J Autom Reasoning
DOI 10.1007/s10817-011-9229-y

A Canonical Locally Named Representation of Binding

Randy Pollack · Masahiko Sato · Wilmer Ricciotti

Received: 2 May 2011 / Accepted: 4 May 2011

Abstract This paper is about completely formal representation of languages with binding. We have previously written about a representation following an approach going back to Frege, based on first-order syntax using distinct syntactic classes for locally bound variables vs. global or free variables (Sato and Pollack, J Symb Comput 45:598–616, 2010). The present paper differs from our previous work by being more abstract. Whereas we previously gave a particular concrete function for canonically choosing the names of binders, here we characterize abstractly the properties required of such a choice function to guarantee canonical representation, and focus on the metatheory of the representation, proving that it is in substitution preserving isomorphism with the nominal Isabelle representation of pure lambda terms. This metatheory is formalized in Isabelle/HOL. The final section outlines a formalization in Matita of a challenging language with multiple binding and simultaneous substitution. The Isabelle and Matita proof files are available online.

Keywords Binding · Lambda calculus · Formal proof

Pollack is partially supported by the project CerCo which acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET-Open grant number: 243881.

R. Pollack (✉)
LFCS, School of Informatics, University of Edinburgh, Edinburgh, United Kingdom
e-mail: rpollack@inf.ed.ac.uk

M. Sato
Graduate School of Informatics, Kyoto University, Kyoto, Japan
e-mail: masahiko@kuis.kyoto-u.ac.jp

W. Ricciotti
Department of Computer Science, University of Bologna, Bologna, Italy
e-mail: ricciott@cs.unibo.it

Published online: 24 May 2011

1 Introduction

This paper is about completely formal representation of languages with binding. The desiderata for such a representation include that it be semantically satisfying, convenient for use with machine proof checking programs and natural for humans to read, write and reason about. There is a large body of prior work in this area, and several widely used technical approaches, each with many variations. We do not try to cover or catalogue that work here, but point to some papers that do include some such discussion: [1, 2, 6, 11, 13, 14, 16, 26].

We have previously written [22, 23] about a representation following an approach going back to Frege [7], Gentzen [9] and Prawitz [21]. This approach is based on first-order syntax using distinct syntactic classes for *locally bound* variables (Frege used German letters [28, page 25]) vs. *global* or *free* variables, also called *parameters* (Frege used Latin letters). This approach was first formalised and used for significant machine checked examples in [15], and independently in an interesting variation, in [10]. For a modern presentation of the latter variation, see [2]. We call our representation 'locally named' because the abstractors carry local variable names. (The variation described in [2, 10] is called 'locally nameless' because its abstractors are nameless as in de Bruijn representation [6].) We call our representation 'canonical' because α-equivalence is syntactic identity, and we need never formally define or discuss α-equivalence or α-conversion.

The representation described in our previous work [22, 23] improves on that of [15] by being canonical, thus giving the good properties of locally nameless representation without the hassle of adjusting indexes.[1] The representation of the present paper differs from [22, 23] by being more abstract. In [22, 23] we use a particular set for local variables (the natural numbers), and a particular function choosing the unique names for binders to attain canonical representation. In the present paper we show the same approach can use any infinite decidable set of atoms for local variables, and characterize abstractly the properties of a choice function for binding names that guarantee canonical representation. This is explained in detail in Section 3.

Given that there are many approaches to reasoning about binding in use, we outline the advantages of our approach. It is a first-order representation in that the collection of lambda terms is an inductively defined predicate (i.e. subset) of a datatype. Thus it does not need a special logic (as, e.g. Twelf [17]), but can be expressed directly in any logic supporting inductive definition of types (or sets) and predicates; e.g. classical extensional Higher Order Logic (HOL) or constructive intensional type theory. It is lightweight enough to prototype directly in Coq or HOL without large scale special purpose tools. (Serious use of our representation would be greatly improved by development of tools, but we have not done this.) The representation is natural, with name-carrying binders that are, nevertheless, injective constructors (unlike nominal Isabelle, where binders are not injective because of

[1] The representations in [2, 10] already avoid deBruijn lifting by using only locally closed terms. However these representations still have messy arguments when reasoning under binders.

α-equivalence). Thus our representation is both canonical and enjoys a kind of direct pattern matching/inductive reasoning. Finally, unlike Higher Order Abstract Syntax approaches, the expressiveness of our representation is limited only by the proof-theoretic strength of the meta language (Coq, Isabelle, ...), not by any anomaly of the approach itself. For example we believe (without having carried out the experiment) that the calculus of constructions is easily formalized in our approach, using, say, Coq as a meta language, and that its normalization is provable in this formalization. See Section 5 for a different example, and some discussion of the limits of our approach. Summing up, our representation is natural, lightweight and expressive.

Outline of the Paper The underlying syntactic datatype of our representation (called symbolic expressions) is presented, with its properties, in Section 2. Section 3 gives an inductively defined predicate on symbolic expressions intended to pick out a subset canonically representing pure lambda terms. This predicate is parametrised by a *height* function for selecting the names of binders; the major part of this section discusses the properties such a height function must have. Section 4 shows that the properties we have outlined are exactly what is required to prove that our lambda term representation is isomorphic with the lambda term representation of nominal Isabelle [26]. This section ends with the example of β-reduction on our lambda terms. In Section 5 we outline our representation of the *multivariate lambda calculus* [20], a system with multiple binding and simultaneous substitution. Section 6 concludes.

1.1 Formalisation

Everything in Sections 2, 3 and 4 has been formalized in nominal Isabelle [26] by the first author. Our Isabelle theory files are available online.[2] We use a nominal Isabelle atom type, and take advantage of convenient automation tools provided by nominal Isabelle. We show, in Isabelle, that our lambda terms are isomorphic (respecting substitution) with lambda terms as usually represented in nominal Isabelle. Although it is an informal issue whether our formal representation adequately captures the lambda terms in your mind, the fact that two formal representations agree adds to confidence about the faithfulness of both representations.

Section 5 outlines a formalization by the third author of *multivariate lambda calculus* [20] in the Matita proof system[3] using our representation. Matita implements an intensional constructive type theory very close to the logic of Coq. The multivariate lambda calculus includes multiple binding and simultaneous substitution. The Matita proof files for this example are available online.[4]

[2] http://homepages.inf.ed.ac.uk/-rpollack/-export/-PollackSatoRicciotti_IsabelleJAR.tgz

[3] http://matita.cs.unibo.it/

[4] http://homepages.inf.ed.ac.uk/rpollack/export/PollackSatoRicciotti_MatitaJAR.tgz

2 Symbolic Expressions

We start with two distinct denumerably infinite sets of atoms: $\mathbb{X}$ for *global* (free) variables (sometimes called *parameters*), and $\mathbb{V}$ for *local* (bound) variables. We reserve 'X', 'Y', 'Z' for global variables and 'x', 'y', 'z' for local variables. The datatype of *symbolic expressions*, $\mathbb{S}$, is defined by:

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

The expression '$(M\ N)$' is said to be the *pair of* M *and* N. The expression '$[x]\, M$' is said to be the *abstraction by* x *of* M; 'x' is said to be the *binder* and 'M' to be the *body* of this expression. We have the usual induction principles on $\mathbb{S}$, namely structural induction and well-founded induction on the size of an expression.

Informally, the body of an abstraction expression is the *scope* of the binder. The body M of an expression $[x]\, M$ may bind x again, as in $[x]\ [x]\, x$. In this case we informally consider that the occurrences of 'x' in the body are bound by the inner binder. This definition of symbolic expressions reflects our idea that local variables may get bound, but global variables can never get bound. Note, however, that there is no actual binding explicit in this free construction.

In this paper we use pure lambda calculus as the running example, and we have given the definition of symbolic expressions for representing pure lambda calculus. However, in more complex languages (such as system F, having binders for term variables and for type variables) each kind of binder will have its own two species of variables.

Occurrences of Variables To each symbolic expression M we assign a set $\mathsf{LV}(M)$ called the *free local variables* of M:

$$\begin{aligned}
\mathsf{LV}(X) &\triangleq \{\} \\
\mathsf{LV}(x) &\triangleq \{x\} \\
\mathsf{LV}((M\ N)) &\triangleq \mathsf{LV}(M) \cup \mathsf{LV}(N) \\
\mathsf{LV}([x]\, M) &\triangleq \mathsf{LV}(M) - \{x\}
\end{aligned}$$

We say that x *occurs free* in M if $x \in \mathsf{LV}(M)$. Similarly, define a set $\mathsf{GV}(M)$ called the *global variables* of M:

$$\begin{aligned}
\mathsf{GV}(X) &\triangleq \{X\} \\
\mathsf{GV}(x) &\triangleq \{\} \\
\mathsf{GV}((M\ N)) &\triangleq \mathsf{GV}(M) \cup \mathsf{GV}(N) \\
\mathsf{GV}([x]\, M) &\triangleq \mathsf{GV}(M)
\end{aligned}$$

In practice we are only interested in whether X occurs in M or not, and borrow the nominal logic notation $X \sharp M$ to mean $X \notin \mathsf{GV}(M)$. Further, we extend this notation to other classes of global variables (e.g. as needed to represent System F, mentioned above) and homomorphically to composite structures that may occur in applications, such as typing contexts. Nominal Isabelle supports this extended notation with a typeclass of atom types.

