Nominal renaming sets

Murdoch J. Gabbay[1] and Martin Hofmann[2]

[1] http://www.gabbay.org.uk
[2] http://www.tcs.informatik.uni-muenchen.de/~mhofmann/

Abstract. Nominal techniques are based on the idea of sets with a finitely-supported atoms-permutation action.
We consider the idea of *nominal renaming sets*, which are sets with a finitely-supported atoms-renaming action; renamings can identify atoms, permutations cannot. We show that nominal renaming sets exhibit many of the useful qualities found in (permutative) nominal sets; an elementary sets-based presentation, inductive datatypes of syntax up to binding, cartesian closure, and being a topos. Unlike is the case for nominal sets, the notion of names-abstraction coincides with functional abstraction. Thus we obtain a concrete presentation of sheaves on the category of finite sets in the form of a category of sets with structure.

Keywords: Nominal renaming sets, nominal sets, abstract syntax with binding.

1 Introduction

The motivation of this work is to provide semantic foundations for formal theories of syntax with variable binding. Several such theories have been proposed in the literature [5,17,16,14,19] and used in concrete applications with varying success [15,4,25,18,21]. All but the most elementary approaches require some semantical or proof-theoretical justification asserting their soundness and also explaining the meaning of the judgements made. Functor categories (categories of (pre)sheaves) are a popular method [7,16,3,8] for providing such foundations. However, proofs involving presheaves are notoriously complicated and require a thorough working knowledge so as to be able to reduce clutter by elision of trivial steps. In some situations [21,25], a sets-based semantics is available in the form of nominal sets [14] which considerably simplifies metatheoretic reasoning. A domains version of nominal sets [20] has also become an important tool in denotational semantics of languages with dynamic allocation [1]

This paper provides a sets-based foundation for another class of systems for higher-order syntax including in particular the *theory of contexts* [17] which could hitherto only be modelled with functor categories. This will allow us to considerably simplify the cumbersome functor-theoretic proof of soundness of that theory [3]. In this way, we expect that our sets-based presentation of the semantics will then allow to justify further extensions which would until now be too complicated to work through.

Note that we do not propose yet another approach to higher-order syntax. Rather, we introduce a simplified presentation, up to categorical equivalence, of an existing semantic model. We now summarise the contributions of the paper in more detail.

Write $\mathbb{I}$ for the category of finite sets and injections between them, write $\mathbb{F}$ for the category of finite sets and all (not necessarily injective) functions between them, and write Set for the category of sets[3] and functions between them.

Previous work presented nominal sets [13]. These can be presented as the category of pullback-preserving presheaves in $\mathsf{Set}^{\mathbb{I}}$. Simultaneously, Fiore et al [7] and Hofmann [16] proposed to use $\mathsf{Set}^{\mathbb{F}}$ for higher-order abstract syntax. Both can be used as mathematical models for inductive specification and reasoning on syntax with binding.

In $\mathsf{Set}^{\mathbb{F}}$ for a presheaf F, the presheaf F^+ given by $F^+(X) = F(X \cup \{x\})$ for $x \notin X$ is isomorphic to the exponential $\mathbf{A} \Rightarrow F$. In the category of nominal sets the corresponding presheaf is isomorphic to $\mathbf{A} \multimap F$, with $\multimap$ being the right adjoint to a tensor product different from cartesian product (it is mentioned in Theorem 34 and is written $[\mathbf{A}]\mathbf{X}$). In either case the presheaf $\mathbf{A}$ is given by $\mathbf{A}(S) = S$. Thus, $\mathsf{Set}^{\mathbb{F}}$ seems better-suited for modelling higher-order abstract syntax, which uses typings like

$$\mathit{lam} : (\mathit{var} \to \mathit{tm}) \to \mathit{tm} \tag{1}$$

for variable binding constructs (in this case: 'lambda'). This arises when using an existing theorem prover or type theory (for example Coq) to model higher-order abstract syntax [6,17]. In particular, in [3] the presheaf category from [16] was used to establish soundness of the theory of contexts from [17].

FreshML [23] and the deep embeddings of the nominal datatypes package [25] provide a syntactic primitive corresponding to the right adjoint $\multimap$ mentioned above. They use typings like

$$\mathit{lam} : (\mathit{var} \multimap \mathit{tm}) \to \mathit{tm}. \tag{2}$$

Precomposing with the canonical map $(A \to B) \longrightarrow (A \multimap B)$, one could justify the typing in (1), but *lam* would not be injective. For the typing

$$\mathit{lam} : (\mathit{tm} \to \mathit{tm}) \to \mathit{tm} \tag{3}$$

used in the Twelf system [19], another presheaf topos ($\hat{\mathbb{S}}$ in [16]) is adequate. The constructions of this paper cannot be applied to this directly, but see the Conclusions.

