

Replace this file with `prentcsmacro.sty` for your meeting,
or with `entcsmacro.sty` for your meeting. Both can be
found at the [ENTCS Macro Home Page](#).

A Syntax for the λ -calculus^{*}

Masahiko Sato

Graduate School of Informatics, Kyoto University

Randy Pollack

LFCS, University of Edinburgh

Frege was the first to formulate the syntax of a language with binders. He used two disjoint sets of variables, *global variables* using Latin letters and *local variables* using German letters [13, page 25]. Later [4], for instance, followed this approach. Traditionally, logic and λ -calculus have been formulated using only one sort of variables, but approaches using two species of variables have reappeared in recent years [2,6,5,7,1]; this is our approach in this paper.

This abstract is organized as follows. In section 1 we introduce the set $\mathbb{S}[\mathbb{X}]$ of *symbolic expressions* freely generated from a countable set \mathbb{X} of *atoms* (*global variables*) and natural numbers (serving as a second species of names for local variables). In contrast with de Bruijn indices [2] our natural number local variables are used as names, directly referring to corresponding name-carrying binders. We also introduce a new notion of B-algebra and characterize the set of symbolic expressions as a free B-algebra. We define substitutions as B-algebra endomorphisms on \mathbb{S} . Atom permutations (i.e. bijective substitutions) are automorphisms and the group of permutations acts on \mathbb{S} , inducing a notion of equivariance. Concretely, the notion of substitution satisfies purely structural equations; there is no need for renaming variables while computing substitutions. This section closes by defining a function on \mathbb{S} for canonically choosing the names of bound local variables.

In Section 2, we introduce the *internal syntax* for λ -calculus, so called because it can be easily implemented on a computer. Rather than quotienting the raw symbolic expressions by α -equivalence, as is usual, we define the set \mathbb{L} of λ -terms as a subset of the free B-algebra \mathbb{S} . Substitution (as defined on \mathbb{S}) is closed on \mathbb{L} . In this paper, we use the untyped λ -calculus as a canonical example of linguistic structure with variable binding.

* This work is an extensively revised version of [10].

Elsewhere, we also introduce *external syntax* which is intended to be used by humans. The external syntax is explained in terms of the internal syntax. We omit the external syntax from this workshop version; it is described in [10] and in an extended paper currently being written.

Formalization

The development of \mathbb{L} , and some examples, has been formalized in nominal Isabelle [12]. We use a nominal Isabelle atom type for \mathbb{X} , and take advantage of convenient automation tools provided by nominal Isabelle. We show that \mathbb{L} and nominal lambda terms are isomorphic by a function that respects substitution, adding some confidence in the faithfulness of both representations.

1 Symbolic expressions

Define the set $\mathbb{S}[\mathbb{X}]$ of *symbolic expressions* as a free algebra (datatype) generated from a denumerably infinite set \mathbb{X} of *atoms* (also called *global variables*) and from the set \mathbb{N} of natural numbers (which includes 0, and are called *local variables*). We use ‘ X ’, ‘ Y ’, ‘ Z ’ for global variables, ‘ x ’, ‘ y ’, ‘ z ’ for local variables, and ‘ M ’, ‘ N ’, ‘ P ’, ‘ Q ’ for symbolic expressions.

$$\frac{X : \mathbb{X}}{X : \mathbb{S}} \quad \frac{x : \mathbb{N}}{x : \mathbb{S}} \quad \frac{M : \mathbb{S} \quad N : \mathbb{S}}{(M N) : \mathbb{S}} \quad \frac{M : \mathbb{S}}{[x]M : \mathbb{S}}$$

This definition reflects our idea that local variables may get bound by abstraction ‘ $[x]M$ ’ but global variables cannot be bound, even syntactically.

The domain of symbolic expressions is our universe of discourse in the rest of the paper. As usual we have structural induction, and well-founded induction on the size of symbolic expressions. Since \mathbb{S} is defined depending on \mathbb{X} , we write ‘ $\mathbb{S}[\mathbb{X}]$ ’ for \mathbb{S} when we wish to emphasize the dependency.

