Stochastic λ-Calculi

Dana S. Scott

University Professor, Emeritus
Carnegie Mellon University

Visiting Scholar in Mathematics
University of California, Berkeley

September 2013

(A report on work in progress.)

Church's λ-Calculus

Definition. ***λ***-calculus — as a formal theory — has rules for the ***explicit definition*** of functions *via* equational axioms:

α-*conversion*

λX.[...X...] = λY.[...Y...]

β-*conversion*

(λX.[...X...])(T) = [...T...]

η-*conversion*

λX.F(X) = F

The basic syntax has one binary operation of ***application*** and one variable-binding operator of ***abstraction***. These are the "logical" notions of the theory, but we can add ***other constants*** for special operators.

Note that third axiom will be dropped in favor of a theory employing properties of a partial ordering.

The Graph Model

Definitions. (1). ***Pairing:*** $(n,m) = 2^n(2m+1)$.

(2). ***Sequence numbers:*** $\langle\rangle = 0$ and

$$\langle n_0, n_1, \ldots, n_{k-1}, n_k\rangle = (\langle n_0, n_1, \ldots, n_{k-1}\rangle, n_k).$$

(3). ***Sets:*** $\mathbf{set}(0) = \emptyset$ and $\mathbf{set}((n,m)) = \mathbf{set}(n) \cup \{m\}$.

(4). ***Kleene star:*** $X^* = \{n \mid \mathbf{set}(n) \subseteq X\}$, for sets $X \subseteq \mathbb{N}$.

Definition. The ***enumeration operator model*** is given by these definitions on ***sets*** of integers:

Application

$$F(X) = \{m \mid \exists n \in X^*. (n,m) \in F\}$$

Abstraction

$$\lambda X.[\ldots X\ldots] =$$

$$\{0\} \cup \{(n,m) \mid m \in [\ldots \mathbf{set}(n) \ldots]\}$$

NOTE: This model could easily have been defined in 1957, and it satisfies the rules of α, β-conversion (but not η).

(Some historical comments can be found at the end of these notes.)

What is the Secret?

(1) The powerset $\mathcal{P}(\mathbb{N}) = \{\mathtt{X} \mid \mathtt{X} \subseteq \mathbb{N}\}$ is a ***topological space*** with the sets $\mathcal{U}_{\mathtt{n}} = \{\mathtt{X} \mid \mathtt{n} \in \mathtt{X}^*\}$ as a ***basis*** for the topology—the ***positive*** topology.

(2) Functions $\Phi : \mathcal{P}(\mathbb{N})^{\mathtt{n}} \longrightarrow \mathcal{P}(\mathbb{N})$ are ***continuous*** iff, for all integers, $\mathtt{m} \in \Phi(\mathtt{X}_0, \mathtt{X}_1, \ldots, \mathtt{X}_{\mathtt{n-1}})$ iff there are $\mathtt{k}_{\mathtt{i}} \in \mathtt{X}_{\mathtt{i}}{}^*$ for all $\mathtt{i} < \mathtt{n}$, such that $\mathtt{m} \in \Phi(\mathbf{set}(\mathtt{k}_0), \ldots, \mathbf{set}(\mathtt{k}_{\mathtt{n-1}}))$.

(3) The application operation $\mathtt{F(X)}$ is continuous as a function of ***two*** variables.

(4) If $\Phi(\mathtt{X}_0, \mathtt{X}_1, \ldots, \mathtt{X}_{\mathtt{n-1}})$ is continuous, then the abstraction $\lambda \mathtt{X}_0 . \Phi(\mathtt{X}_0, \mathtt{X}_1, \ldots, \mathtt{X}_{\mathtt{n-1}})$ is continuous in all of the ***remaining variables***.

(5) If $\Phi(\mathtt{X})$ is continuous, then $\lambda \mathtt{X} . \Phi(\mathtt{X})$ is the ***largest set*** $\mathtt{F}$ such that for all sets $\mathtt{T}$, we have $\mathtt{F(T)} = \Phi(\mathtt{T})$.

(6) And, note, therefore, that generally $\mathtt{F} \subseteq \lambda \mathtt{X}.\mathtt{F(X)}$.

Some Lambda Properties

For all sets of integers F and G we have:

$$\lambda X.F(X) \subseteq \lambda X.G(X) \iff \forall X.F(X) \subseteq G(X),$$

$$\lambda X.(F(X) \cap G(X)) = \lambda X.F(X) \cap \lambda X.G(X),$$

and

$$\lambda X.(F(X) \cup G(X)) = \lambda X.F(X) \cup \lambda X.G(X).$$

Definition. A continuous operator $\Phi(X_0, X_1, \ldots, X_{n-1})$ is ***computable*** iff in the model this set is RE:

$$F = \lambda X_0 \lambda X_1 \ldots \lambda X_{n-1}.\Phi(X_0, X_1, \ldots, X_{n-1}).$$

Theorems.