The nominal approach benefits from a *sets-based* presentation. Denotations of types are sets with structure — whence the name 'nominal *sets*' — rather than functors.

In this paper we offer a sets-based presentation of the full subcategory of $\mathsf{Set}^{\mathbb{F}}$ of functors preserving pullbacks of monos. We present it as a category of sets with a not-necessarily-injective *renaming action*. The pullback requirement implies that $F(X \cap Y)$ is isomorphic to 'the intersection of' $F(X)$ and $F(Y)$, in line with the intuition 'objects with free variables from X' for $F(X)$. This rules out artefacts present in $\mathsf{Set}^{\mathbb{F}}$ like $F(X) = \mathit{if}\ |X| > 1\ \mathit{then}\ \{\star\}\ \mathit{else}\ \emptyset$, which never arise as denotations of types in higher-order abstract syntax.

In [16] it was argued that the internal logic of $\mathsf{Set}^{\mathbb{F}}$ qua topos might be unsuited to reasoning with higher-order abstract syntax since equality of atoms (names, variables) is not decidable in it. It was proposed in *loc. cit.* to import the logic of the Schanuel topos

[3] It is convenient, but not necessary, to take this to be the category of 'ordinary' ZF sets (not mentioning atoms). Since we are working at the meta-level, this is in fact not important and any sufficiently rich collection of collections of elements will do.

using a pullback of triposes. Our presentation of this pullback construction amounts to interpreting predicates as *equivariant subsets*; subsets of a renaming set stable under injective renamings. Staton and Fiore [8] define a category of substitution sets equivalent to a sheaf subcategory of $\mathsf{Set}^{\mathbb{F}}$. Their category is not defined concretely in terms of ordinary sets but as a theory within the Schanuel topos. It does not make $\mathsf{Set}^{\mathbb{F}}$ easier to work with (and Staton and Fiore never introduced it for that purpose).

We also characterise the function space of nominal renaming sets as a set of functions, rather than a Kripke exponential transported along the equivalence, and we identify the tripos structure needed to reason about renaming sets in a robust way, and in particular, to justify the theory of contexts.

Some notation: This paper uses three different kinds of arrow. For the reader's convenience we survey their definitions; this is all standard.

– If X and Y are sets then $X \to Y$ is the set of functions from X to Y.

– If $\mathbf{X}$ and $\mathbf{Y}$ are nominal renaming sets then $\mathbf{X} \longrightarrow \mathbf{Y}$ is the set of arrows from $\mathbf{X}$ to $\mathbf{Y}$ in the category Ren (Definition 5). These are maps in $|\mathbf{X}| \to |\mathbf{Y}|$ with empty support (Lemma 25).

– $\mathbf{X} \Rightarrow \mathbf{Y}$ is the exponential (Definition 20 and Theorem 26). These are maps in $|\mathbf{X}| \to |\mathbf{Y}|$ with finite support (Lemma 21).

2 Nominal renaming sets

Definition 1. *Fix a countably infinite* set of atoms $\mathbf{A}$. *We assume that atoms are disjoint from numbers* $0, 1, 2, \ldots$, *truth-values* $\bot, \top$, *and other standard mathematical entities.* $\mathbf{A}$ *can be viewed as a set of* urelemente *[2,13,10]; we view them as a model of* names *or* variable symbols.

$a, b, c, \ldots$ *will range over atoms. We follow a* permutative convention *that simultaneously introduced metavariables for atoms range* permutatively*; for example* a *and* b *range over any two* distinct *atoms.*[4]

Definition 2. *Let* Fin *be the set of functions* $\sigma \in \mathbf{A} \to \mathbf{A}$ *such that there exists some finite* $S \subseteq \mathbf{A}$ *such that for all* $b \in \mathbf{A} \setminus S$ *it is the case that* $\sigma(b) = b$.

σ, τ will range over elements of Fin. We call these *(finitely supported) renamings*.