Replacement of Variables We define an operation of replacement for global variables, called *substitution*, by structural recursion:

$$\begin{aligned}
[P/X]Y &\triangleq \begin{cases} P & \text{if } X = Y, \\ Y & \text{if } X \neq Y. \end{cases} \\
[P/X]x &\triangleq x \\
[P/X](M\ N) &\triangleq ([P/X]M\ [P/X]N) \\
[P/X][x]M &\triangleq [x][P/X]M
\end{aligned} \tag{1}$$

From the fourth clause notice that if P contains free occurrences of x, these occurrences will be bound after the substitution (e.g. $[x/X][x]X = [x]x$). This is not the intended behaviour of substitution; known ways to avoid this include renaming x in $[x]M$ or renaming x in P. The former is called α-renaming (e.g. as in [5, 24]); the latter is called lifting (e.g. as in [6]). In Section 3 we will use a third way in which we only consider a subset of $\mathbb{S}$ whose members contain no free occurrences of local variables: there simply are no free occurrences of local variables to get captured. This is the historical reason for using two distinct species of names for local vs. global variables. See [2, 15, 16] for previous modern formalizations using this approach. The operation $[P/X]M$ will be the correct notion of substitution for our canonical representation of lambda terms.

We also need a purely technical operation of replacement for local variables, $[P/y]M$, which is used in our development but does not correspond to a natural operation on lambda terms. It is defined by structural recursion:

$$\begin{aligned}
[P/y]X &\triangleq X \\
[P/y]x &\triangleq \begin{cases} P & \text{if } x = y, \\ x & \text{if } x \neq y. \end{cases} \\
[P/y](M\ N) &\triangleq ([P/y]M\ [P/y]N) \\
[P/y][x]M &\triangleq \begin{cases} [x]M & \text{if } x = y, \\ [x][P/y]M & \text{if } x \neq y. \end{cases}
\end{aligned} \tag{2}$$

If $x = y$ in the fourth clause, we have $[P/y][x]M = [x]M$, which is natural since $\mathsf{LV}([x]M)$ does not contain y in this case. If $x \neq y$, then substitution commutes with the abstraction operation, also natural. As above, this operation does not prevent capture (e.g. $[x/y][x]y = [x]x$), but will be correct on the subset of $\mathbb{S}$ containing no free local variables.

We can show the following useful lemmas by induction on the construction of M.

Lemma 1 (Permutation Lemma) *Both forms of substitution are equivariant: if* π *is a finite permutation on* $\mathbb{X}$*, then*

$$\pi\cdot[P/Y]M = [\pi\cdot P/\pi\cdot Y]\pi\cdot M \quad \textit{and} \quad \pi\cdot[P/y]M = [\pi\cdot P/y]\pi\cdot M.$$

Proof Induction on structure of M. □

In the following, π will range over finite permutations on $\mathbb{X}$. The importance of permuting names in reasoning about binding is discussed in [15, 16]. The connection with the general notion of equivariance of a group action is pointed out in [8, 18]. For a more detailed and abstract discussion in the present context, see [23].

Finite permutations are compositions of pairwise swapping of atoms, which we write as $(X, Y){\cdot}M$.

Lemma 2 (Substitution and Swapping) *If* $Y \sharp M$ *then* $[Y/X]M = (X, Y){\cdot}M$.

Lemma 3 (Substitution Lemma) *If* $X \neq Y$ *and* $X \sharp Q$, *then we have*

$$[Q/Y][P/X]M = [[Q/Y]P/X][Q/Y]M.$$

Lemma 4 (Substitutions Cancel) *If* $X \sharp M$ *then* $M = [x/X][X/x]M$.

Lemma 5 (Substitutions Commute) *If* $X \neq Y$ *and* $x \notin \mathsf{LV}(N)$ *then*

$$[Y/x][N/X]M = [N/X][Y/x]M.$$

Occurrences of Binders In order to achieve canonical representation of lambda terms, we define one more technical function $\mathsf{E}_X(M) : \mathbb{X} \times \mathbb{S} \to (\mathbb{V}\ \text{set})$ computing the set of local names occurring in binding position between the root of M and any occurrence of X in M. This is not needed by users of our representation, but is needed for the metatheory we are presenting here. The definition (by structural recursion) is straightforward.

$$\begin{aligned}
\mathsf{E}_X(Y) &\triangleq \{\} \\
\mathsf{E}_X(y) &\triangleq \{\} \\
\mathsf{E}_X(\,(M\ \ N)\,) &\triangleq \mathsf{E}_X(M) \cup \mathsf{E}_X(N) \\
\mathsf{E}_X(\,[x]\,M) &\triangleq \begin{cases} \{\} & \text{if } X \sharp M\text{: no paths to } X \text{ in } M \\ \{x\} \cup \mathsf{E}_X(M) & \text{otherwise: } x \text{ in every path} \end{cases}
\end{aligned}$$

E is equivariant

$$\pi{\cdot}\mathsf{E}_X(M) = \mathsf{E}_{\pi\cdot X}(\pi{\cdot}M). \tag{3}$$

The intuition behind the definition of E is suggested by the observation:

$$x \in \mathsf{E}_X(M) \implies M \neq [X/x][x/X]M \tag{4}$$

because the inner substitution captures x. More positively we have the following crucial lemma.

Lemma 6 (Decomposition of Substitution)

1. $[N/x]M = [N/X][X/x]M$ *if* $X \sharp M$.
2. $[N/X]M = [N/x][x/X]M$ *if* $x \notin \mathsf{LV}(M)$ *and* $x \notin \mathsf{E}_X(M)$.

Proof For both claims the proof is by structural induction on M. In (2.), when $M = [y]\,M'$ consider the subcases $x = y$ and $x \neq y$. □

3 Lambda Terms

As mentioned above, the first step toward a good representation of lambda terms based on symbolic expressions is to consider the subset of symbolic expressions that contain no unbound local variables, so that the two replacement operations of Section 2 are capture free. We have $\mathsf{LV}(-)$ with which to express this subset, but it is convenient to define it inductively. In [15, 16] this predicate is called *variable closed* ($\mathsf{vclosed}$)

$$\frac{}{\mathsf{vclosed\ X}} \qquad \frac{\mathsf{vclosed\ M} \quad \mathsf{vclosed\ N}}{\mathsf{vclosed\ (M\ \ N)}} \qquad \frac{\mathsf{vclosed\ M}}{\mathsf{vclosed\ [x]\,[x/X]M}}\ (+)$$

We trivially have $\mathsf{vclosed\ M}$ iff $\mathsf{LV}(M) = \{\}$, but the inductive definition of $\mathsf{vclosed}$ comes with an induction principle that is more useful than the structural induction principle on $\mathbb{S}$ in that it has no case for free local variables. We also have that $\mathsf{vclosed}$ is closed under substitution:

$$\mathsf{vclosed\ M} \wedge \mathsf{vclosed\ N} \implies \mathsf{vclosed\ [N/X]M} \tag{5}$$

as substitution cannot create unbound local variables. You might worry that '$[x/X]M$' in the conclusion of rule (+) is not capture avoiding, but this is unimportant.

The set of $\mathsf{vclosed}$ symbolic expressions can be used as a representation of lambda terms, but it is not a canonical representation; e.g. the distinct $\mathsf{vclosed}$ expressions $[x]\,x$ and $[y]\,y$ should be considered equal as lambda terms. As a consequence, for example, the Church–Rosser theorem for the usual β-reduction does not hold concretely for $\mathsf{vclosed}$ symbolic expressions, but only "up to α-equivalence" [19]. Our game here is to avoid the need to reason about, or even define, α-equivalence, so $\mathsf{vclosed}$ is unsatisfactory. However [15, 16] show that much dependent type theory can be carried out concretely over $\mathsf{vclosed}$ symbolic expressions. The idea is to use well-behaved relations. E.g. Tait–Martin-Löf parallel reduction, defined on $\mathsf{vclosed}$ symbolic expressions, does have the Church–Rosser theorem concretely, and this is the notion of reduction used in [15, 16]. This may seem ad hoc, but can also be seen as a technical justification for the definition of Tait–Martin-Löf parallel reduction.

