\ \rangle$). This x is also a solution to

$$x = \langle L, x * R\rangle$$

but not the unique such solution since I is also a solution. All three can be obtained as projections of the unique solution to the fixed point equation

$$u = \langle L * R * u * R * R * u, \langle\langle R, L * R * u * L\rangle,\ \langle R * R * u * R, L\rangle\rangle\rangle.$$

Below in Section 6 we will present a representation of the set of natural numbers in TOPS. Members of POPS induce partial functions from numbers to numbers

on this representation, and a member of POPS which is a projection of the unique solution to a fixed point equation induces a partial recursive function. It will follow from the construction in Section 6 that every partial recursive function can be obtained in this way.

Now any polynomial equation $p(x) = q(x)$ which does not imply $L = R$ has a model (Cartesian monoid) with a recursively enumerable inequality relation by adding the constant F and defining $r(F) \# s(F) \iff$ there exists $t(x)$ such that $t(r(F)) = L$ and $t(s(F)) = R$. By Cayley's argument, as in Section 3, this model can be embedded into the universal monoid and the members of the model are recursive in the halting problem. So we can compute with equations, with no particular semantics, with the expected degree of complexity.

Let C be a Cartesian monoid, P a set of univariate polynomials (possibly with parameters from C) and D a subset of C. We define a relation $\sqsubset$ depending on C, D and P by $F \sqsubset G \iff$ whenever $p(x) \in P$ and $p(F) \in D$ then $p(G) = p(F)$. $\sqsubset$ is always a quasi order. Define $F \sim G \iff F \sqsubset G$ and $G \sqsubset F$. If P is all univariate polynomials with parameters from C, and D is a Cartesian sub-monoid of C, then $\sim$ is a congruence relation and $\{F/\sim \mid F \in C\}$ is a Cartesian monoid, with the attending homomorphism $F \mapsto F/\sim$.

For some examples:

(i) $C =$ TOPS, $P =$ all univariate polynomials, and $D = \{L, R\}$. Then by [12] $F \sim G \iff F = G$.

(ii) $C =$ POPS, $P =$ all linear polynomials, and $D =$ TOPS. Then $F \sim G \iff F = G$.

We call this latter condition *shiftiness*; namely, for $P =$ all linear polynomials, and $D =$ TOPS, if for all F, G

$$F \sim G \iff F = G$$

C is said to be *shifty*.

The significance of POPS in this respect is just

Theorem 5.1 *If the Cartesian monoid C has the same monoid of left units as TOPS then there is a homomorphism from C into POPS. Indeed any polynomial equation solvable in C is solvable in POPS. If C is shifty then the homomorphism is an embedding.*

Proof. To each member of C we assign an infinite binary tree with finite strings of L's and R's at its leaves (here the empty string is I). This determines a map into POPS. This map can only fail to be a Cartesian monoid homomorphism if there exist $F, G \in C$ and a string s of L's and R's such that $s * F$ is not a member of TOPS but $s * F * G$ is a string of L's and R's, say t. Now t is right invertible, by say r, but then

$$(s * F) * (G * r) = I$$

so $s * F$ is a left unit in $C -$ TOPS.

Homomorphisms preserve equations but as alluded to above the homomorphism theory of Cartesian monoids is complicated. The homomorphisms of POPS above depend on sub-Cartesian monoids D of POPS of which there are many. A subset S of Cantor space is said to be *closed under shift* if whenever $u\hat{}f \in S$ we have also $v\hat{}f \in S$. Examples are

(a) All f ultimately constant.

(b) All f ultimately constantly 0.

(c) All f of Turing degree $>$ halting problem.