A *B-algebra* (‘B’ is for ‘binding’) is a triple

$$\langle A, () : A \times A \rightarrow A, [] : \mathbb{N} \times A \rightarrow A \rangle$$

where A is a set containing \mathbb{N} as a subset. A *B-algebra homomorphism* is a function h from a B-algebra A to a B-algebra B such that for all $M, N \in A$ and $x \in \mathbb{N}$:

$$h(x) = x, \quad h((M N)) = (h(M) h(N)), \quad \text{and} \quad h([x]M) = [x]h(M).$$

It is easy to see that $\langle \mathbb{S}[\mathbb{X}], () : \mathbb{S} \times \mathbb{S} \rightarrow \mathbb{S}, [] : \mathbb{N} \times \mathbb{S} \rightarrow \mathbb{S} \rangle$ is a *free* B-algebra with the free generating set \mathbb{X} . In fact, let B be a B-algebra; any $\rho : \mathbb{X} \rightarrow B$ can be uniquely extended to a B-algebra homomorphism $[\rho] : \mathbb{S}[\mathbb{X}] \rightarrow B$ as follows:

- (i) $[\rho]X \triangleq \rho(X)$.
- (ii) $[\rho]x \triangleq x$.
- (iii) $[\rho](M N) \triangleq ([\rho]M [\rho]N)$.
- (iv) $[\rho][x]M \triangleq [x][\rho]M$.

Consider the case where B is \mathbb{S} and $\rho : \mathbb{X} \rightarrow \mathbb{S}$ is a finite map. We call such a map a *substitution*. If ρ sends X_i to P_i ($1 \leq i \leq n$, X_i distinct) and fixes the rest, $[\rho] : \mathbb{S} \rightarrow \mathbb{S}$ is an endomorphism and we write ‘ $[P_i/X_i]M$ ’ for $[\rho]M$. The substitution operation commutes with the B-algebra operations.

An endomorphism $[\rho]$ is an automorphism if and only if ρ is a permutation, that is $\rho : \mathbb{X} \rightarrow \mathbb{X}$ is a bijection. The group $G_{\mathbb{X}}$ of finite permutations on \mathbb{X} acts on the B-algebra $\mathbb{S}[\mathbb{X}]$ by the group action $[\pi]M$. For each $\pi \in G_{\mathbb{X}}$ the group action $[\pi](-)$ determines a B-algebra automorphism on $\mathbb{S}[\mathbb{X}]$. From this group action we have the general notion of $G_{\mathbb{X}}$ -equivariant maps. That permutations of global names preserve interesting judgements about languages with binding was observed and used in formal reasoning by [7,8]. That this is an instance of the general notion of equivariance was first emphasized by [3,9].

We need to define another operation for “going under binders”, which substitutes a symbolic expression P for *free* occurrences of a local variable y in M :

- (i) $[P/y]X \triangleq X$.
- (ii) $[P/y]x \triangleq \begin{cases} P & \text{if } x = y, \\ x & \text{if } x \neq y. \end{cases}$
- (iii) $[P/y](M N) \triangleq ([P/y]M [P/y]N)$.
- (iv) $[P/y][x]M \triangleq \begin{cases} [x]M & \text{if } x = y, \\ [x][P/y]M & \text{if } x \neq y. \end{cases}$

This operation is purely technical; it doesn’t correspond to a natural operation on the informal notion of terms. In particular $[P/y]$ is a function from \mathbb{S} to \mathbb{S} but, unless P is y , it is not a B-algebra homomorphism since it neither preserves y nor commutes with the abstraction operation $[y](-)$.

Now we define a function $H : \mathbb{X} \times \mathbb{S} \rightarrow \mathbb{N}$ which will play a crucial role in our development of the internal syntax (section 2).