Definition 3. *Write* $[a_1 \mapsto y_1, \ldots, a_k \mapsto y_k]$ *for the function that maps* a_i *to* y_i *for* $1 \leq i \leq k$ *and maps all other* b *(that is, atoms* b *not in the set* $\{a_1, \ldots, a_k\}$*) to themselves. Note that every function in* Fin *can be written in this fashion.*

In particular, write $[a \mapsto b]$ *for the function which 'maps* a *to* b*':*

$$[a \mapsto b](a) = b \qquad [a \mapsto b](b) = b \quad \textit{and} \quad [a \mapsto b](c) = c$$

We write $\circ$ *for functional composition. For example* $[a \mapsto b] \circ [b \mapsto a] = [a \mapsto b]$ *(and* $[a \mapsto b] \circ [b \mapsto a] \neq [a \mapsto b, b \mapsto a]$*). Write* id *for the* identity renaming. $id(a) = a$ *always.*

[4] Note that we are working in the usual naïve set-theoretic metalanguage; atoms form a set. At that level, atoms have a decidable equality.

Fin with $\circ$ and id is a monoid. That is, $id \circ \sigma = \sigma \circ id = \sigma$ and $\sigma \circ (\sigma' \circ \sigma'') = (\sigma \circ \sigma') \circ \sigma''$. If $S \subseteq \mathbf{A}$ and $\sigma \in Fin$, write $\sigma|_S$ for the partial function equal to σ on S and undefined elsewhere.

Definition 4. *A* nominal renaming set $\mathbf{X}$ *is a pair* $(|\mathbf{X}|, \cdot)$ *of an* underlying set $|\mathbf{X}|$ *and a* finitely-supported renaming action $\cdot$.

A finitely-supported renaming action $\cdot$ *is a map from* $Fin \times |\mathbf{X}|$ *to* $|\mathbf{X}|$ *such that:*

– $id \cdot x = x$ *always.*
– $\sigma' \cdot (\sigma \cdot x) = (\sigma' \circ \sigma) \cdot x$ *always.*
– *For every* $x \in |\mathbf{X}|$ *there is a finite* $S \subseteq \mathbf{A}$ *such that* $\sigma|_S = \sigma'|_S$ *implies* $\sigma \cdot x = \sigma' \cdot x$.
We say that every $x \in |\mathbf{X}|$ *has* finite support.

Henceforth $\mathbf{X}$ and $\mathbf{Y}$ range over nominal renaming sets. x and x' range over elements of $|\mathbf{X}|$ and y and y' range over elements of $|\mathbf{Y}|$ unless stated otherwise.

Definition 5. *Nominal renaming sets form a category* Ren. *Objects are nominal renaming sets. An arrow* $F : \mathbf{X} \longrightarrow \mathbf{Y}$ *is a function* $F \in |\mathbf{X}| \to |\mathbf{Y}|$ *such that* $\sigma \cdot F(x) = F(\sigma \cdot x)$ *always.* F, G, H *will range over arrows.*

For example: $\mathbf{A}$ with action $\sigma \cdot a = \sigma(a)$ is a nominal renaming set. We have $supp(a) = \{a\}$. We will write this renaming set by $\mathbf{A}$, too. In general, we will distinguish notationally between a renaming set $\mathbf{X}$ and its underlying set $|\mathbf{X}|$.

Call $\mathbf{X}$ *trivial* when $\sigma \cdot x = x$ for all $x \in |\mathbf{X}|$ and $\sigma \in Fin$ (so $supp(x) = \emptyset$ always). Write $\mathbf{B} = (\{\top, \bot\}, \cdot)$ for a two-element trivial nominal renaming set. We conclude with one more useful definition:

Definition 6. *Let* $\mathbf{X} \times \mathbf{Y}$ *have underlying set* $|\mathbf{X}| \times |\mathbf{Y}|$ *(that is,* $z \in |\mathbf{X} \times \mathbf{Y}|$ *is a pair* (x, y) *where* $x \in \mathbf{X}$ *and* $y \in \mathbf{Y}$*) and pointwise renaming action (that is,* $\sigma \cdot (x, y) = (\sigma \cdot x, \sigma \cdot y)$*). Call this the* product *of* $\mathbf{X}$ *and* $\mathbf{Y}$.

2.1 Support of nominal renaming sets

Definition 7. *Say* $S \subseteq \mathbf{A}$ supports x *when for all* σ, σ', *if* $\sigma|_S = \sigma'|_S$ *then* $\sigma \cdot x = \sigma' \cdot x$.