If S is closed under shift then the set of all piecewise shift operators F such that

$$S \cup \mathrm{dom}(f) = \text{Cantor Space}$$

forms a Cartesian monoid. (b) is essentially the total piecewise shift operators on Baire Space. In the case of (c), the recursive piecewise shift operators are all total by König's lemma. Thus they are essentially all finite automata. Looking at such sub-Cartesian monoids led us to find a homomorphic image of POPS with a solution to our example E above with $x = I$. Of course, such a Cartesian monoid has a larger monoid of units than TOPS. □

6 HYPE

Each member of POPS is an infinite automaton whose computing power is the computational complexity of its tree. The algebraic ones are the ones uniquely specifiable by an equation. The hyperarithmetical functions were first introduced by Kleene in the 1960's (although the terminology is apparently due to Hartley Rogers) as a generalization of the notion of recursive function which allows infinite terminating searches.

Theorem 6.1 *The algebraic elements of POPS are precisely the ones with hyperarithmetic trees.*

To prove that the algebraic elements of POPS have hyperarithmetic trees is a routine coding argument with a theorem of Spector ([16] corollary XLIV(d) page 424).

We now know two quite distinct proofs of $\Rightarrow$. The first uses a representation of recursive functions and the above result of Spector, and the second requires doing bar recursion inside POPS. Bar recursion is recursion over well founded trees, and is a natural extension of general recursion. Since both require representing general recursion we have chosen to outline the second proof. Each proof provides an equational specification for each hyperarithmetic function and a demonstration that the function meets the specification.

We begin with some notation:

- $X := Y$ means X is by definition equal to Y.

Here are some definitions of data structures. Along with these definitions come

some easily verified properties.

(1) **Booleans.**

$$T := L$$
$$F := R$$
$$\text{Boole} := \{L, R\}.$$

Then

$$\text{Not}(x) = x * \langle R, L\rangle$$
$$\text{And}(x, y) = x * y * \langle I, \langle R, R\rangle\rangle.$$

(2) **Strings.**

I is the empty string
L, R are atomic strings
if s is a string then so are $s * L$ and $s * R$.

(3) **Simplices** (complete binary trees with all leaves labeled with I).

$$\text{Sim}(0) := I$$
$$\text{Sim}(n+1) := \langle \text{Sim}(n), \text{Sim}(n)\rangle$$
$$\text{Simplex} := \{\text{Sim}(n) \mid n = 0, \ldots\}$$

Then

$$\text{Next}(x) := x * \langle I, I\rangle$$
$$\text{Prev}(x) := R * x.$$

(4) **Natural Numbers.**

$$0 := L$$
$$n := L * R^n$$
$$\text{Nat} := \{L, L * R, L * R * R, \ldots\}$$

Then

$$\text{Succ}(x) = x * R$$
$$\text{Pred}(x) = x * \langle L, I\rangle.$$

(5) **Wreath Product with Nat.**

An element $f \in$ POPS defines a function (with the same name) with domain = the natural numbers, in other words a sequence, by

$$\text{Apply}(f, n) = L * R^n * f.$$

We shall write $f(n)$ for $\text{Apply}(f, n)$.

$$\text{Seq}(f) := \langle L * f, \langle L * R * f, \langle \ldots \langle L * R * f, \langle \ldots\rangle\rangle \ldots\rangle\rangle\rangle.$$

We write $[f(n) \mid n = 0, 1, \ldots]$ for $\text{Seq}(f)$.

$$\text{Im}(f) := \{L * R^n * f \mid n = 0, 1, \ldots\}$$

$\text{Seq}(f)$ may or may not be equal to f. For example, $\text{Seq}(I) \in \text{POPS} - \text{TOPS}$. If $b : N^k \to N$ then f *represents* b if $n_1 * \ldots * n_k * f = b(n_1, \ldots, n_k)$ for all $n_1, \ldots, n_k \in \text{Nat}$.

$$\text{Prefix}(f,g) := [f(n) * g(n) \mid n = 0, 1, \ldots].$$

(6) A functional F defined on a subset of POPS is said to be *algebraic* if there exist multivariate polynomials $p(s,y,z)$, $q(x,y,z)$ such that for each x in the domain of F there exists unique y, z such that

$$p(x,y,z) = q(x,y,z)$$

and $z = F(x)$.

