Free-Algebra Models for the π-Calculus

Ian Stark

Laboratory for Foundations of Computer Science
School of Informatics
University of Edinburgh

Monday 4 April 2005

http://www.ed.ac.uk/~stark/freamp.html

Summary

The finite π-calculus:

$$P ::= \bar{x}y.P \mid x(y).P \mid \nu x.P \mid P + Q \mid P|Q \mid 0$$

has an explicit set-theoretic model, fully-abstract for strong late bisimulation congruence. [Fiore, Moggi, Sangiorgi; Stark]

We characterise this as the minimal free algebra for certain operations and equations, in the setting of Power and Plotkin's enriched Lawvere theories.

Overview

- Equational theories for different features of computation
- Using the functor category $Set^{\mathcal{I}}$
- A theory of π
- Free-algebra models and full abstraction

Notions of computation

Moggi: Computational monads for programming language features

- Nondeterminism $TX = \mathcal{P}_{\mathtt{fin}}X$
- Mutable state $TX = (S \times X)^S$
- Interactive I/O $TX = \mu Y.(X + V \times Y + Y^V)$
- Exceptions $TX = X + E$

Power and Plotkin:

Monad $\longleftrightarrow$ Algebraic theory

Use correspondence to characterize each T as free model for appropriate "notion of computation"

Algebras for nondeterministic computation

An object of nondeterministic computation A in Cartesian $\mathcal{C}$ needs ...

Operations

$$\mathrm{choice} : A^2 \longrightarrow A$$
$$\mathrm{nil} : 1 \longrightarrow A$$

Equations

$$\mathrm{choice}(p, q) = \mathrm{choice}(q, p)$$
$$\mathrm{choice}(\mathrm{nil}, p) = \mathrm{choice}(p, p) = p$$
$$\mathrm{choice}(p, (\mathrm{choice}(q, r)) = \mathrm{choice}(\mathrm{choice}(p, q), r)$$

... giving a category $\mathcal{ND}(\mathcal{C})$ of algebras $(A, \mathrm{choice}, \mathrm{nil})$

Free algebras

Free $\mathcal{ND}$-algebras over sets give a computational monad:

$$\text{free} \quad F \; \left(\begin{array}{c} \mathcal{ND}(\mathrm{Set}) \\ \uparrow \;\dashv\; \downarrow \\ \mathrm{Set} \end{array} \right) \; U \quad \text{forgetful}$$

$$T = (U \circ F) = \mathcal{P}_{\mathrm{fin}}$$

Operations induce generic effects in the Kleisli category:

$$\left.\begin{array}{r} \mathrm{choice} : A^2 \longrightarrow A^1 \\ \mathrm{nil} : A^0 \longrightarrow A^1 \end{array}\right\} \implies \left\{\begin{array}{r} \mathrm{arb} : 1 \longrightarrow T\,2 \\ \mathrm{deadlock} : 1 \longrightarrow T\,0 \end{array}\right.$$

Notions of computation determine monads

Power/Plotkin

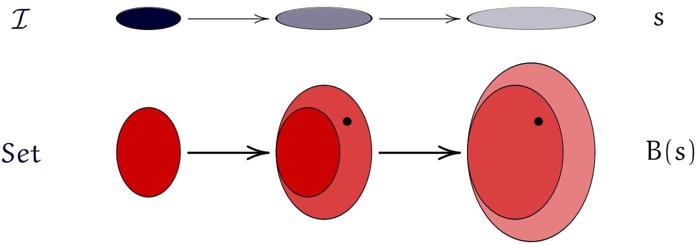
Operations + Equations $\longrightarrow$ Free-algebra models of computational features

$\longrightarrow$ Monads + generic effects

- Characterisation of known computational monads *and* effects
- Simple and flexible combination of theories
- Enriched models and arities: countably infinite, posets, $\omega\mathrm{Cpo}$

Varying sets

Functor category $Set^{\mathcal{I}}$: structures that vary with the available names where $\mathcal{I}$ = finite name sets and injections



Object $B \in Set^{\mathcal{I}}$ is a varying set: for finite name set s it gives a set $B(s)$ of values using names from s, and says how they change with renaming.

Structure in $Set^{\mathcal{I}}$

$Set^{\mathcal{I}}$ has two jobs:

- Arena for building name-aware algebras and monads
- Source of arities for operations

Relevant structure:

- Pairs $A \times B$ and function space $A \to B$
- Separated pairs $A \otimes B$ and fresh function space $A \multimap B$
- Object of names N
- Shift endofunctor $\delta A = A(\_ + 1)$, with $\delta A \cong N \multimap A$

In particular, object N serves as a varying arity.