Lemma 8 and Theorem 9 echo [13, Proposition 3.4]. The proofs for nominal renaming set are simpler:

Lemma 8. *If* $S, S' \subseteq \mathbf{A}$ *support* x *then so does* $S \cap S'$.

Proof. Suppose $\sigma|_{S \cap S'} = \sigma'|_{S \cap S'}$. Define $\sigma'' \in Fin$ by: $\sigma''(a) = \sigma(a)$ if $a \in S$, and $\sigma''(a) = \sigma'(a)$ if $a \in \mathbf{A} \setminus S$. $\sigma''|_S = \sigma|_S$ so $\sigma'' \cdot x = \sigma \cdot x$. It is not hard to verify that $\sigma''|_{S'} = \sigma'|_{S'}$ so $\sigma'' \cdot x = \sigma' \cdot x$. The result follows.

Theorem 9. – x *has a unique finite least supporting set* $supp(x)$*; the* support *of* x.
– $\sigma|_{supp(x)} = \sigma'|_{supp(x)}$ *implies* $\sigma \cdot x = \sigma' \cdot x$.

Proof. There is a finite $S \subseteq \mathbf{A}$ supporting x. The first part follows by Lemma 8. The second part is then by Definition 7.

If the reader thinks of fv (free variables of) when they see $supp$, they will not go far wrong [13, Example 6.11]. However, support is an abstract notion valid for any element of any nominal renaming set.

As is standard in nominal techniques we write $a\#x$ for $a \notin supp(x)$, and read it as 'a is fresh for x'. We may write $a\#x, y$ for '$a\#x$ and $a\#y$', and so on.

Definition 10. *Let $PFin$ have underlying set the collection of finite sets of atoms, with pointwise renaming action.*

That is, if $S \subseteq \mathbf{A}$ is finite then $\sigma \cdot S = \{\sigma(a) \mid a \in S\}$. It is not hard to prove that $supp(S) = S$ always.

Lemma 11. *1. $supp(\sigma \cdot x) \subseteq \sigma \cdot supp(x)$.*
2. If σ is injective on $supp(x)$ then $supp(\sigma \cdot x) = \sigma \cdot supp(x)$.

Proof. For the first part, we will show that $\sigma \cdot supp(x)$ supports $\sigma \cdot x$. The claim then follows. To see the former suppose that $\sigma_1|_{\sigma \cdot supp(x)} = \sigma_2|_{\sigma \cdot supp(x)}$. We then have $(\sigma_1 \circ \sigma)|_{supp(x)} = (\sigma_2 \circ \sigma)|_{supp(x)}$, hence $\sigma_1 \cdot \sigma \cdot x = (\sigma_1 \circ \sigma) \cdot x = (\sigma_2 \circ \sigma) \cdot x = \sigma_2 \cdot \sigma \cdot x$.

For the second part, it suffices to prove the reverse inclusion. Suppose $\sigma|_{supp(x)} = [a_1 \mapsto y_1, \ldots, a_n \mapsto y_n]|_{supp(x)}$. By assumption if $y_i = y_j$ then $i = j$ for $1 \leq i, j \leq n$. So we can form $\sigma' = [y_1 \mapsto a_1, \ldots, y_n \mapsto a_n]$. By Theorem 9 $\sigma' \cdot \sigma \cdot x = x$. By part 1 of this result $supp(\sigma' \cdot \sigma \cdot x) \subseteq \sigma' \cdot supp(\sigma \cdot x)$. Thus, $\sigma \cdot supp(x) = \sigma \cdot supp(\sigma' \cdot \sigma \cdot x) \subseteq \sigma \cdot \sigma' \cdot supp(\sigma \cdot x)$.

Now, if $a \in supp(\sigma \cdot x)$ then by part 1 of this result we can write $a = \sigma(b)$ for some $b \in supp(x)$. So, $\sigma \cdot \sigma' \cdot a = \sigma \cdot \sigma' \cdot \sigma \cdot b = \sigma \cdot b = a$. We have thus shown $\sigma \cdot \sigma' \cdot supp(\sigma \cdot x) \subseteq supp(\sigma \cdot x)$ and hence the claim.

We remark that the extra assumption of injectivity for the second part is not redundant.

Corollary 12 helps to calculate support:

Corollary 12. *If $a \in supp(x)$ then $[a \mapsto b] \cdot x \neq x$. Taking the contrapositive, if $[a \mapsto b] \cdot x = x$ then $a\#x$. Note that $a \neq b$ is an implicit assumption.*

Proof. By part 1 of Lemma 11 $a \notin supp([a \mapsto b] \cdot x)$. We assumed $a \in supp(x)$, the result follows.