- All pure λ-terms define ***computable*** operators.
- If $\Phi(X)$ is continuous and we let $\nabla = \lambda X.\Phi(X(X))$, then $P = \nabla(\nabla)$ is the ***least fixed point*** of Φ.
- The least fixed point of a ***computable*** operator is always computable.

Succ$(X) = \{n+1 \mid n \in X\}$, **Pred**$(X) = \{n \mid n+1 \in X\}$, **and**

Test$(Z)(X)(Y) = \{n \in X \mid 0 \in Z\} \cup \{m \in Y \mid \exists k.k+1 \in Z\}$,

with λ-calculus, suffice for defining all RE sets.

Gödel Numbering

Lemma. There is a computable $\mathbf{V} = \lambda \mathtt{X}.\, \mathbf{V}(\mathtt{X})$ where

- **(i)** $\mathbf{V}(\{0\}) = \lambda \mathtt{Y}.\lambda \mathtt{X}.\,\mathtt{Y}$,
- **(ii)** $\mathbf{V}(\{1\}) = \lambda \mathtt{Z}.\lambda \mathtt{Y}.\lambda \mathtt{X}.\,\mathtt{Z}(\mathtt{X})(\mathtt{Y}(\mathtt{X}))$,
- **(iii)** $\mathbf{V}(\{2\}) = \mathbf{Test}$,
- **(iv)** $\mathbf{V}(\{3\}) = \mathbf{Succ}$,
- **(v)** $\mathbf{V}(\{4\}) = \mathbf{Pred}$, and
- **(vi)** $\mathbf{V}(\{4+(\mathtt{n},\mathtt{m})\}) = \mathbf{V}(\{\mathtt{n}\})(\mathbf{V}(\{\mathtt{m}\}))$.

Theorem. Every ***recursively enumerable set*** is of the form $\mathbf{V}(\{\mathtt{n}\})$.

Definition. Modify the definition of **V** *via* ***finite approximations***:

- **(i)** $\mathbf{V}_{\mathtt{k}}(\{\mathtt{n}\}) = \mathbf{V}(\{\mathtt{n}\}) \cap \{\mathtt{i} \mid \mathtt{i} < \mathtt{k}\}$ for $\mathtt{n} < 5$, and
- **(ii)** $\mathbf{V}_{\mathtt{k}}(\{4+(\mathtt{n},\mathtt{m})\}) = \mathbf{V}_{\mathtt{k}}(\{\mathtt{n}\})(\mathbf{V}_{\mathtt{k}}(\{\mathtt{m}\}))$.

Theorem. Each $\mathbf{V}_{\mathtt{k}}(\{\mathtt{n}\}) \subseteq \mathbf{V}_{\mathtt{k+1}}(\{\mathtt{n}\})$ is ***finite***, the predicate $\mathtt{j} \in \mathbf{V}_{\mathtt{k}}(\{\mathtt{n}\})$ is ***recursive***, and we have:

$$\mathbf{V}(\{\mathtt{n}\}) = \bigcup_{\mathtt{k}<\infty} \mathbf{V}_{\mathtt{k}}(\{\mathtt{n}\}).$$

Theorem. The sets $\mathcal{L}_0$ and $\mathcal{L}_1$ are ***recursively enumerable***, ***disjoint***, and ***recursively inseparable***:

$$\mathcal{L}_0 = \{\mathtt{n} \mid \exists\, \mathtt{j}\, [\, 0 \in \mathbf{V}_{\mathtt{j}}(\{\mathtt{n}\})(\{\mathtt{n}\}) \wedge 1 \notin \mathbf{V}_{\mathtt{j}}(\{\mathtt{n}\})(\{\mathtt{n}\})\,]\}$$

$$\mathcal{L}_1 = \{\mathtt{n} \mid \exists\, \mathtt{k}\, [\, 1 \in \mathbf{V}_{\mathtt{k}}(\{\mathtt{n}\})(\{\mathtt{n}\}) \wedge 0 \notin \mathbf{V}_{\mathtt{k}}(\{\mathtt{n}\})(\{\mathtt{n}\})\,]\}$$

How to Randomize?