3.1 A Subset of Symbolic Expressions

The failure of $\mathsf{vclosed}$ to give a canonical representation for lambda terms is because rule (+) above, viewed as a constructor of lambda terms, takes X and M and constructs $[x]\,[x/X]M$ for *any* x. To make the representation canonical we must choose x canonically in this construction. We use functions of type $\mathbb{X} \times \mathbb{S} \rightarrow \mathbb{V}$, called *height functions*, to make this choice canonical. For height function F we write $\mathsf{F}_X(M)$ for the F-height of X in M. Inductively define a predicate on $\mathbb{S}$, parameterised by an arbitrary height function F:

$$\frac{}{X : \mathbb{L}_{\mathsf{F}}} \qquad \frac{M : \mathbb{L}_{\mathsf{F}} \quad N : \mathbb{L}_{\mathsf{F}}}{(M\ \ N) : \mathbb{L}_{\mathsf{F}}} \qquad \frac{M : \mathbb{L}_{\mathsf{F}} \quad x = \mathsf{F}_X(M)}{[x]\,[x/X]M : \mathbb{L}_{\mathsf{F}}}\ (*)$$

Compare rule $(*)$ with rule $(+)$ above. Now the question is: what properties must F have for $\mathbb{L}_{\mathsf{F}}$ to be an adequate representation of lambda terms? It is straightforward that $M : \mathbb{L}_{\mathsf{F}} \Longrightarrow \mathsf{vclosed\,M}$, so $M : \mathbb{L}_{\mathsf{F}} \Longrightarrow (\mathsf{LV}(M) = \{\})$, and the replacement operations of Section 2 are capture free on $\mathbb{L}$. But it is not obvious that $\mathbb{L}$ is closed under substitution:

$$M : \mathbb{L} \;\wedge\; N : \mathbb{L} \;\;\Longrightarrow\;\; [N/X]M : \mathbb{L} \tag{6}$$

This is Theorem 1 in the concrete setting of [23], and follows from the properties of height functions we discuss below.

Before proceeding we improve our notation. The whole presentation that follows is implicitly parameterised by a height function F, so we drop the subscript 'F'. Noticing that the conclusion of rule $(*)$ is a common construction, we define a function $\mathsf{abs}\colon \mathbb{X} \times \mathbb{S} \to \mathbb{S}$ by

$$\mathsf{abs}_X M \;\triangleq\; \big[\mathsf{F}_X(M)\big]\big[\mathsf{F}_X(M)/X\big]\,M.$$

Rule $(*)$ can now be rewritten as

$$\frac{M : \mathbb{L}}{\mathsf{abs}_X M : \mathbb{L}}\;(**)$$

The reader should keep in mind that $\mathsf{abs}_X M$ is analogous to the informal lambda abstraction $\lambda x.m$. For example, for any F we have $X \sharp\, \mathsf{abs}_X M$, just as "$x$ doesn't occur free in $\lambda x.m$" (in nominal language, x is not in the support of $\lambda x.m$). Thus, for example, $[N/X]\mathsf{abs}_X M = \mathsf{abs}_X M$. Also, while the concrete constructors of $\mathbb{S}$, including $[-]-$, are injective ($\mathbb{S}$ is a datatype), abs is not necessarily so. For example we will see that for well behaved F (i.e. equivariant), $\mathsf{F}_X(X) = \mathsf{F}_Y(Y)$ (hence $[\mathsf{F}_X(X)/X]X = [\mathsf{F}_Y(Y)/Y]Y$) so

$$\mathsf{abs}_X X \;=\; \big[\mathsf{F}_X(X)\big]\big[\mathsf{F}_X(X)/X\big]\,X \;=\; \big[\mathsf{F}_Y(Y)\big]\big[\mathsf{F}_Y(Y)/Y\big]\,Y \;=\; \mathsf{abs}_Y Y. \tag{7}$$

even when $X \neq Y$. Similarly, although the constructors of $\mathbb{S}$ are equivariant, we do not know that abs is equivariant in general; that depends on properties of F.

Finally, we define the notion of *instantiation*, $\triangledown : \mathbb{S} \times \mathbb{S} \to \mathbb{S}$:

$$([x]\,M) \triangledown N \;\triangleq\; [N/x]M.$$

Instantation will only be applied to abstractions in this paper, so its value on other constructors is irrelevant. Informally, one might see instantiation as a way to "lift" the improper operation of (2) to proper terms in $\mathbb{L}$. Instantiation inherits equivariance from Lemma 1. For well behaved F it will be provable that

$$[x]\,M : \mathbb{L} \;\wedge\; N : \mathbb{L} \;\;\Longrightarrow\;\; ([x]\,M) \triangledown N : \mathbb{L}\,. \tag{8}$$

This is Theorem 2 in the more concrete setting of [23].

3.1.1 Good F

Here are three properties of F that together guarantee that $\mathbb{L}$ is an adequate representation in the sense that there is a substitution preserving isomorphism between $\mathbb{L}$ and the set of pure lambda terms represented in nominal Isabelle.

$$\begin{array}{llll}
\text{(XHE)} & & \mathsf{F}_X(M) \;=\; \mathsf{F}_{\pi\cdot X}(\pi\cdot M) & \mathsf{F}\text{ equivariant,}\\
\text{(XHF)} & & \mathsf{F}_X(M) \;\notin\; \mathsf{E}_X(M) & \mathsf{F}\text{ fresh,}\\
\text{(XHP)} & X \neq Y \wedge X \sharp Q \implies & \mathsf{F}_X(M) = \mathsf{F}_X([Q/Y]M) & \mathsf{F}\text{ preserved by substitution.}
\end{array}$$

We call a height function, F, *excellent* if it satisfies these properties. In fact it is sufficient for F to satify these properties relativized to $\mathbb{L}$ to give an adequate representation (Theorem 1).

$$\begin{array}{lll}
\text{(HE)} & M : \mathbb{L} \implies \mathsf{F}_X(M) = \mathsf{F}_{\pi\cdot X}(\pi\cdot M) & \mathsf{F}\text{ equivariant on }\mathbb{L},\\
\text{(HF)} & M : \mathbb{L} \implies \mathsf{F}_X(M) \notin \mathsf{E}_X(M) & \mathsf{F}\text{ fresh on }\mathbb{L},\\
\text{(HP)} & M : \mathbb{L} \wedge Q : \mathbb{L} \implies & \mathsf{F}\text{ preserved by substitution on }\mathbb{L}.\\
 & \quad X \neq Y \wedge X \sharp Q \implies \mathsf{F}_X(M) = \mathsf{F}_X([Q/Y]M) &
\end{array}$$

A height function satisfying (HE), (HP) and (HF) is called *good*. Clearly every excellent height function is good. Since we are interested in the weakest restrictions on F that guarantee an adequate representation, we focus on good F, although in applications to reasoning about languages with binding, the relativization to $\mathbb{L}$ is inessential.

In the next three paragraphs we discuss the goodness properties individually, showing as motivation that they give the informally expected equations for "α-conversion" and substitution under abs. In Section 3.1.2 we show that they are together consistent and independent. Section 3.2 digresses from the main argument to present some other aspects of height functions. Section 4 contains the main results connecting height functions with canonical representation of lambda terms.

(HE) F *is Equivariant on* $\mathbb{L}$ Thus $\mathsf{F}_X(X) = \mathsf{F}_Y(Y)$ for any X and Y. Note that $\mathsf{F}_X(M) \in \mathbb{V}$ is distinct from any global variable, so $\pi\cdot\mathsf{F}_X(M) = \mathsf{F}_X(M)$. It is trivial from the definition of abs that F is equivariant iff abs is equivariant:

$$\mathsf{F}_X(M) = \mathsf{F}_{\pi\cdot X}(\pi\cdot M) \iff \pi\cdot\mathsf{abs}_X M = \mathsf{abs}_{\pi\cdot X}(\pi\cdot M). \tag{9}$$

so in the presence of (HE), abs is equivariant on $\mathbb{L}$. Also, (HE) implies equivariance of $\mathbb{L}$ itself:

$$\text{(HE)} \implies (\forall \pi M.\; M : \mathbb{L} \iff \pi\cdot M : \mathbb{L})$$

(proof by well founded induction on the size of M).

To use our representation in practice it is essential to derive strengthened induction principles for the relations of interest, as discussed in [15, 16, 25]. (HE) is required to prove these principles, and in particular a strengthened induction principle for $\mathbb{L}$ is derivable from (HE) using well founded induction on the size of a symbolic expression, as discussed in [23].

(HF) F *is Fresh* Recall the definition of E from Section 2. (HF) says $\mathsf{F}_X(M)$ does not occur in binding position on any path from the root of M to any occurrence of X in M, so using $\mathsf{F}_X(M)$ as the local name to bind positions of X in M will not inadvertently capture any occurrences of $\mathsf{F}_X(M)$ in M.