Lemma 13. *$S \subseteq \mathbb{A}$ supports x if and only if $\sigma|_S = id|_S$ implies $\sigma \cdot x = x$ for all σ.*

Proof. The left-to-right implication is direct from the definition of 'supports'. For the right-to-left implication we will show $supp(x) \subseteq S$. To that end, pick $a \in supp(x)$ and $b \notin supp(x)$. From Corollary 12 we then obtain $\sigma \cdot x \neq x$ for $\sigma = [a \mapsto b]$. Hence $\sigma|_S \neq id|_S$, hence $a \in S$.

We need Lemma 14 for Lemma 33 and Corollary 29:

Lemma 14. *Suppose $c\#x$ and $c\#x'$.*
Then $[a \mapsto c] \cdot x = [a' \mapsto c] \cdot x'$ if and only if $a'\#x$ and $x' = [a \mapsto a'] \cdot x$.

Proof. Suppose $a'\#x$ and $x' = [a{\mapsto}a'] \cdot x$. We reason as follows:

$$\begin{array}{lll} [a'{\mapsto}c] \cdot x' & = [a'{\mapsto}c] \cdot [a{\mapsto}a'] \cdot x & x' = [a{\mapsto}a'] \cdot x \\ & = [a{\mapsto}c] \cdot x & \text{Theorem 9, } a'\#x \end{array}$$

Conversely if $[a{\mapsto}c] \cdot x = [a'{\mapsto}c] \cdot x'$ then $[c{\mapsto}a'] \cdot [a{\mapsto}c] \cdot x = [c{\mapsto}a'] \cdot [a'{\mapsto}c] \cdot x'$. By Theorem 9 $[c{\mapsto}a'] \cdot [a'{\mapsto}c] \cdot x' = x'$ and $[c{\mapsto}a'] \cdot [a{\mapsto}c] \cdot x = [a{\mapsto}a'] \cdot x$. It follows that $x' = [a{\mapsto}a'] \cdot x$. By similar reasoning $x = [a'{\mapsto}a] \cdot x'$, and by part 1 of Lemma 11 it follows that $a'\#x$.

3 The exponential

Definition 15. *Let $|\mathbf{X} \Rightarrow \mathbf{Y}|$ be the set of functions $f \in |\mathbf{X}| \to |\mathbf{Y}|$ such that there exists some finite $S_f \subseteq \mathbf{A}$ (for each f, we fix one such S_f) such that for all $\sigma \in Fin$ and $x \in |\mathbf{X}|$ if $\sigma|_{S_f} = id|_{S_f}$ then*

$$\sigma \cdot f(x) = f(\sigma \cdot x). \tag{4}$$

$|\mathbf{X} \Rightarrow \mathbf{Y}|$ serves as an underlying set in Definition 20. First, we consider some examples and prove properties of $|\mathbf{X} \Rightarrow \mathbf{Y}|$.

Remark 16. $-\ \pi_1 \in |\mathbf{X} \times \mathbf{Y}| \to |\mathbf{X}|$ mapping (x, y) to x is in $|(\mathbf{X} \times \mathbf{Y}) \Rightarrow \mathbf{X}|$.
$-$ The map $= \in |\mathbf{A} \times \mathbf{A}| \to |\mathbf{B}|$ mapping (x, y) to $\top$ if $x = y$ and to $\bot$ if $x \neq y$, is *not* an element of $|(\mathbf{A} \times \mathbf{A}) \Rightarrow \mathbf{B}|$. Unpacking definitions, the reason is that there is no finite $S \subseteq \mathbf{A}$ such that if $\sigma|_S = id|_S$ then for all $x, y \in \mathbf{A}$, $x = y$ if and only if $\sigma(x) = \sigma(y)$.
$-$ $supp_{\mathbf{X}} \in |\mathbf{X}| \to |PFin|$ mapping x to $supp(x)$, may or may not be an element of $|\mathbf{X} \Rightarrow PFin|$. If $\mathbf{X} = \mathbf{A}$ then $supp(a) = \{a\}$ and $\sigma \cdot supp(a) = \sigma \cdot \{a\} = \{\sigma(a)\} = supp(\sigma(a))$. Therefore $supp_{\mathbf{A}} \in |\mathbf{A} \Rightarrow PFin|$. Similarly for $supp_{PFin}$.
On the other hand if we let $\mathbf{X}$ have $|\mathbf{X}| = |PFin|$ (finite sets of atoms) and the renaming action such that $\sigma \cdot S = \{\sigma(a) \mid a \in S\}$ if $\sigma|_S$ is injective, and $\sigma \cdot S = \emptyset$ otherwise, then $supp_{\mathbf{X}} \notin |\mathbf{X} \Rightarrow PFin|$.