- (i) $H_X(Y) \triangleq \begin{cases} 1 & \text{if } X = Y, \\ 0 & \text{if } X \neq Y. \end{cases}$
- (ii) $H_X(x) \triangleq 0$.
- (iii) $H_X((M N)) \triangleq \max(H_X(M), H_X(N))$.
- (iv) $H_X([x]M) \triangleq \begin{cases} H_X(M) & \text{if } H_X(M) = 0 \text{ or } x = 0 \text{ or } H_X(M) > x, \\ x + 1 & \text{otherwise.} \end{cases}$

We call $H_X(M)$ *the height of X in M*. Note that H is equivariant

$$[\pi]H_X(M) = H_{[\pi]X}([\pi]M), \quad (1)$$

and $H_X(M) = 0$ if and only if X does not occur in M . If $H_X(M) = n + 1$, then (writing M in tree form) either $n = 0$ and X occurs as a leaf at least once in the tree and all the paths from the root to X do not go through a non-zero binder, or

n is the largest binder among all the binders we encounter if we go down the tree from the root to all the occurrences of X .

2 The internal syntax

We inductively define ‘ \mathbb{L} ’, the set of λ -terms, as a subset of the free B-algebra $\mathbb{S}[\mathbb{X} \cup \{\mathbf{app}, \mathbf{lam}\}]$, where \mathbb{X} contains neither \mathbf{app} nor \mathbf{lam} .

$$\frac{}{X : \mathbb{L}} \quad \frac{M : \mathbb{L} \quad N : \mathbb{L}}{(\mathbf{app} \ M \ N) : \mathbb{L}} \quad \frac{M : \mathbb{L} \quad x = H_X(M)}{(\mathbf{lam} \ [x] [x/X]M) : \mathbb{L}} \quad (*)$$

\mathbb{L} is an equivariant predicate on $\mathbb{S}[\mathbb{X} \cup \{\mathbf{app}, \mathbf{lam}\}]$. Although \mathbb{L} is not a subalgebra of \mathbb{S} , it enjoys the property of being closed under the substitution operation (Theorem 2.1).

Motivation:

Symbolic expressions do not faithfully represent lambda terms for two reasons:

- (i) Some symbolic expressions are ill formed by having local variables (natural numbers) that are not bound, e.g. the symbolic expression x .
- (ii) The set of well formed symbolic expressions in the above sense does not canonically represent the set of informal lambda terms; i.e. there are “alpha-convertible” symbolic expressions, such as $[x]x$ and $[y]y$.

The first of these problems is handled by inductively defining a predicate (i.e. a subset) on \mathbb{S} picking out the well formed symbolic expressions. In [7,8] this predicate is called *variable closed* ($\mathbf{vclosed}$), defined by:

$$\frac{}{\mathbf{vclosed} \ X} \quad \frac{\mathbf{vclosed} \ M \quad \mathbf{vclosed} \ N}{\mathbf{vclosed} \ (\mathbf{app} \ M \ N)} \quad \frac{\mathbf{vclosed} \ M}{\mathbf{vclosed} \ (\mathbf{lam} \ [x] [x/X]M)} \quad (+)$$

This representation still has the second problem mentioned above; e.g. both $[x]x$ and $[y]y$ are $\mathbf{vclosed}$. A consequence of this weakness, for example, is that the Church–Rosser theorem does not hold concretely of $\mathbf{vclosed}$ symbolic expressions, but only up to α -equality.

A main contribution of the present work, developed by the first author, is an alternative solution to this second problem, the failure of canonical representation. Rule (+) above, viewed as a constructor of lambda terms, takes X and M and constructs $(\mathbf{lam} \ [x] [x/X]M)$ for *any* x . To make the representation canonical it suffices to choose x canonically in this construction, and that is the purpose of the height function $H_X(M)$; compare rules (*) and (+). Of course, this canonical choice must be well behaved; e.g. must respect substitution:

Theorem 2.1 (Substitution) *If $P, Q : \mathbb{L}$ then $[Q/Y]P : \mathbb{L}$.*

In the forthcoming paper we develop a theory of \mathbb{L} (formalised in Isabelle) sufficient to prove this result, and to carry out examples such as typing and reduction

properties of lambda terms. As is usual [7,8,12,1], we derive a strengthened induction principle for \mathbb{L} that encapsulates the use of sufficiently fresh variables via equivariance.