We now need to show that some of the above defined concepts have algebraic definitions. We have 10 small propositions (i)–(x).

(i) $x \in \text{Boole} \iff x * \langle I, I\rangle = I$ and $\text{Prev}(\text{Next}(x * \langle I, \langle I, I\rangle\rangle)) = x * \langle I, \langle I, I\rangle\rangle$.

Proof. $\Leftarrow$. If $x * \langle I, I\rangle = I$ then x is right invertible thus belongs to TOPS. Indeed x can be written as a complete binary tree of depth n with each of the 2^n strings of L's and R's (in lexicographic order) followed by a random L or R at its leaves. We need to see that these are all L's or all R's. It suffices to show that $x * \langle I, \langle I, I\rangle\rangle$ is a simplex, but this follows from the second equation. □

(ii) $x = [C \mid n = 0, 1, \ldots] \iff x = \langle C, x\rangle$.

Proof. $\Leftarrow$. $x = \langle C, x\rangle$ has no solution in TOPS. The only possibility is $C = L$ and $x = R$ but this yields $I = R$. □

(iii) $x = [\text{Sim}(n) \mid n = 0, 1, \ldots] \iff x = \langle I, \text{Next}(x)\rangle$.

Here and below we write additional variables unquantified on the r.h.s. if when such objects exist they must be unique.

(iv) $x \in \text{Nat} \iff$
$x * [I \mid n = 0, 1, \ldots] = I$ and $x = L * y$ and $R * y = y * R$ and
$y * x * [\text{Sim}(n) \mid n = 0, 1, \ldots] = I$.

Proof. $\Leftarrow$. Again x is right invertible so x belongs to TOPS. Similarly for y. Thus, by [8], $y = R^n$ for some n and thus x is a natural number. □

(v) $x \in \text{Simplex} \iff y \in \text{Nat}$ and $x = y * [\text{Sim}(n) \mid n = 0, 1, \ldots]$.

(vi) $x = [n \mid n = 0, 1, \ldots] \iff$
$y = \langle R, y * L\rangle$ and $z = \langle z * R, L\rangle$ and $x = y * z$.

(vii) $\text{Seq}(y)$ is linear in y.
$x = \text{Seq}(y) \iff x = [n \mid n = 0, 1, \ldots] * y$

The next 3 constructions are essential for what follows.

(viii) $x = [[n * m \mid n = 0, 1, \ldots] \mid m = 0, 1, \ldots] \iff$
$x = \langle [n \mid n = 0, 1, \ldots] * L,\ x * R\rangle$.

(ix) $x = [[n * m \mid m = 0, 1, \ldots] \mid n - 0, 1, \ldots] \iff$
$y = \langle [n \mid n = 0, 1, \ldots], [R * m \mid m = 0, 1, \ldots] * y\rangle$ and
$x = [[L * m * n \mid n = 0, 1, \ldots] \mid n = 0, 1, \ldots] * y$.

(x) $x = [n * n \mid n = 0, 1, \ldots] \iff$
$y = \langle L, [R * n \mid n = 0, 1, \ldots] * y * R\rangle$ and
$x = [L * n \mid n = 0, 1, \ldots] * y.$

Now, the next two are definitions.

(xi) $x = \mathrm{Pre}(y) \iff x = [n * n \mid n = 0, 1, \ldots] * [y * n \mid n = 0, 1, \ldots].$

(xii) $x = \mathrm{Pre}(y, z) \iff x = \mathrm{Pre}(\mathrm{Pre}(y) * z)$,
where we may want to substitute $\mathrm{Seq}(x)$ for x on r.h.s. below, to allow members of TOPS.

Next are two small propositions.