Remark 17. Intuitively, a map that does not compare atoms in its argument for inequality will be in the underlying set of the exponential. A map that compares atoms for inequality, will not. See the Conclusions for further discussion.

Elements of $|\mathbf{X} \Rightarrow \mathbf{Y}|$ are determined by their 'asymptotic behaviour':

Lemma 18. *Suppose $f \in |\mathbf{X} \Rightarrow \mathbf{Y}|$ and $g \in |\mathbf{X} \Rightarrow \mathbf{Y}|$. Suppose also $S \subseteq \mathbf{A}$ is finite and assume that for all $x \in |\mathbf{X}|$ if $supp(x) \cap S = \emptyset$ then $f(x) = g(x)$. Then $f = g$.*

Proof. Choose any $x \in |\mathbf{X}|$. There are two cases:

$-$ If $supp(x) \cap S = \emptyset$ then by assumption $f(x) = g(x)$.

– If $supp(x) \cap S \neq \emptyset$ then let $C = \{c_1, \ldots, c_k\}$ be a fresh choice of atoms (so $C \cap (S \cup S_f \cup S_g \cup supp(x)) = \emptyset$). Let $\tau = [a_1 \mapsto c_1, \ldots, a_k \mapsto c_k]$ and $\tau^{\mathsf{T}} = [c_1 \mapsto a_1, \ldots, c_k \mapsto a_k]$. By Lemma 11 $supp(\tau \cdot x) \cap S = \emptyset$. We reason as follows:

$$\begin{aligned} f(x) &= f(\tau^{\mathsf{T}} \cdot \tau \cdot x) && \text{Theorem 9} \\ &= \tau^{\mathsf{T}} \cdot f(\tau \cdot x) && \text{(4), Definition 15} \\ &= \tau^{\mathsf{T}} \cdot g(\tau \cdot x) && \text{Assumption} \\ &= g(\tau^{\mathsf{T}} \cdot \tau \cdot x) && \text{(4), Definition 15} \\ &= g(x) && \text{Theorem 9} \end{aligned}$$

An element f of the function space can be reconstructed from 'asymptotic' behaviour:

Lemma 19. *Suppose f is a partial function from $|\mathbf{X}|$ to $|\mathbf{Y}|$. Suppose $S \subseteq \mathbf{A}$ is finite and: $supp(x) \cap S = \emptyset$ implies $f(x)$ is defined, and $\sigma|_S = id|_S$ implies $\sigma \cdot f(x) = f(\sigma \cdot x)$, where both are defined.*

Then there is a unique $f' \in |\mathbf{X} \Rightarrow \mathbf{Y}|$ extending f (so $f'(x) = f(x)$ if $f(x)$ is defined).

Proof. First, we define f'. Consider $x \in |\mathbf{X}|$, write $supp(x) = \{a_1, \ldots, a_k\} = S$. Let $C = \{c_1, \ldots, c_k\}$ be fresh atoms (so $C \cap supp(x) = \emptyset = C \cap S$). Define

$$\tau = [a_1 \mapsto c_1, \ldots, a_k \mapsto c_k] \quad \tau^{\mathsf{T}} = [c_1 \mapsto a_1, \ldots, c_k \mapsto a_k] \quad \text{and} \quad f'(x) = \tau^{\mathsf{T}} \cdot f(\tau \cdot x).$$

We first show the choice of fresh C does not matter. Suppose $C' = \{c'_1, \ldots, c'_k\}$ is also fresh (so $C' \cap supp(x) = \emptyset = C' \cap S$). We put $\tau' = [a_1 \mapsto c'_1, \ldots, a_k \mapsto c'_k]$ and ${\tau'}^{\mathsf{T}} = [c'_1 \mapsto a_1, \ldots, c'_k \mapsto a_k]$. We must show $\tau^{\mathsf{T}} \cdot f(\tau \cdot x) = {\tau'}^{\mathsf{T}} \cdot f(\tau' \cdot x)$. We assume the special case that

$$C \cap C' = \emptyset \quad \text{and} \quad C' \cap supp(f(\tau \cdot x)) = \emptyset. \tag{5}$$