Definition. By a ***random variable*** we mean a function

$$\mathbf{X}\colon [0,1] \longrightarrow \mathcal{P}(\mathbb{N}),$$

where, for $n \in \mathbb{N}$, the set $\{\, t \in [0,1] \mid n \in \mathbf{X}(t) \,\}$

is always ***Lebesgue measurable***.

Definition. For ***random variables*** $\mathbf{X},\mathbf{Y}\colon [0,1] \longrightarrow \mathcal{P}(\mathbb{N})$,

$$[\![\, \mathbf{X} \subseteq \mathbf{Y} \,]\!] = \{\, t \in [0,1] \mid \forall n \in \mathbf{X}(t) .\, n \in \mathbf{Y}(t) \,\} / \mathtt{Null}.$$

Theorem. The random variables over $\mathcal{P}(\mathbb{N})$ form a ***Boolean-valued model*** for the $\boldsymbol{\lambda}$-calculus — expanding the two-valued model $\mathcal{P}(\mathbb{N})$.

- **This last definition is the beginning of putting a Boolean-valued Logic on random variables using the complete Boolean algebra of measurable sets modulo sets of measure zero.**

NOTE: This new model gives us a programming language with randomized parameters.

Randomized Coin Tossing

Definition. A ***coin flip*** is a random variable

$$\mathbf{F}\colon [0,1] \to \{\{0\},\{1\}\},$$

It is ***fair*** iff $\mu[\![\, \mathbf{F} = \{0\} \,]\!] = 1/2$.

Definition. ***Pairing functions*** for sets in $\mathcal{P}(\mathbb{N})$ can be defined by these enumeration operators:

$$\mathbf{Pair}(X)(Y) = \{2n \mid n \in X\} \cup \{2m+1 \mid m \in Y\}$$

$$\mathbf{Fst}(Z) = \{n \mid 2n \in Z\} \text{ and } \mathbf{Snd}(Z) = \{m \mid 2m+1 \in Z\}.$$

Definition. A ***tossing process*** is a random variable $\mathbf{T}$ where $\mathbf{Fst}(\mathbf{T})$ is a fair coin flip and where $\mathbf{Snd}(\mathbf{T})$ is ***another*** tossing — with the successive flippings all being ***mutually independent***.

The problem with using a coin-tossing process $\mathbf{T}$ is that once $\mathbf{Fst}(\mathbf{T})$ has been looked at, then that toss should be discarded, and only the coins from $\mathbf{Snd}(\mathbf{T})$ should be used in the future.

A Prototype Algorithm Language

Perhaps a solution is always to evaluate programs in the order in which expressions are written. Let's try a very sparse language.

$\mathtt{V_i}$ — a ***variable***

$\mathtt{M(N)}$ — an ***application***

$\boldsymbol{\lambda}\,\mathtt{V_i.M}$ — an ***abstraction***

$\mathtt{M}\oplus\mathtt{N}$ — a ***stochastic choice***

Let $\mathtt{V_i}=\mathtt{M}$ **in** $\mathtt{N}$ — a ***direct valuation***

The idea here is that the text $\mathtt{M}$ is evaluated in an environment giving the values of free variables. Then the result is passed on to a continuation. In case a random choice is needed, the tossing process is called.

We will try to employ a continuation semantics where the denotation of a program uses the λ-calculus formulation:

$$\langle\!\langle \mathbf{M} \rangle\!\rangle\mathtt{(env)(cont)(toss)}$$

The Semantical Equations

- $\llparenthesis V_i \rrparenthesis(E)(C)(T) =$

  $C(E(\{i\}))(T)$

- $\llparenthesis M(N) \rrparenthesis(E)(C)(T) =$

  $\llparenthesis M \rrparenthesis(E)(\lambda X.\llparenthesis N \rrparenthesis(E)(\lambda Y.C(X(Y))))(T)$

- $\llparenthesis \lambda V_i.M \rrparenthesis(E)(C)(T) =$

  $C(\lambda X.\llparenthesis M \rrparenthesis(E[X/\{i\}]))(T)$

- $\llparenthesis M \oplus N \rrparenthesis(E)(C)(T) =$

  $\mathbf{Test}(\mathbf{Fst}(T))(\llparenthesis M \rrparenthesis(E))(\llparenthesis N \rrparenthesis(E))(C)(\mathbf{Snd}(T))$

- $\llparenthesis \mathbf{Let}\ V_i = M\ \mathbf{in}\ N \rrparenthesis(E)(C)(T) =$

  $\llparenthesis N \rrparenthesis(E[\llparenthesis M \rrparenthesis(E)/\{i\}])(C)(T)$

Running a (closed) program means evaluating:

$$\llparenthesis M \rrparenthesis(\emptyset)(\lambda X.\lambda Y.X)(\mathbf{T})$$

The semantics and model as presented here, however, are only sketches. Examples of randomized algorithms need to be worked out, as well as good methods of proving probabilistic properties of programs.