(xiii) $\mathrm{Im}(x) \subseteq \mathrm{Nat} \iff$
$\mathrm{Pre}(y) * x * [\mathrm{Sim}(n) \mid n = 0, 1, \ldots] = [I \mid n = 0, 1, \ldots]$ and
$y * R = [R * n \mid n = 0, 1, \ldots] * y$ and
$x = [L * n \mid n = 0, 1, \ldots] * y.$

Similarly for $\mathrm{Im}(\mathrm{Im}(x)) \subseteq \mathrm{Nat}$

(xiv) $\mathrm{Im}(x) \subseteq \mathrm{Boole} \iff$
$x * [\langle I, I\rangle \mid n = 0, 1, \ldots] = [I \mid n = 0, 1, \ldots]$ and
$[R * n \mid n = 0, 1, \ldots] * \mathrm{Next}(x * \langle I, \langle I, I\rangle\rangle) = x * \langle I, \langle I, I\rangle\rangle.$

Similarly for $\mathrm{Im}(\mathrm{Im}(x)) \subseteq \mathrm{Boole}$.

Next we turn our attention to representing recursive functions.

(7) **Arithmetical Closure Properties.**
For $n_1, \ldots, n_k \in \mathrm{Nat}$

(a) Transportation

$$n_1 * \ldots * n_k * \mathrm{Pre}^k(x) = x * n_1 * \ldots * n_k.$$

(b) Thinning

$$n_1 * \ldots * n_k * \mathrm{Pre}^{k-1}([I \mid n = 0, 1, \ldots]) * x = n_2 * \ldots * n_k * x.$$

(c) Permutation

$$\begin{aligned} & n_1 * \ldots * n_i * n_{i+1} * \ldots * n_k * \\ & \qquad \mathrm{Pre}^{n-i-2}([[n * m \mid m = 0, 1, \ldots] \mid n = 0, 1, \ldots]) * x \\ = {} & n_1 * \ldots * n_{i+1} * n_i * \ldots * n_k * x. \end{aligned}$$

(d) Contraction

$$\begin{aligned} & n_1 * \ldots * n_i * \ldots * n_k * \mathrm{Pre}^{k-i-1}([n * n \mid n = 0, 1, \ldots]) * x \\ = {} & n_1 * \ldots * n_i * n_i * \ldots * n_k * x. \end{aligned}$$

(8) **Composition.**

There exists $X_0, \ldots, X_k$ such that for all $n_1, \ldots, n_k \in \mathrm{Nat}$

$$n_1 * \ldots * n_k * X_0 * x_1 * X_1 * \ldots * x_k * X_k * y$$
$$= n_1 * \ldots * n_k * (n_1 * \ldots * n_k * x_1) * \ldots * (n_1 * \ldots * n_k * x_k) * y,$$

provided that $n_1 * \ldots * n_k * x_i \in \mathrm{Nat}$, $i = 1 \ldots k-1$. Set

$$\mathrm{Comp}(x_1, \ldots, x_k, y) := X_0 * x_1 * X_1 * \ldots * x_k * X_k * y.$$

(9) **Primitive Recursion.** Let

$$x = \langle L, [n+1 \mid n = 0, 1, \ldots] \rangle * \langle y, \langle L, I \rangle * \mathrm{Comp}(x, z) \rangle$$

provided for all $n_1, \ldots, n_k, n, m \in \mathrm{Nat}$ we have $n_1 * \ldots * n_k * y \in \mathrm{Nat}$ and $n_1 * \ldots * n_k * n * m * z \in \mathrm{Nat}$, then we have

$$n_1 * \ldots * n_k * 0 * x = n_1 * \ldots * n_k * y,$$
$$n_1 * \ldots * n_k * (n+1) * x = n_1 * \ldots * n_k * n * (n_1 * \ldots * n_k * n * x) * z.$$

We assume that finite sequences $n_1, \ldots, n_k$ of natural numbers have been coded as natural numbers

$$\sharp n_1, \ldots, n_k \sharp$$

with the coding satisfying the following:

(a) The empty sequence $\sharp\ \sharp$ is coded by 0.