Lemma 7 (Height Lemma) *Assume* $\mathsf{F}_X(M) \notin \mathsf{LV}(M)$. *The following are equivalent:*

$$\mathsf{F}_X(M) \notin \mathsf{E}_X(M) \tag{10}$$

$$\forall N : \mathbb{L}.\ [N/X]M = (\mathsf{abs}_X M)\triangledown N \tag{11}$$

$$\forall Z.\ [Z/X]M = (\mathsf{abs}_X M)\triangledown Z \tag{12}$$

Proof Unfolding the definitions of abs and $\triangledown$, (10) $\Longrightarrow$ (11) by Lemma 6. Equation (11) $\Longrightarrow$ (12) is trivial. Taking $Z = X$ in (12), (10) follows by (4). □

Lemma 8 *Assume* (HF), $M : \mathbb{L}$ *and let* $x = \mathsf{F}_X(M)$. *Then*

(1) $[N/X]M = [N/x][x/X]M.$

(2) *Suppose also* $N : \mathbb{L}$, $x = \mathsf{F}_Y(N)$ *and* $[x/X]M = [x/Y]N$. *Then*

$$M = [X/Y]N \qquad \text{and} \qquad X \neq Y \implies Y \sharp M.$$

Proof (1) is a special case of Lemma 6(2.), using (HF). (2) follows from (1). □

Lemma 9 (abs and "α-conversion") *Assume* $M : \mathbb{L}$ *and* $N : \mathbb{L}$. *Let* α *be the formula*

$$\alpha \ \stackrel{\triangle}{=}\ (X = Y \ \wedge\ M = N) \ \vee\ (X \neq Y \ \wedge\ M = [X/Y]N \ \wedge\ X \sharp N)$$

which informally expresses α*-equivalence of '*$\lambda X.M$*' and '*$\lambda Y.N$*'.*

1. *From* (HE) *we have:*

$$\alpha \implies \mathsf{abs}_X M = \mathsf{abs}_Y N.$$

2. *From* (HF) *we have:*

$$\mathsf{abs}_X M = \mathsf{abs}_Y N \implies \alpha.$$

Thus, from properties (HE) and (HF) together we see that abs behaves like informal λ-abstraction for α-conversion. We have informally argued above that properties (HE) and (HF) are natural, or anyway required for our representation to work adequately (the formal argument is set out in Section 4). What is interesting about Lemma 9 is how the informal notion of α-equivalence factors through these two properties.

(HP) F *is Preserved by Substitution* Using Lemma 3 it is easy to see that:

$$X \neq Y \ \wedge\ X \sharp Q \implies$$
$$(\ \mathsf{F}_X(M) = \mathsf{F}_X([Q/Y]M) \iff [Q/Y]\mathsf{abs}_X M = \mathsf{abs}_X[Q/Y]M\ ). \tag{13}$$

Thus (HP) shows that moving substitution under abs has the same equation as moving substitution under informal λ-abstraction. Equations (7), (13) and Lemma 9, exemplify how our representation constructs the intended behaviour of abstraction out of simple structural and functional operations in standard logic.

From (HE) and (HP) we can prove (6) using a strengthened induction principle (derived using (HE) as mentioned above) on premise $M : \mathbb{L}$. From (6) and (HF) (via Lemma 7) we can prove (8).

3.1.2 Consistency and Independence of the Goodness Properties

Before proceeding to show that, for good F, $\mathbb{L}$ is a faithful reresentation of lambda terms, we show that the 'good' properties make sense.

Lemma 10 (Good F Exist) *Interpret $\mathbb{V}$ by the natural numbers. The height function H defined by*

$$\begin{aligned}
\mathsf{H}_X(Y) &\triangleq \begin{cases} 1 & \textit{if } X = Y \\ 0 & \textit{if } X \neq Y \end{cases} \\
\mathsf{H}_X(x) &\triangleq 0 \\
\mathsf{H}_X((M\ \ N)) &\triangleq \max(\mathsf{H}_X(M), \mathsf{H}_X(N)) \\
\mathsf{H}_X([x]\,M) &\triangleq \begin{cases} \mathsf{H}_X(M) & \textit{if } \mathsf{H}_X(M) = 0 \textit{ or } \mathsf{H}_X(M) > x \\ x+1 & \textit{otherwise} \end{cases}
\end{aligned}$$

is excellent, hence also good.

This is the original height function defined by the second author in [22], and machine checked by the first author as reported in [23].

Proof All three properties are proved by induction on the structure of M. The proof of (XHF) uses a lemma

$$x \in \mathsf{E}_Y(P) \implies \mathsf{H}_Y(P) > x$$

also proved by structural induction on P. □

Lemma 11 ((HE), (HP) and (HF) are Independent)

1. (HE) $\wedge$ (HF) $\not\Longrightarrow$ (HP)
2. (HE) $\wedge$ (HP) $\not\Longrightarrow$ (HF)
3. (HF) $\wedge$ (HP) $\not\Longrightarrow$ (HE)

Proof Case (1.) is satisfied by the height function

$$\begin{aligned}
\mathsf{F1}_X Y &\triangleq 0 \\
\mathsf{F1}_X x &\triangleq 0 \\
\mathsf{F1}_X\,(M\ \ N) &\triangleq \mathsf{F1}_X M + \mathsf{F1}_X N + 1 \\
\mathsf{F1}_X\,[x]\,M &\triangleq x + \mathsf{F1}_X M + 1.
\end{aligned}$$

To see that $\mathsf{F1}$ doesn't satisfy (HP) take $X \neq Y$, $M = (X\ Y)$ and $Q = (Y\ Y)$. To see that it does satisfy (HF) note that if x occurs bound in M then $\mathsf{F1}_X M > x$. (HE) of $\mathsf{F1}$ is proved by well-founded induction on the size of M.

Case (2.) is satisfied by the constant height function $\mathsf{F2}_X M \triangleq 0$. To see that this function doesn't satisfy (HF) take $M = [0]\,X$.

Case (3.) is satisfied by height function

$$\mathsf{F3}_X M \triangleq \text{ if } X \neq X_0 \vee X_0 \sharp M \text{ then } \mathsf{H}_X(M) \text{ else } \mathsf{F3'}_X M.$$

where X_0 is any distinguished global variable, and

$$\begin{aligned} \mathsf{F3'}_X Y &\triangleq 0 \\ \mathsf{F3'}_X x &\triangleq 0 \\ \mathsf{F3'}_X (M\ N) &\triangleq \mathsf{F3'}_X M + \mathsf{F3'}_X N \\ \mathsf{F3'}_X [x]\, M &\triangleq \text{if } X \sharp M \text{ then } 0 \text{ else } 1 + x + \mathsf{F3'}_X M. \end{aligned}$$

To see that this function is not equivariant, take $X \neq X_0$, $M = X$, $\pi = (X_0, X)$. □

Remark 1 In light of the equivalence between (11) and (12), one might wonder if, in the presence of (HF), (HP) might be equivalently replaced by (HP0):

$$\text{(HP0)} \qquad (M : \mathbb{L} \wedge \mathrm{X} \neq \mathrm{Y} \wedge \mathrm{X} \neq \mathrm{Z}) \implies \mathsf{F}_\mathrm{X}(\mathrm{M}) = \mathsf{F}_\mathrm{X}([\mathrm{Z}/\mathrm{Y}]\mathrm{M}).$$

We can now see that this conjecture is false, because the example $\mathsf{F1}$ above satisfies (HP0) as well as (HF) and (HE), but, as shown above, fails to satisfy (HP). Further, our proof of adequacy of the representation (Section 4) really seems to need the strong property (HP).

3.2 Free Choices in Good F

This subsection is an aside from the main argument. The specific height function H from Lemma 10 has the property:

$$X \sharp M \iff \mathsf{H}_X(M) = 0$$

In general we have:

Lemma 12 *For given* F *and* $X \sharp M$, (HE) *says that* $\mathsf{F}_X(M)$ *does not depend on* X; (HP) *says that* $\mathsf{F}_X(M)$ *does not depend on* M; *and* (HE) $\wedge$ (HP) *says that* $\mathsf{F}_X(M)$ *does not depend on* X *or* M.

$$\begin{aligned} \text{(HE)} &\implies \forall (M : \mathbb{L}).\ \exists! v.\ \forall X.\ (X \sharp M) \implies \mathsf{F}_X(M) = v \\ \text{(HP)} &\implies \forall X.\ \exists! v.\ \forall (M : \mathbb{L}).\ (X \sharp M) \implies \mathsf{F}_X(M) = v \\ \text{(HE)} \wedge \text{(HP)} &\implies \exists! v.\ \forall X (M : \mathbb{L}).\ (X \sharp M) \implies \mathsf{F}_X(M) = v \end{aligned}$$

Conversely, if F is excellent and $X \neq Y$, we cannot prove that $\mathsf{F}_X(X) \neq \mathsf{F}_X(Y)$.

Lemma 13 (Free Choice in Excellent Functions) *If* F *is excellent, and* $v \in \mathbb{V}$ *then*

$$\mathsf{F}'_X(M) \triangleq \textit{if } M = X \textit{ then } v \textit{ else } \mathsf{F}_X(M)$$

is excellent.