Some Background References

There are many approaches to modeling λ-calculus, and expositions and historical references can be found in Cardone-Hindley [2009]. In 1972 Plotkin wrote an AI report at the University of Edinburgh entitled "A set-theoretical definition of application" which remained unpublished until it was incorporated into the more extensive paper Plotkin [1993], which discusses many kinds of models. Scott developed his model based on the powerset of the integers subsequently, but he only later realized it was basically the same as Plotkin's model. See Scott [1976] for further details where he called the idea The Graph Model.

• F. Cardone and J.R. Hindley. Lambda-Calculus and Combinators in the 20th Century. In: Volume 5, pp. 723-818, of Handbook of the History of Logic, Dov M. Gabbay and John Woods eds., North-Holland/Elsevier Science, 2009.

• Gordon D. Plotkin. Set-theoretical and other elementary models of the λ-calculus. Theoretical Computer Science, vol. 121 (1993), pp. 351-409.

• Dana S. Scott. Data types as lattices. SIAM Journal on Computing, vol. 5 (1976), pp. 522-587.

Much earlier, enumeration reducibility was introduced by Rogers in lecture notes and mentioned by Friedberg-Rogers [1959] as a way of defining a positive reducibility between sets. Enumeration degrees are discussed at length in Rogers [1967]. There is now a vast literature on the subject. Enumeration operators are also studied in Rogers [1967] as well. Earlier, Myhill-Shepherdson [1955] defined functionals on partial functions with similar properties. Neither team saw that their operators possessed an algebra that would model λ-calculus, however.

• John Myhill and John C. Shepherdson, Effective operations on partial recursive functions, Zeitschrift für Mathematische Logik und Grundlagen der Mathematik, vol. 1 (1955), pp. 310-317.

• Richard M. Friedberg and Hartley Rogers jr., Reducibility and Completeness for Sets of Integers. Mathematical Logic Quarterly, vol. 5 (1959), pp. 117-125. Some of the results of this paper are presented in abstract, Journal of Symbolic Logic, vol. 22 (1957), p. 107.

• Hartley Rogers, Jr., Theory of Recursive Functions and Effective Computability, McGraw-Hill, 1967, xix + 482 pp.

More Background References

Some historical remarks on the notion of partial equivalence relations (PERs) as an interpretation of types are given by Bruce et al. [1990], where we learn that they were introduced by Myhill and Shepherdson [1955] for types of first-order functions, and then extended to simple types by Kreisel [1959]. Scott took the use of partial equivalence relations from the work of Kreisel and collaborators.

• K. Bruce, A. A. Meyer, and J. C. Mitchell. The semantics of second-order lambda calculus. In G. Huet, editor. Logical Foundations of Functional Programming, pp. 273–284. Addison-Wesley, 1990.

• G. Kreisel. Interpretation of analysis by means of constructive functionals of finite type. In A. Heyting, editor, Constructivity in Mathematics, pp. 101–128. North-Holland Co., Amsterdam, 1959.

Two papers about introducing random features in λ-calculus are Deliguoro-Piperno [1995] and Dal Lago-Zorzia [2012]. Both of those articles have many historical references. Thanks to Thomas F. Icard III for pointing out these two references in connection with work on his Stanford Ph.D. thesis.

• U. Deliguoro and A. Piperno. Nondeterministic Extensions of Untyped λ-Calculus. Information and Computation, vol. 122 (1995), pp. 149–177.

• Ugo Dal Lago and Margherita Zorzia. Probabilistic operational semantics for the lambda calculus. RAIRO - Theoretical Informatics and Applications, vol. 46 (2012), pp. 413-450.

There is a very large literature on probabilistic powerdomains, and many technical details, as well as background and historical references, can be found in the recent papers of Michael Mislove (see his WWW site). Connections between random variables and domains of distributions are also explained in these papers.

• Michael Mislove. Discrete Random Variables over Domains. Theoretical Computer Science, vol. 380 (2007), pp. 181-198.

• Michael Mislove. Anatomy of a Domain of Continuous Random Variables I. Submitted to TCS (2013), 19 pp. and Anatomy of a Domain of Continuous Random Variables II. Springer LNCS, vol. 7860 (2013), pp. 225-245.