(b) The functions

$$\mathrm{suffix}(\sharp n_1, \ldots, n_k \sharp,\ n_{k+1}) = \sharp n_1, \ldots, n_k, n_{k+1} \sharp$$
$$\mathrm{prefix}(\sharp n_1, \ldots, n_k \sharp, n_0) = \sharp n_0, n_1, \ldots, n_k \sharp$$

are both primitive recursive.

(c) Moreover, there is a primitive recursive function enum such that

$$\mathrm{enum}(e, \sharp b_1, \ldots, b_k \sharp) = 1$$

if the eth Turing machine halts in at most k steps when applied to e using the Oracle with initial segment encoded by the bit string $b_1 \ldots b_k$ and never querying any integer $> k$, and

$$\mathrm{enum}(e, \sharp b_1, \ldots, b_k \sharp) = 0$$

otherwise.

(10) **Course of Values Recursion.**

A definition:

$$y = \mathrm{Course}(x) \Longleftrightarrow y = \langle 0 * x, R * [n * n * x \mid n = 0, 1] * \mathrm{Suffix} * y \rangle$$

(11) **General Recursion (minimization).**

Suppose that for all $n_1, \ldots, n_k, n, m \in \mathrm{Nat}$,

$$n_1 * \ldots * n_k * m * n * u \in \mathrm{Nat}.$$

Then by the above there is an algebraic definition $D[u, v, y]$ such that $D[u, v, y]$ has a unique solution and for all $n_1, \ldots, n_k, n, m$,

$$n_1 * \ldots * n_k * m * n * y = \begin{cases} 0 & \text{if there exists } l < m+1 \text{ such that} \\ & n_1 * \ldots * n_k * l * n * u = 0 \\ m+2 & \text{else.} \end{cases}$$

Now assume that for all n there exists m such that $n_1 * \ldots * n_k * m * n * u = 0$ then

$$\begin{aligned} & x = \min\{m \mid n_1 * \ldots * n_k * m * n * u = 0\} \\ \iff & D[u, v, y] \text{ and} \\ & x = 0 * [[n * m \mid n = 0, 1, \ldots] \mid m = 0, 1, \ldots] * z \text{ and} \\ & z = \mathrm{Pre}(y, [[n * m \mid n = 0, 1, \ldots] \mid m = 0, 1, \ldots]) * \\ & \qquad \langle [n * L \mid n = 0, 1, \ldots], [[n * m \mid n = 0, 1, \ldots] \mid m = 0, 1, \ldots] * z * R \rangle \end{aligned}$$

Proposition 6.2 *Suppose that $a, b : N^k \to N$ and f, g represent resp. a, b. If a is general recursive in b then there exists an algebraic definition $D[u, v, w]$ with a unique solution when $u = g$ and in this solution $w = f$.*

Remark: It is clear that $D[u, v, w]$ in the proposition does not depend on b but only on the general recursive functional F such that $F(b) = a$.

To obtain all hyperarithmetic functions we need to be able to iterate the jump operation along arbitrary recursive well orderings. The meaning of "iterate" will be explained in more detail in (13). First, the arithmetical quantifier.

(12) **Universal Quantifier.**
Another definition

$$\begin{aligned} & x = \mathrm{Uni}(y) \\ \iff & \mathrm{Im}(x) \subseteq \mathrm{Boole} \text{ and} \\ & \mathrm{Im}(\mathrm{Im}(y)) \in \mathrm{Boole} \text{ and} \\ & x * \langle y, [[L \mid n = 0, 1, \ldots] \mid n = 0, 1, \ldots] \rangle \\ & \qquad = [[L \mid n = 0, 1, \ldots] \mid n = 0, 1 \ldots] \text{ and} \\ & z = [[m * y * \langle (n + 1) * m * z * R, L \rangle \mid n = 0, 1, \ldots] \mid m = 0, 1, \ldots] \text{ and} \\ & \mathrm{Im}([m * x * \langle 0, 0 * m * z \rangle \mid m = 0, 1, \ldots]) \subseteq \mathrm{Nat}, \end{aligned}$$

so $n * x = L \iff$ for all m, $m * n * y = L$.