Thus there is no definition schema for all excellent height functions (or all good height functions) parameterised by a single choice function. At least two independent choices are involved in the definition of a height function; one choice for $\mathsf{F}_X(X)$ (Lemma 13) and an independent choice for $\mathsf{F}_X(Y)$ where $X \neq Y$ (Lemma 12).

4 A Canonical Representation of Lambda Terms

As has been pointed out in [12] the claim that a formalization is an adequate representation of some informal notion is not itself formalizable. For one thing, there can be no formal connection between an informal thing and a formal thing. Also, the notion of "adequate representation", even of one formal thing by another formal thing, depends on which properties are meant to be preserved. Here we will show that if F is good then $\mathbb{L}$ is in substitution preserving isomorphism with the formal nominal Isabelle representation of pure lambda terms. With this we can show, for example, that β-reduction is preserved by this representation. In [26] there is a proof that the nominal representation of pure lambda terms is isomorphic to the quotient of raw lambda-term syntax up to α-conversion.

We use the set $\mathbb{X}$ (the set of global variables of symbolic expressions) for the variables of the nominal representation, and let A, B, C range over nominal lambda terms. To fix notation, we write the nominal datatype of lambda terms, $\mathsf{n}\mathbb{L}$, as if it is generated by

$$\frac{}{X : \mathsf{n}\mathbb{L}} \qquad \frac{A : \mathsf{n}\mathbb{L} \quad B : \mathsf{n}\mathbb{L}}{(A \;\; B) : \mathsf{n}\mathbb{L}} \qquad \frac{A : \mathsf{n}\mathbb{L}}{[X]\, A : \mathsf{n}\mathbb{L}}$$

$\mathsf{n}\mathbb{L}$ has HOL equality up to α-conversion; the reader who wants to know what this means must consult [26].

Substitution of nominal lambda terms is defined [26] by

$$\begin{aligned} X[Y{::=}C] &\triangleq \text{if } X = Y \text{ then } C \text{ else } X \\ (A \;\; B)[Y{::=}C] &\triangleq (A[Y{::=}C] \;\; B[Y{::=}C]) \\ ([X]\, A)[Y{::=}C] &\triangleq \text{if } X \sharp (Y, C) \text{ then } [X]\,(A[Y{::=}C]) \end{aligned}$$

This last conditional equation means that one must sometimes α-convert $[X]\, A$ away from bound name X to avoid capture. Compare this last equation with (13) for bringing substitution under abs.

In analogy with $\triangledown$ (Section 3.1) we also define a notion of *instantiation* for nominal terms, $\blacktriangledown : \mathsf{n}\mathbb{L} \times \mathsf{n}\mathbb{L} \to \mathsf{n}\mathbb{L}$.

$$([X]\, A) \blacktriangledown B \triangleq A[X{::=}B]$$

Instantation will only be applied to abstractions in what follows.

Now we can define a relation $\,!!\, : \mathsf{n}\mathbb{L} \times \mathbb{S} \to \mathsf{bool}$ that will become the representation function:

$$\frac{}{X \mathbin{!!} X} \qquad \frac{A \mathbin{!!} M \quad B \mathbin{!!} N}{(A\ \ B) \mathbin{!!} (M\ \ N)} \qquad \frac{A \mathbin{!!} M}{[X]\,A \mathbin{!!} \mathsf{abs}_X M} \tag{14}$$

In the last rule, $\mathsf{abs}_X M$ implicitly mentions a height function F. For any F we have the following lemma, which we use without further mention.

Lemma 14 (Straightforward Properties of the Representation $\,!!\,$)

$$\text{(RS)} \qquad \begin{array}{rl} \exists M.\ A \mathbin{!!} M & !!\ \textit{is total,} \\ A \mathbin{!!} M \implies M : \mathbb{L} & !!\ \textit{hits only well formed terms.} \\ M : \mathbb{L} \implies \exists A.\ A \mathbin{!!} M & !!\ \textit{is surjective,} \\ A \mathbin{!!} M \implies (X \sharp A \iff X \sharp M) & !!\ \textit{respects global names.} \end{array}$$

We want to think of !! as a function. Clearly !! is defined for every $A : \mathsf{n}\mathbb{L}$, but *a priori* it is not obvious that !! is single valued because, in the third rule, the abstraction constructor of nominal terms is not injective. For example, let $X_0 \neq X_1$ be distinguished global names, and consider a height function

$$\mathsf{F}_X(M) \ \stackrel{\triangle}{=}\ \text{if } X = X_0 \text{ then } 0 \text{ else } 1\,.$$

We have $[X_0]\,X_0 = [X_1]\,X_1$ (nominal terms), $[X_0]\,X_0 \mathbin{!!} [0]\,0$ and $[X_1]\,X_1 \mathbin{!!} [1]\,1$, but $[0]\,0 \neq [1]\,1$ (symbolic expressions).

To turn the relation !! into a function $! : \mathsf{n}\mathbb{L} \to \mathbb{S}$ we use the HOL definite description operator, $\mathtt{THE}$:

$$!A \ \stackrel{\triangle}{=}\ \mathtt{THE}\ M.\ A \mathbin{!!} M.$$

$!A$ is some $M : \mathbb{S}$ such that $A \mathbin{!!} M$ *if there is a unique such* M. Otherwise $!A$ is a value about which we know nothing except its type. To show that this function behaves as desired, we must know that !! is single valued:

$$\text{(RSV)} \qquad A \mathbin{!!} M_1 \wedge A \mathbin{!!} M_2 \implies M_1 = M_2.$$

This is proved from (HE) in Lemma 16 below.

Lemma 15 (Equations for Representation Function $!$) *Assume* (RSV). *Then* $(!A) : \mathbb{L}$ *and*

$$\begin{array}{rcl} A \mathbin{!!} M & \iff & M = !A \\ !X & = & X \\ !(A\ \ B) & = & (!A\ \ !B) \\ ![X]\,A & = & \mathsf{abs}_X !A \end{array}$$

Proof By the meaning of the definite description operator. □

Note that assuming (RSV), ! inherits the properties of Lemma 14 via Lemma 15.

Lemma 16 ((HE), (RSV) and Equivariance of !)

$$\text{(HE)} \iff (\text{(RSV)} \wedge \pi{\cdot}!\mathrm{A} = !\pi{\cdot}\mathrm{A})$$

Proof For direction $\Longrightarrow$, first prove !! is equivariant:

$$A \mathrel{!!} M \implies \pi{\cdot}A \mathrel{!!} \pi{\cdot}M \tag{15}$$

by induction on $A \mathrel{!!} M$, using (HE) in the abstraction case. Then (RSV) is proved by induction on its first premise followed, in each resulting case, by inversion of its second premise; the only non-trivial case is for abstraction, which uses (15) and Lemma 9(1). Finally equivariance of ! follows easily from these two facts.

For direction $\Longleftarrow$, assume $M : \mathbb{L}$ and let $M = !A$. We have:

$$\begin{aligned}
\pi{\cdot}\mathsf{abs}_X M &= \pi{\cdot}\mathsf{abs}_X !A \\
&= \pi{\cdot}!([X]\,A) && \text{by Lemma 15} \\
&= !(\pi{\cdot}[X]\,A) && \text{by equivariance of !} \\
&= ![\pi{\cdot}X]\,\pi{\cdot}A && \text{by equivariance of nominal abstraction} \\
&= \mathsf{abs}_{\pi\cdot X}(!\pi{\cdot}A) && \text{by Lemma 15} \\
&= \mathsf{abs}_{\pi\cdot X}(\pi{\cdot}!A) && \text{by equivariance of !} \\
&= \mathsf{abs}_{\pi\cdot X}(\pi{\cdot}M).
\end{aligned}$$

giving (HE) by (9). □

It is interesting to note that (HP) independently implies that !! is equivariant, but we still seem to need (HE) (via Lemma 9(1)) to show (RSV) from that fact.

Let us name some more properties of ! that play a part in what follows:

(RI)	$!A = !B \implies A = B$	! is injective,
(RRS)	$!(A[X::=B]) = [!B/X]!A$	! respects substitution,
(RRI0)	$!(([X]\,A)\blacktriangledown Y) = (\mathsf{abs}_X !A)\triangledown Y$	! respects instantiation by parameters.

These properties are well-formed, if meaningless, on their own, but in the presence of (RSV) take on their intended meaning via Lemma 15.

4.1 Good F Give an Adequate Representation

Theorem 1 (Adequacy of representation) *If* F *is a good height function then* ! *is an adequate representation. In particular,* ! *is a bijection* ((RS)[5] *and* (RI)) *that satisfies* (RRS) *and* (RRI0)*:*

$$\text{(HE)} \wedge \text{(HP)} \wedge \text{(HF)} \implies \text{(RSV)} \wedge \text{(RS)} \wedge \text{(RI)} \wedge \text{(RRS)} \wedge \text{(RRI0)}.$$

[5] Defined in Lemma 14.