Example 2.2 We can define β -reduction on \mathbb{L} along standard lines.

$$\frac{(\mathbf{1am} [x]P) : \mathbb{L} \quad N : \mathbb{L}}{(\mathbf{app} (\mathbf{1am} [x]P) N) \rightarrow [N/x]P} (\beta).$$

$$\frac{M_1 \rightarrow M_2 \quad N : \mathbb{L}}{(\mathbf{app} M_1 N) \rightarrow (\mathbf{app} M_2 N)} \quad \frac{M : \mathbb{L} \quad N_1 \rightarrow N_2}{(\mathbf{app} M N_1) \rightarrow (\mathbf{app} M N_2)}$$

$$\frac{M \rightarrow N \quad x = H_X(M) \quad y = H_X(N)}{(\mathbf{1am} [x][x/X]M) \rightarrow (\mathbf{1am} [y][y/X]N)} (\xi)$$

The subtlety that rule (ξ) (reduction under a binder) may change the bound name is essential for this concrete representation with canonical names to work.

This relation is equivariant and has natural properties. The side conditions on the rules guarantee that $M \rightarrow N$ implies $M : \mathbb{L}$ and $N : \mathbb{L}$.

Many other interesting aspects of this work are detailed in our forthcoming paper.

References

- [1] B. Aydemir, A. Charguéraud, B.C. Pierce, R. Pollack, and S. Weirich. Engineering Formal Metatheory. In *Proceedings of the 35th Annual ACM SIGPLAN-SIGACT Symposium on Principles on Programming Languages*, pages 3–15. ACM Press, 2008.
- [2] N.G. de Bruijn. Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation, with application to the Church-Rosser theorem. *Indag. Math.*, 34(5), 1972.
- [3] Murdoch Gabbay and Andrew Pitts. A new approach to abstract syntax involving binders. In G. Longo, editor, *Proceedings of the 14th Annual Symposium on Logic in Computer Science (LICS'99)*, pages 214–224, 1999.
- [4] Gerhard Gentzen. Untersuchungen über das logische schliessen. *Math. Zeitschrift*, 39, 1934. English translation in [11].
- [5] Andrew Gordon. A mechanism of name-carrying syntax up to alpha-conversion. In *Higher Order Logic Theorem Proving and its Applications. Proceedings, 1993*, LNCS 780, pages 414–426. Springer-Verlag, 1993.
- [6] Gérard Huet. The constructive engine. In R. Narasimhan, editor, *A Perspective in Theoretical Computer Science*. World Scientific Publishing, 1989.
- [7] J. McKinna and R. Pollack. Pure Type Systems formalized. In M. Bezem and J.F. Groote, editors, *Proceedings of the International Conference on Typed Lambda Calculi and Applications, TLCA '93, Utrecht*, number 664 in LNCS, pages 289–305. Springer-Verlag, 1993.
- [8] J. McKinna and R. Pollack. Some lambda calculus and type theory formalized. *Journal of Automated Reasoning*, 23(3–4):373–409, 1999.
- [9] A.M. Pitts. Nominal logic, a first order theory of names and binding. *Information and Computation*, 186:165–193, 2003.
- [10] M. Sato. External and internal syntax of the λ -calculus. In Buchberger, Ida, and Kutsia, editors, *Proc. of the Austrian-Japanese Workshop on Symbolic Computation in Software Science, SCSS 2008*, number 08–08 in RISC-Linz Report Series, pages 176–195, 2008.
- [11] M. E. Szabo, editor. *The Collected Papers of Gerhard Gentzen*. North Holland, 1969.
- [12] C. Urban, S. Berghofer, and M. Norrish. Barendregt’s variable convention in rule inductions. In *Automated Deduction – CADE-21*, number 4603 in LNCS, pages 35–50. Springer-Verlag, 2007.
- [13] Jean van Heijenoort. *From Frege to Gödel: A source book in mathematical logic, 1879–1931*. Harvard University Press, Cambridge, Mass., 1967.