(13) **Jump Operation.**
Another definition

$$\begin{aligned} & x = \mathrm{Jump}(y) \\ \iff & z = y * \langle \sharp 0 \sharp, \sharp 1 \sharp \rangle \text{ and} \\ & x = \mathrm{Uni}([[n * m \mid n = 0, 1, \ldots] \mid m = 0, 1 \ldots] * \ \mathrm{Course}(z) * \mathrm{enum}) \\ & \qquad * \langle R, L \rangle \end{aligned}$$

.

Proposition 6.3 *Suppose that $a, b : N^k \to N$ and f, g represent resp. a, b. If a is arithmetical in b then there exists an algebraic definition $D[u, v, w]$ with a unique solution when $u = g$ and in this solution $w = f$.*

Remark: It is clear that $D[u, v, w]$ in the proposition does not depend on b but only on the arithmetical functional F such that $F(b) = a$.

(14) **Bar Recursion.**

A mapping T from finite sequences of natural numbers to Boolean values is a *well founded tree* if

(i) $T(n_1, \ldots, n_k) = \text{true} \Rightarrow T(n_2, \ldots, n_k) = \text{true}$,

(ii) for all functions $b : N \to N$ there exists k such that $T(b_1, \ldots, b_k) = \text{false}$.

If T is a well founded tree then the well founded tree $T \ @ \ n_1, \ldots, n_k$ is defined by $T \ @ \ n_1, \ldots, n_k \ (s) = T(n_1, \ldots, n_k, s)$ for any sequence s of natural numbers.

The iteration of the jump operation along the well founded tree T, here denoted $J(T)$, is defined recursively by

- $T(n_1, \ldots, n_k) = \text{false} \Rightarrow J(T \ @ \ n_1, \ldots, n_k)) = \{\ \}$.
- $T(n_1, \ldots, n_k) = \text{true} \Rightarrow J(T \ @ \ n_1, \ldots, n_k) =$ the halting problem relative to the set of all pairs (m, n) s.t. $m \in T \ @ \ n_1, \ldots, n_k, n$.

Suppose that $f \in \text{POPS}$ represents a well founded tree; that is

(a) f is Boolean valued;

(b) $\sharp n_1, \ldots, n_k \sharp * f = L \Rightarrow \sharp n_2, \ldots, n_k \sharp * f = L$;

(c) $\sharp n_1, \ldots, n_k \sharp * f = R \Rightarrow$ for all n_0,

$$\sharp n_0, n_1, \ldots, n_k \sharp * f = R;$$

(d) for all functions $b : N \to N$ there exists k such that

$$\sharp b_1, \ldots, b_k \sharp * f = R.$$

With $f = y$ then a solution x to

$$x = [n * n \mid n = 0, 1, \ldots] * y * \langle [[L \mid n = 0, 1, \ldots] \mid n = 0, 1, \ldots], \text{suffix} * x \rangle$$

and

$$\text{suffix} * x = \text{Jump}(\text{suffix} * \text{suffix} * x)$$

represents the iteration of the jump operation over the well founded tree represented by f.

Proposition 6.4 *Suppose that $a, b : N^k \to N$ and f, g represent resp. a, b. If a is hyperarithmetical in b then there exists an algebraic definition $D[u, v, w]$ with a unique solution when $u = g$ and in this solution $w = f$.*

It remains to show that we can pass from the coding of a tree as a number theoretic function to the tree itself, but this is straightforward. This completes the proof.

7 Future Developments

In the future we would like to develop new methods to prove that certain systems of polynomial equations have unique solutions. The Banach fixed point theorem can be useful if one can formulate the problem as a fixed point equation for a mapping

contractive in a convenient complete metric space. For example, for $F, G \in$ POPS define $d(F,G) =$ the smallest natural number n such that there is a string s of L's and R's, strung by $*$, of length n such that $s * F$ or $s * G$ is such a string of L's and R's and $s * F$ and $s * G$ are distinct. Under the usual tree topology POPS is a complete metric space with distance measure

$$m(F,G) = 2^{-d(F,G)}.$$

Here the mapping $t(y) = \langle R, y * L\rangle$ is contractive with Lipschitz constant $1/2$. On the other hand it is not obvious how to do this with u below E above.