The general case follows by two applications of the special case for an 'even fresher' set of fresh atoms C''. We write $\mu = [c_1 \mapsto c'_1, \ldots, c_k \mapsto c'_k]$ and $\mu^{\mathsf{T}} = [c'_1 \mapsto c_1, \ldots, c'_k \mapsto c_k]$. By Theorem 9 and $C \cap supp(x) = \emptyset$ we have $\tau' \cdot x = \mu \cdot \tau \cdot x$. Also, by $C' \cap S = \emptyset$ we have $f(\tau' \cdot x) = \mu \cdot f(\tau \cdot x)$. Therefore ${\tau'}^{\mathsf{T}} \cdot f(\tau' \cdot x) = \tau^{\mathsf{T}} \cdot \mu^{\mathsf{T}} \cdot \mu \cdot f(\tau \cdot x)$.

We recall (5); by Theorem 9 and $C' \cap supp(f(\tau \cdot x)) = \emptyset$, $\mu^{\mathsf{T}} \cdot \mu \cdot f(\tau \cdot x) = f(\tau \cdot x)$. The claim that the choice of fresh C does not matter, follows.

To see that f' indeed extends f we suppose that $x \in |\mathbf{X}|$ and that $f(x)$ is defined. Then $supp(\tau \cdot x) \cap S = \emptyset$ so $f(\tau \cdot x)$ is also defined. By assumption on f we have $\tau^{\mathsf{T}} \cdot f(\tau \cdot x) = f(\tau^{\mathsf{T}} \cdot \tau \cdot x) = f(x)$ where the last equality uses Theorem 9.

To see that $f' \in |\mathbf{X} \Rightarrow \mathbf{Y}|$ we put $S_{f'} = S$. Suppose that $\sigma|_S = id|_S$. We have seen that the choice of the fresh C does not matter so that we can assume that τ and τ^{T} commute with σ. We then have $\sigma \cdot f'(x) = \sigma \cdot \tau^{\mathsf{T}} \cdot f(\tau \cdot x) = \tau^{\mathsf{T}} \cdot \sigma \cdot f(\tau \cdot x) = f'(\sigma \cdot x)$.

Uniqueness of f' is now by Lemma 18.

Definition 20. *Let $\mathbf{X} \Rightarrow \mathbf{Y}$ have underlying set $|\mathbf{X} \Rightarrow \mathbf{Y}|$ with renaming action*

$$\text{if} \quad \sigma \cdot x = x \quad \text{then} \quad (\sigma \cdot f)x = \sigma \cdot f(x). \tag{6}$$

By Lemma 19 this uniquely determines the renaming action for all $x \in |\mathbf{X}|$.

Lemma 21. $\mathbf{X} \Rightarrow \mathbf{Y}$ *is a nominal renaming set.*

Proof. $f \in |\mathbf{X} \Rightarrow \mathbf{Y}|$ is supported by the set S_f from Definition 15, for suppose that $\sigma|_{S_f} = id|_{S_f}$. By Lemma 13 it suffices to show $\sigma \cdot f = f$. Now put $S = S_f \cup \{b \mid \sigma(b) \neq b\}$. This is a finite set so by Lemma 18 it suffices to show $(\sigma \cdot f)(x) = f(x)$ for all x such that $supp(x) \cap S = \emptyset$. Fix such an x. By Theorem 9 $\sigma \cdot x = x$ so by (6) we know $(\sigma \cdot f)(x) = \sigma \cdot f(x)$. $supp(x) \cap S_f = \emptyset$ by assumption so by (4) we know $\sigma \cdot f(x) = f(\sigma \cdot x)$. Then $f(\sigma \cdot x) = f(x)$ since $\sigma \cdot x = x$. The result follows.

Theorem 22. $\sigma \cdot f(x) = (\sigma \cdot f)(\sigma \cdot x)$, *for* $f \in |\mathbf{X} \Rightarrow \mathbf{Y}|$.