Theory of π: operations

Nondeterministic computation

$nil : 1 \longrightarrow A$	inactive process	0
$choice : A^2 \longrightarrow A$	process sum	$P + Q$

Input/Output

$out : A \longrightarrow A^{N \times N}$	output prefix	$\bar{x}y.P$
$in : A^N \longrightarrow A^N$	input prefix	$x(y).P$
$tau : A \longrightarrow A$	silent prefix	$\tau.P$

Dynamic name creation

$new : \delta A \longrightarrow A$	restriction	$\nu x.P$

Theory of π: component equations

Nondeterministic computation

$choice$: commutative, associative and idempotent with unit nil

Input/Output

None

Dynamic name creation

$$new(x.p) = p$$

$$new(x.new(y.p)) = new(y.new(x.p))$$

Theory of π: combining equations

Commuting component theories

$$new(x.choice(p, q)) = choice(new(x.p), new(x.q))$$

$$new(z.out_{x,y}(p)) = out_{x,y}(new(z.p)) \qquad z \notin \{x, y\}$$

$$new(z.in_x(p_y)) = in_x(new(z.p_y)) \qquad z \notin \{x, y\}$$

$$new(z.tau(p)) = tau(new(z.p))$$

Interaction between component theories

$$new(x.out_{x,y}(p)) = nil$$

$$new(x.in_x(p_y)) = nil$$

Models for the theory of π

- Category $\mathcal{PI}(Set^{\mathcal{I}})$ of π-algebras $(A \in Set^{\mathcal{I}}; \mathrm{in}, \mathrm{out}, \ldots, new)$
- Process P with free names in s interpreted by $[\![P]\!]_A : N^s \longrightarrow A$
- Definition by induction over the structure of P, using operations of the theory (and the expansion law for parallel composition)

Theorem

Every such π-algebra interpretation respects strong late bisimulation congruence:

$$P \approx Q \quad \Longrightarrow \quad [\![P]\!]_A = [\![Q]\!]_A$$

Of course, this doesn't yet give us any actual π-algebras to work with

Free models for π

Each component theory has a standard monad:

Nondeterminism	$\mathcal{P}_{\mathtt{fin}}(X)$
Input/Output	$\mu Y.(X + N \times N \times Y + N \times Y^N + Y)$
Name creation	$\mathrm{Dyn}(X) = \int^k X(\_ + k)$

For the full theory of π:

$$\mathrm{Pi}(X) = \mu Y.\mathcal{P}_{\mathtt{fin}}(\mathrm{Dyn}(X) + N \times N \times Y + N \times \delta Y + N \times Y^N + Y)$$

... which is *not quite* an interleaving of the component monads

Results

Theorem

The category of π-algebras is monadic over $\mathrm{Set}^{\mathcal{I}}$:

$$\begin{array}{ccccc} & & \mathcal{PI}(\mathrm{Set}^{\mathcal{I}}) & & \\ \textit{free} & \mathrm{Pi} \uparrow & \dashv & \downarrow \mathrm{U} & \textit{forgetful} \\ & & \mathrm{Set}^{\mathcal{I}} & & \end{array}$$

Monad $T_\pi = (\mathrm{U} \circ \mathrm{Pi})$ for concurrent name-passing programs:

$$\begin{aligned} \mathrm{arb} : 1 &\longrightarrow T2 & \mathrm{send} : N \times N &\longrightarrow T1 \\ \mathrm{deadlock} : 1 &\longrightarrow T0 & \mathrm{receive} : N &\longrightarrow TN \\ \mathrm{skip} : 1 &\longrightarrow T1 & \mathrm{fresh} : 1 &\longrightarrow TN \end{aligned}$$

Results

We have the following:

- A category $\mathcal{PI}(\mathcal{S}et^{\mathcal{I}})$ of π-algebras, all sound models of π-calculus bisimulation:

$$P \approx Q \quad \Longrightarrow \quad [\![P]\!]_A = [\![Q]\!]_A$$

- An explicit free-algebra construction $\mathtt{Pi} : \mathcal{S}et^{\mathcal{I}} \to \mathcal{PI}(\mathcal{S}et^{\mathcal{I}})$ such that all $\mathtt{Pi}(X)$ are fully-abstract models of π:

$$P \approx Q \quad \Longleftrightarrow \quad [\![P]\!]_{\mathtt{Pi}(X)} = [\![Q]\!]_{\mathtt{Pi}(X)}$$

- The initial free algebra $\mathtt{Pi}(0)$ is in fact the previously known fully-abstract model.

What next?

- Use $\mathrm{FM\text{-}Cpo}$ for the full π-calculus
- Partial order arities for testing equivalences [Hennessy]
- Modify equations for early/open/weak bisimulation
- Try $\mathrm{Pi}(X)$ for applied π
- Investigate algebraic par (with effect $\mathrm{fork} : 1 \to T2$)
- Expose $\mathcal{S}et^{\mathcal{I}}$ as the category of algebras for a theory of equality testing in $\mathcal{S}et^{\mathcal{F}}$; and redo everything in the single Cartesian closed structure of $\mathcal{S}et^{\mathcal{F}}$. ($\mathcal{F}$ finite sets and all maps)