Proof We have already shown (Lemma 16) that (HE) implies (RSV). From this, Lemma 14 shows ! is surjective. It remains to show (RI), (RRS) and (RRI0).

(RI) is proved by double induction on the structures of A and B, using Lemma 8(2) (which depends on (HF)) and equivariance of ! (which depends on (HE)).

(RRS) is proved by induction on the structure of A, using (RSV), (HP) and a strengthened induction principle over the structure of A. In the case where A is some $[Y]A'$, this strengthened induction principle assures (by α-conversion) that $Y \sharp$ (X,B). See [2, 15, 16, 23, 25, 27] for discussion of such strengthened induction principles.

(RRI0) Unfolding definitions, we must show

$$!(A[X::=Y]) \;=\; (\mathsf{abs}_X !A) \triangledown Y$$

Using (HF) and Lemma 7 this follows from (RRS), which we already know holds from (RSV) and (HP). □

4.2 Good F are Required for Adequate Representation

We present a converse to Theorem 1.

Theorem 2 (RSV) $\wedge$ (RRS) $\wedge$ (RRI0) $\Longrightarrow$ (HE) $\wedge$ (HP) $\wedge$ (HF).

There are two weaknesses with the statement of this theorem. First, we really seem to need assumption (RRI0) to prove it although it isn't clear why (RRI0) is part of the notion of adequacy. (One might think that (RRS) $\Longrightarrow$ (RRI0) but we have not been able to prove that.) Second, although (RI) clearly is part of the notion of adequacy, it is not required for this proof. (Thus, via Theorem 1, (RI) is not independent of (RSV), (RRS) and (RRI0).) First we prove a lemma.

Lemma 17 *Assume* (RSV) *and* (RRS)*, then* ! *is equivariant:* $\pi \cdot !A = !\pi \cdot A$.

Proof Any function $f : \mathsf{n}\mathbb{L} \to \mathbb{L}$ that preserves substitution is equivariant. To see this note that every permutation π is a composition of name swaps, (X, Y), so it suffices to show that f preserves swaps:

$$(X, Y)\cdot f A = f((X, Y)\cdot A).$$

Picking $Z \sharp (X, Y, A, M)$, swapping can be expressed in terms of substitution:

$$(X, Y)\cdot A = A[Y::=Z][X::=Y][Z::=X]$$

$$(X, Y)\cdot !A = [X/Z][Y/X][Z/Y]!A$$

Since f preserves substitutions, the lemma follows. □

Proof (of theorem 2) From (RSV) we know that the function ! satisfies Lemma 15. From Lemma 17 we know that ! is equivariant. (HE) follows from Lemma 16.

For (HP), assuming $(M, N) : \mathbb{L}$ and $X \sharp (Y, N)$ we must show

$$\mathsf{F}_X(M) \;=\; \mathsf{F}_X([N/Y]M).$$

Let $M = !A$ and $N = !B$; hence also $X \sharp B$ (Lemma 14). Using (13), it suffices to note:

$$\begin{aligned}[N/Y]\mathsf{abs}_X M &= [!B/Y]\mathsf{abs}_X !A \\ &= [!B/Y]!\,[X]\,A \qquad \text{using Lemma 15} \\ &= !(([X]\,A)[Y::=B]) \qquad \text{using (RRS)} \\ &= !([X]\,(A[Y::=B])) \qquad \text{using } X \sharp (Y, B) \\ &= \mathsf{abs}_X([!B/Y]!A) \qquad \text{using Lemma 15 and (RRS)} \\ &= \mathsf{abs}_X([N/Y]M).\end{aligned}$$

It is interesting to note that in the presence of (RSV), (HP) $\Longleftrightarrow$ (RRS).

For (HF), using Lemma 7 and assuming $M : \mathbb{L}$ (so $\mathsf{LV}(M) = \{\}$), it suffices to show

$$[Z/X]M \;=\; (\mathsf{abs}_X M) \triangledown Z \qquad \text{for some } Z\,.$$

Letting $M = !A$ we have

$$\begin{aligned}[Z/X]M &= !(A[X::=Z]) \qquad \text{using (RRS)} \\ &= !(([X]\,A)\blacktriangledown Z) \\ &= (!([X]\,A))\triangledown !Z \qquad \text{using (RRI0)} \\ &= (\mathsf{abs}_X M)\triangledown Z\end{aligned}$$

as desired. □

4.3 Example: β-Reduction

We can define β-reduction over $\mathbb{L}$ by the rules:

$$\frac{P : \mathbb{L} \qquad N : \mathbb{L}}{((\mathsf{abs}_X P)\;\; N) \to_\beta (\mathsf{abs}_X P)\triangledown N}\;(\beta)$$

$$\frac{M_1 \to_\beta M_2 \qquad N : \mathbb{L}}{(M_1\;\; N) \to_\beta (M_2\;\; N)} \qquad\qquad \frac{M : \mathbb{L} \qquad N_1 \to_\beta N_2}{(M\;\; N_1) \to_\beta (M\;\; N_2)}$$

$$\frac{M \to_\beta N}{\mathsf{abs}_X M \to_\beta \mathsf{abs}_X N}\;(\xi)$$

Notice how the uses of notations abs and $\triangledown$ in rules (ξ) and (β) abstract details. For example, the expanded form of rule (ξ)

$$\frac{M \to_\beta N \qquad x = \mathsf{F}_X(M) \qquad y = \mathsf{F}_X(N)}{[x]\,[x/X]M \to_\beta [y]\,[y/X]N}\;(\xi)$$

shows that the bound variables x and y need not be the same, as indeed careful reading of nominal Isabelle notation for this rule [4], and even informal notation show. However unlike nominal Isabelle, our underlying abstraction constructor of $\mathbb{S}$ is injective.

For good F, this definition of $\to_\beta$ is well behaved; e.g.

$$\begin{aligned}
M \to_\beta N &\iff \pi{\cdot}M \to_\beta \pi{\cdot}N,\\
M \to_\beta N &\implies M : \mathbb{L} \wedge N : \mathbb{L},\\
M \to_\beta N \wedge X \sharp M &\implies X \sharp N.
\end{aligned}$$

For good F, the representation w.r.t. nominal Isabelle lambda terms respects β-reduction:

$$A \to_\beta B \iff {!A} \to_\beta {!B}.$$

(See [4] for the definition of β-reduction on nominal Isabelle lambda terms.)

5 A Different Example: The Multivariate Lambda Calculus

In this section we outline a formalization of the *multivariate lambda calculus* of Pottinger [20]: "[...] a single λ may bind an arbitrary finite sequence of variables. This introduces terms of the form $\lambda x_0 \ldots x_{n-1}.X$ which are *not* the result of performing n univariate abstractions. For example, we have $\lambda xy.x \neq \lambda x.\lambda y.x$. Redexes have the form $(\lambda x_0 \ldots x_{n-1}.X)Y_0 \ldots Y_{n-1}$, and such a redex contracts to the result of simultaneously substituting $Y_0, \ldots, Y_{n-1}$ for $x_0, \ldots, x_{n-1}$ in X." The reason for independent interest in this system is that, because reduction waits for enough arguments, it gives a better notion of *combinator* than ordinary lambda calculus. For our purposes, formalization of multiple binding and simultaneous substitution are the interesting aspects.

As in previous sections we use $\mathbb{V}$ for the set of local (bindable) variables (ranged over by x, y, ...), and $\mathbb{X}$ for the set of global names (ranged over by X, Y, ...). Let $\mathbb{X}s$ be the set of lists of atoms (ranged over by Xs, Ys, ...)

We distinguish between symbolic expressions for *value terms*, $\mathbb{VS}$ (ranged over by A, B, ...) and, mutually defined, symbolic expressions for general terms $\mathbb{S}$, (ranged over by M, N, ...). Let $\mathbb{S}s$ be the set of lists of symbolic expressions (ranged over by Ms, Ns, ...). Finally, let m, n, ...range over natural numbers that will be used as indexes into lists. We use the notation Xs_n and Ms_n for the n^{th} element of the indicated list.

The syntax of $\mathbb{S}$ and is given by the mutual definition:

$$\frac{}{X : \mathbb{VS}} \qquad \frac{}{(x,n) : \mathbb{VS}} \qquad \frac{M : \mathbb{S}}{[x,n]\, M : \mathbb{VS}} \qquad \frac{A : \mathbb{VS} \quad Ns : \mathbb{S}s}{(A \;\; Ns) : \mathbb{S}}$$

Notice that application to a null list makes a value expression into an expression: $(A \;\; [])$: $\mathbb{S}$. The value expression $[x,n]\, M$ results from abstracting a list of n global names from expression M. The local variable (x,n) refers to the nth member of the list abstracted by x. We will make this precise.

Define the local variables and global variables of an expression $\mathsf{LV}(M)$ and $\mathsf{GV}(M)$ in the obvious way. We write $Xs \sharp M$ to mean that no member of Xs occurs in $\mathsf{GV}(M)$.