By the main theorem above and another theorem of Spector the problem of determining whether a given Cartesian monoid polynomial equation has a unique solution in POPS is equivalent to the problem of deciding whether a given Π^1_1 sentence is true. So in the general case we cannot do better than recursive in Kleene's set of ordinal notations O.

In contrast to Klop's famous counter-example to the Church-Rosser theorem for lambda calculus with the vanilla reduction rules for surjective pairing, there are recent positive Church-Rosser results for combinatory logic with several strong forms of SP ([13,14]). These should carry over to a re-write version of the theory of Cartesian monoids supplemented by a finite number of given (but not necessarily unique) solutions to fixed point equations. It remains to see how much of the reduction theory of combinatory logic carries over to this context.

8 Conclusion

After taking our 6 steps we conclude that the axioms of Cartesian monoids provide us with a concise mathematical description of a model of computation; a description which is useful for proving facts about the model. This model is oriented toward a functional style of programming (variable free) and provides a useful algebra of programs. It is of great interest to learn how to reason productively about these structures. We remark here that, in addition, state transitions and storage can be included through Backus' AST approach. The objects in the free model and its generalizations behave like finite automata which can write (we call these "literate finite state machines") adding further state transitions. These will be discussed at a later date.

References

[1] Backus, *Can programming be liberated from the von Neumann style?*, Communications of A.C.M., **21(8)** (1978), 613–641.

[2] Belk, "Thompson's Group F", Cornell University, Department of Mathematics, Ph.D. thesis, 2004.

[3] Cannon, Floyd, and Parry, *Introductory notes on Richard Thompson's groups*, L'Enseignement Mathematique, **42** (1996), 215–256.

[4] Curien, "Categorical Combinators, Sequential Algorithms, and Functional Programming", Pittman, 1986.

[5] Freyd, Heller, *Splitting homotopy idempotents II*, Journal of Pure and Applied Algebra, **89** (1993), 93–195.

[6] Kleene, "Turing machine computable functionals of finite types I in Logic, Methodology, and the Philosophy of Science", Proceedings of the 1960 International Congress, Stanford University Press, (1962), 38–45.

[7] Lambek, *From lambda calculus to Cartesian closed categories*, in Seldin and Hindley eds., To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus, and Formalism, Academic Press, 1980.

[8] Lambek and Scott, "Introduction to Higher Order Categorical Logic", Cambridge University Press, 1986.

[9] Scott, *Relating theories of the lambda calculus*, in Seldin and Hindley eds., To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus, and Formalism, Academic Press, 1980.

[10] Statman, *Combinators and the theory of partitions*, CMU Department of Mathematics Research Report 88-31, 1988.

[11] ——— *Simply typed lambda calculus with surjective pairing*, CMU Department of Mathematics Research Report 92-164, 1992.

[12] ——— *On Cartesian monoids, CMU Department of Mathematics Research Report* 96-188, 1996. Appeared in Proceedings of CSL 1996. Springer Lecture Notes in Computer Sciences, 1258. Springer-Verlag, 1996, pp 446–459.

[13] ——— *Surjective pairing revisited* in Klop, van Oostrom, and van Raamsdonk eds., Liber Amicorum for Roel deVrijer, University of Amsterdam, 2009.

[14] Stovring, *Extending the extensional lambda calculus with surjective pairing is conservative*, LMCS, **2** (2006), 1–14.

[15] Thompson, *Embeddings into finitely generated simple groups which preserve the word problem*, in Adian, Boone, and Higman eds., The Word Problem II, Studies in Logic and the Foundation of Mathematics, North Holland, **95** (1980), 401–441.

[16] ——— *Theory of Recursive Functions and Effective Computability*, MIT Press, 1987.