Replacement of Variables Simultaneous substitutions are concretely represented as association lists, i.e. lists of $\mathbb{X} \times \mathbb{S}$ pairs. The variable σ ranges over substitutions. We assume the standard lookup operation on association lists, saying that $(X, M) \in \sigma$ when (X, M) is the first pair in σ whose first component is X, and $X \notin \mathsf{dom}(\sigma)$ if no such pair exists. We will also sometimes write a substitution by giving its typical element: $[M_i / X_i]$.

The action of a substitution is defined by:

$$\begin{aligned}
[\sigma]Y &\triangleq \begin{cases} M & \text{if } (Y, M) \in \sigma, \\ Y & \text{if } Y \notin \mathsf{dom}(\sigma). \end{cases} \\
[\sigma](x, n) &\triangleq (x, n) \\
[\sigma]([x, n]\, M) &\triangleq [x, n]\, ([\sigma]M) \\
[\sigma]\,(A \;\; Ns) &\triangleq ([\sigma]A \;\; ([\sigma]Ns))
\end{aligned}$$

Replacement of local variables is defined by:

$$\begin{aligned}
[Ms/y]X &\triangleq X \\
[Ms/y](x, n) &\triangleq \begin{cases} Ms_n & \text{if } x = y, \\ (x, n) & \text{if } x \neq y. \end{cases} \\
[Ms/y]\,(A \;\; Ns) &\triangleq ([Ms/y]A \;\; [Ms/y]Ns) \\
[Ms/y]\,[x, n]\, M &\triangleq \begin{cases} [x, n]\, M & \text{if } x = y, \\ [x, n]\, [Ms/y]M & \text{if } x \neq y. \end{cases}
\end{aligned}$$

By writing Ms_n in the second case above, we are being loose about 'arities', but the definition will only be used in correct cases.

5.1 Well Formed Multivariate Lambda Terms

The type of height functions is $\mathbb{X}s \times \mathbb{S} \to \mathbb{V}$. Suppose $Xs = [X_0, \ldots, X_{n-1}]$, and let $f : \mathbb{V}$ abbreviate $\mathsf{F}_{Xs}(M)$ for some height function F. Suppose a symbolic expression $M = \ldots X_i \ldots X_j \ldots$ contains some of the X_i. We want abstraction to behave as follows:

$$\mathsf{abs}_{Xs} M \;=\; [\, f, n\,]\; \ldots (f, i) \ldots (f, j) \ldots .$$

Notice that the abstraction carries the length of the vector of abstracted global names, and each occurrence of an abstracted global name in the body is replaced by the (single) local name chosen by $\mathsf{F}_{Xs}(M)$ to be the binding name, along with the appropriate index into the vector of abstracted global names. One complication arises: if there are duplicates in Xs we intend the innermost occurrence (biggest index) to bind. Thus, given that we use the first matching item in a substitution, we should reverse the substitution. Taking

$$\sigma = [(f, i)/X_i]_{i=(n-1)\ldots 0}$$

we define abstraction by:

$$\mathsf{abs}_{Xs} M \triangleq [\, f, \mathsf{lngth}(Xs)\,]\, [\sigma]M.$$

Abstraction of a vector of global names is truly an atomic operation, rather than an iteration of unitary abstractions, as is described, for example, in [3]. It is worth mentioning that this approach to atomically abstracting a vector of names appears in [2] and the details were worked out in that setting by Arthur Charguéraud.

Now we can define the well formed multivariate lambda terms $\mathbb{L}$, and well formed multivariate lambda values $\mathbb{VL}$, as mutually inductive properties of symbolic expressions:

$$\frac{}{X : \mathbb{VL}_{\mathsf{F}}} \qquad \frac{M : \mathbb{L}_{\mathsf{F}}}{\mathsf{abs}_{Xs} M : \mathbb{VL}_{\mathsf{F}}} \qquad \frac{A : \mathbb{VL}_{\mathsf{F}} \quad \forall N \in Ns.\ N : \mathbb{L}_{\mathsf{F}}}{(A\ \ Ns) : \mathbb{L}_{\mathsf{F}}}$$

As before, we drop the explicit parameterization of $\mathbb{L}_{\mathsf{F}}$ by F. We want to prove the analogue, in this setting, of (6), Lemma 19 below. As before, we consider properties of height functions.

5.2 Well Behaved Height Functions

First, lift the definition of E to the current setting:

$$\begin{aligned}
\mathsf{E}_{Xs}(Y) &\triangleq \{\} \\
\mathsf{E}_{Xs}(y) &\triangleq \{\} \\
\mathsf{E}_{Xs}((A\ \ Ns)) &\triangleq \mathsf{E}_{Xs}(A) \cup \left(\bigcup_{N \in Ns} \mathsf{E}_{Xs}(N) \right) \\
\mathsf{E}_{Xs}([x,n]\,M) &\triangleq \begin{cases} \{\} & \text{if } Xs \sharp M\text{: no paths to any } X \text{ in } M \\ \{x\} \cup \mathsf{E}_{Xs}(M) & \text{otherwise: } x \text{ in every path} \end{cases}
\end{aligned}$$

The three properties of an excellent height function.

$$\begin{array}{lrcl}
\text{(XHE)} & \mathsf{F}_{Xs}(M) = \mathsf{F}_{\pi\cdot Xs}(\pi\cdot M) & & \mathsf{F}\text{ equivariant,} \\
\text{(XHF)} & \mathsf{F}_{Xs}(M) \notin \mathsf{E}_{Xs}(M) & & \mathsf{F}\text{ fresh,} \\
\text{(XHP)} & Xs \sharp \sigma \implies \mathsf{F}_{Xs}(M) = \mathsf{F}_{Xs}([\sigma]M) & & \mathsf{F}\text{ preserved by substitution.}
\end{array}$$

Lemma 18 *There exists an excellent height function.*

Proof We adapt the excellent height function from Section 3.1.2:

$$\begin{aligned}
\mathsf{H}_{Xs}(Y) &\triangleq \begin{cases} 1 & \text{if } Y \in Xs \\ 0 & \text{if } Y \notin Xs \end{cases} \\
\mathsf{H}_{Xs}((x,n)) &\triangleq 0 \\
\mathsf{H}_{Xs}((A\ \ Ns)) &\triangleq \max\left\{\mathsf{H}_{Xs}(A), \mathsf{H}_{Xs}(Ns)\right\} \\
\mathsf{H}_{Xs}([x,n]\,M) &\triangleq \begin{cases} \mathsf{H}_{Xs}(M) & \text{if } \mathsf{H}_{Xs}(M) = 0 \text{ or } \mathsf{H}_{Xs}(M) > x \\ x+1 & \text{otherwise} \end{cases} \\
\mathsf{H}_{Xs}(Ns) &\triangleq \max\left\{\mathsf{H}_{Xs}(N) \mid N \in Ns\right\}
\end{aligned}$$

Lemma 19 *Assume* F *has* (XHE) *and* (XHP). *Then substitution is well behaved on well formed terms:*

$$M : \mathbb{L} \ \wedge\ (\forall (X,N) \in \sigma.\ N : \mathbb{L}) \implies [\sigma]M : \mathbb{L}.$$

$$\frac{N_0 \to_\beta N_0' \quad \forall N \in Ns.\ N : \mathbb{L}}{N_0{::}Ns \to_\beta N_0'{::}Ns} \qquad \frac{N_0 : \mathbb{L} \quad Ns \to_\beta Ns'}{N_0{::}Ns \to_\beta N_0{::}Ns'}$$

$$\frac{A \to_\beta A' \quad N : \mathbb{L}}{(A\ \ N) \to_\beta (A'\ \ N)} \qquad \frac{A : \mathbb{L} \quad Ns \to_\beta Ns'}{(A\ \ Ns) \to_\beta (A\ \ Ns')} \qquad \frac{M \to_\beta N}{\mathsf{abs}_{Xs} M \to_\beta \mathsf{abs}_{Xs} N}\ (\xi)$$

Fig. 1 Congruence rules for multivariate β -reduction

Proof By (strengthened) induction on $M : \mathbb{L}$. □

We define the notion of *instantiation*, $\triangledown : \mathbb{S} \times \mathbb{S}s \to \mathbb{S}$:

$$([x, n]\ M) \triangledown Ns \triangleq [Ns/x]M.$$

This is only used when $\mathsf{lngth}(Ns) = n$.

Lemma 20 *For excellent* F*, and* $\mathsf{lngth}(Xs) = \mathsf{lngth}(Ns)$*, we have*

$$M : \mathbb{L}\ \wedge\ (\forall N \in Ns.\ N : \mathbb{L}) \implies (\mathsf{abs}_{Xs} M) \triangledown Ns : \mathbb{L}.$$

5.3 β -Reduction

There is still one remaining complication. Applications of an abstraction to a vector of terms that is too short do not contract. But applications to a vector of terms that is too *long* do contract, with some arguments left over. This shows up in rule (β), where ' @ ' is list concatenation and lngth is the length function on lists.[6]

$$\frac{\begin{array}{ccc} M : \mathbb{L} & Ns = Ns^a @ Ns^b & \\ \forall N \in Ns.\ N : \mathbb{L} & \mathsf{lngth}(Xs) = \mathsf{lngth}(Ns^a) & (\mathsf{abs}_{Xs} M) \triangledown Ns^a = (A\ \ Qs) \end{array}}{((\mathsf{abs}_{Xs} M)\ \ Ns) \to_\beta (A\ \ (Qs @ Ns^b))}\ (\beta)$$

We have to explain the meta-typing of rule (β). Instantiating the abstraction $\mathsf{abs}_{Xs} M$ with the right number of arguments gives $(\mathsf{abs}_{Xs} M) \triangledown Ns^a$, a symbolic expression, not a symbolic value. So it has shape $(A\ \ Qs)$ (for some A and Qs, with Qs possibly null). Then the contractum in rule (β) is $(A\ \ (Qs @ Ns^b))$, carrying along the leftover arguments. The congruence rules (Fig. 1) use an auxiliary judgement $Ns \to_\beta Ns'$ saying that exactly one expression in a list of expressions reduces (' :: ' is list cons).

Lemma 21 *Assume* F *is excellent.* β *reduction is well behaved:*

$$M \to_\beta N \iff \pi \cdot M \to_\beta \pi \cdot N,$$

$$M \to_\beta N \implies M : \mathbb{L} \wedge N : \mathbb{L},$$

$$M \to_\beta N \wedge X \sharp M \implies X \sharp N.$$

[6] To be completely precise, 'Xs' doesn't actually occur in '$\mathsf{abs}_{Xs} M$', so the occurrence of '$\mathsf{lngth}(Xs)$' in the premises of rule (β) is abuse of notation. However $\mathsf{abs}_{Xs} M$ does carry $\mathsf{lngth}(Xs)$ in its official syntax as defined above, so the rule can be made formal.

6 Conclusion

In [23] we presented a concrete canonical approach to name-carrying representation for binding. We also showed some simple applications of our approach to beta reduction and simple type assingnment. In the present paper we address exactly what is needed of an abstract canonical choice of binding names (F) to make this approach work.

We give three properties of a good height function F that guarantee F gives an adequate representation of pure lambda terms, defined as a substitution and instantiation preserving isomorphism with nominal lambda terms. Unfortunately the need to use (RRI0) in this proof does not seem intuitively natural.

Should the user interested in actually reasoning about some language with binding be interested in this paper? Not very much. Such a user could just use the concrete height function H of Lemma 10. An advantage of doing this is that one can actually compute with H, and do case analysis in proofs about H. But the disadvantage is that one is tempted to always do such case analysis when more general principles are simpler to reason about. Thus we recommend that a user take the abstract approach with an unspecified height function F satisfying the 'excellent' properties of Section 3.1.1. In this way one avoids petty and unnecessary reasoning about relativisation of the 'good' properties to $\mathbb{L}_{\mathsf{F}}$ without committing to concrete reasoning about a particular height function. Our formalization of the multivariate lambda calculus in Section 5 takes this approach, and shows that our approach applies to systems with considerably more challenging syntax than pure lambda calculus.

References

1. Ambler, S.J., Crole, R.L., Momigliano, A.: A definitional approach to primitive recursion over higher order abstract syntax. In: MERLIN '03: Proceedings of the 2003 Workshop on Mechanized Reasoning About Languages with Variable Binding, pp. 1–11. ACM Press (2003)
2. Aydemir, B., Charguéraud, A., Pierce, B.C., Pollack, R., Weirich, S.: Engineering formal metatheory. In: Proceedings of the 35th Annual ACM SIGPLAN-SIGACT Symposium on Principles on Programming Languages, pp. 3–15. ACM Press (2008)
3. Bengtson, J., Parrow, J.: Psi-calculi in isabelle. In: TPHOLs. LNCS, vol. 5674 (2009)
4. Berghofer, S., Urban, C.: Nominal inversion principles. In: Theorem Proving in Higher Order Logics, TPHOLs 2008. LNCS. Springer-Verlag (2008)
5. Curry, H.B., Feys, R.: Combinatory Logic, vol. 1. North Holland (1958)
6. de Bruijn, N.G.: Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation, with application to the Church-Rosser theorem. Indag. Math., **34**(5), 381–392 (1972)
7. Frege, G.: Begriffsschrift, eine der arithmetischen nachgebildete Formelsprache des reinen Denkens. Halle (1879) (Translated in van Heijenoort, J.: From Frege to Gödel: a source book in mathematical logic, 1879–1931, pp. 1-82. Harvard University Press, Cambridge, MA (1967))
8. Gabbay, M., Pitts, A.: A new approach to abstract syntax involving binders. In: Longo, G. (ed.) Proceedings of the 14th Annual Symposium on Logic in Computer Science (LICS'99), pp. 214–224 (1999)
9. Gentzen, G.: Untersuchungen über das logische schliessen. Math. Zeitschrift **39**, 176–210 (1934) (English translation in Szabo, M.E. (ed.): The Collected Papers of Gerhard Gentzen. North Holland (1969))
10. Gordon, A.: A mechanism of name-carrying syntax up to alpha-conversion. In: Higher Order Logic Theorem Proving and its Applications. Proceedings, 1993. LNCS 780, pp. 414–426. Springer-Verlag (1993)

11. Gordon, A., Melham, T.: Five axioms of alpha conversion. In: Von Wright, J., Grundy, J., Harrison, J. (eds.) Ninth Conference on Theorem Proving in Higher Order Logics TPHOL'96, Turku. LNCS, vol. 1125, pp. 173–190. Springer-Verlag (1996)
12. Harper, R., Honsell, F., Plotkin, G.: A framework for defining logics. J. ACM **40**(1), 143–184 (1993) (Preliminary version in LICS'87)
13. Harper, R., Licata, D.R.: Mechanizing metatheory in a logical framework. J. Funct. Program. **17**(4–5) (2007)
14. Honsell, F., Miculan, M., Scagnetto, I.: The theory of contexts for first order and higher order abstract syntax. Electronic Notes Theor. Comp. Sci. **62**, 116–135 (2002)
15. McKinna, J., Pollack, R.: Pure type systems formalized. In: Bezem, M., Groote, J.F. (eds.) Proceedings of the International Conference on Typed Lambda Calculi and Applications, TLCA'93, Utrecht. LNCS, number 664, pp. 289–305. Springer-Verlag (1993)
16. McKinna, J., Pollack, R.: Some lambda calculus and type theory formalized. J. Autom. Reason. **23**(3–4), 373–409 (1999)
17. Pfenning, F., Schürmann, C.: System description: twelf: a meta-logical framework for deductive systems. In: Proceedings of the 16th International Conference on Automated Deduction (CADE-16). LNAI, Springer-Verlag (1999)
18. Pitts, A.M.: Nominal logic, a first order theory of names and binding. Inf. Comput. **186**, 165–193 (2003)
19. Pollack, R.: The theory of LEGO: a proof checker for the extended calculus of constructions. Ph.D. thesis, Univ. of Edinburgh (1994)
20. Pottinger, G.: A tour of the multivariate lambda calculus. In: Dunn, J.M., Gupta, A. (eds.) Truth or Consequences: Essays in Honor of Nuel Belnap. Kluwer (1990)
21. Prawitz, D.: Natural Deduction: Proof Theoretical Study. Almquist and Wiksell, Stockholm (1965)
22. Sato, M.: External and internal syntax of the λ-calculus. In: Buchberger, B., Ida, T., Kutsia, T. (eds.) Proc. of the Austrian-Japanese Workshop on Symbolic Computation in Software Science, SCSS 2008. RISC-Linz Report Series, number 08–08, pp. 176–195 (2008)
23. Sato, M., Pollack, R.: External and internal syntax of the λ-calculus. J. Symb. Comput. **45**, 598–616 (2010)
24. Stoughton, A.: Substitution revisited. Theor. Comp. Sci. **17**, 317–325 (1988)
25. Urban, C., Berghofer, S., Norrish, M.: Barendregt's variable convention in rule inductions. In: Automated Deduction—CADE-21. LNCS, number 4603, pp. 35–50. Springer-Verlag (2007)
26. Urban, C.: Nominal techniques in isabelle/hol. J. Autom. Reason. **40**(4), 327–356 (2008)
27. Urban, C., Pollack, R.: Strong induction principles in the locally nameless representation of binders (preliminary notes). Presented at (ACM) Workshop on Mechanizing Metatheory (2007)
28. van Heijenoort, J.: From Frege to Gödel: A Source Book in Mathematical Logic, 1879–1931. Harvard University Press, Cambridge, MA (1967)