Proof. Write $supp(f) \cup supp(\sigma \cdot f) \cup \{a \mid \sigma(a) \neq a\} = \{a_1, \ldots, a_k\}$. Choose fresh $C = \{c_1, \ldots, c_k\}$, so $C \cap (supp(x) \cup S_f \cup S_{\sigma \cdot f}) = \emptyset$ and $\sigma|_C = id|_C$. Define

$$\tau = [a_1 \mapsto c_1, \ldots, a_k \mapsto c_k] \qquad \tau^{\mathsf{T}} = [c_1 \mapsto a_1, \ldots, c_k \mapsto a_k]$$
$$\pi = [a_1 \mapsto c_1, c_1 \mapsto a_1, \ldots, a_k \mapsto c_k, c_k \mapsto a_k] \qquad \sigma' = \pi \circ \sigma \circ \pi.$$

Then:	$(\tau^{\mathsf{T}} \circ \sigma' \circ \tau)|_{supp(x)} = \sigma|_{supp(x)}$	$C \cap supp(x) = \emptyset$	(7)
	$\tau^{\mathsf{T}} \circ \sigma' \circ \sigma = \sigma \circ \tau^{\mathsf{T}}$	$\sigma(c){=}c$ for all $c{\in}C$	(8)
	$(\tau^{\mathsf{T}} \circ \sigma')|_{S_{\sigma \cdot f}} = id|_{S_{\sigma \cdot f}}$	By construction	(9)

So:	$(\sigma \cdot f)(\sigma \cdot x) = (\sigma \cdot f)((\tau^{\mathsf{T}} \circ \sigma') \cdot \tau \cdot x)$	Th. 9 and (7)
	$= (\tau^{\mathsf{T}} \circ \sigma') \cdot (\sigma \cdot f)(\tau \cdot x)$	(9) and (4), Def. 15
	$= (\tau^{\mathsf{T}} \circ \sigma') \cdot \sigma \cdot f(\tau \cdot x)$	(9) and (6), Def. 20
	$= (\sigma \circ \tau^{\mathsf{T}}) \cdot f(\tau \cdot x)$	(8)
	$= \sigma \cdot f(\tau^{\mathsf{T}} \cdot \tau \cdot x)$	(4), Def. 15
	$= \sigma \cdot f(x)$	Theorem 9

Corollary 23. *If* $f{\in}|\mathbf{X}{\Rightarrow}\mathbf{Y}|$ *then the following are equivalent:*

– *For all* $x \in |\mathbf{X}|$ *and* $\sigma \in Fin$, *if* $\sigma|_S = id|_S$ *then* $\sigma{\cdot}f(x) = f(\sigma{\cdot}x)$.
– $supp(f) \subseteq S$.

Proof. If $\sigma|_{supp(f)} = id|_{supp(f)}$ then by Theorems 22 and 9, $\sigma \cdot f(x) = (\sigma \cdot f)(\sigma \cdot x) = f(\sigma \cdot x)$.

Now suppose $S \subseteq \mathbf{A}$ and for all $x \in |\mathbf{X}|$ if $\sigma|_S = id|_S$ then $\sigma{\cdot}f(x) = f(\sigma{\cdot}x)$. If we show that S supports f then by the 'unique least' property of support (Theorem 9) we are done. Suppose $\sigma|_S = id|_S$ and take any $x \in |\mathbf{X}|$ such that $\sigma|_{supp(x)} = id|_{supp(x)}$:

$$f(x) \overset{\text{Theorem 9}}{=} f(\sigma \cdot x) \overset{\text{Assumption}}{=} \sigma \cdot f(x) \overset{\text{Theorem 22}}{=} (\sigma \cdot f)(\sigma \cdot x) \overset{\text{Theorem 9}}{=} (\sigma \cdot f)(x).$$

By Lemma 18 $\sigma \cdot f = f$ as required.

Compare Corollary 24 with [13, Example 4.9, (24)]:

Corollary 24. *$supp(f(x)) \subseteq supp(f) \cup supp(x)$ always, for $f \in |\mathbf{X} \Rightarrow \mathbf{Y}|$.*

Proof. Using Theorems 22 and 9.

Lemma 25. *Arrows $F : \mathbf{X} \longrightarrow \mathbf{Y}$ are exactly $f \in |\mathbf{X} \Rightarrow \mathbf{Y}|$ such that $supp(f) = \emptyset$.*

Proof. $F : \mathbf{X} \longrightarrow \mathbf{Y}$ is a map in $|\mathbf{X}| \to |\mathbf{Y}|$. By definition $\sigma \cdot F(x) = F(\sigma \cdot x)$. By Corollary 23, $supp(F) = \emptyset$. Conversely if $f \in |\mathbf{X} \Rightarrow \mathbf{Y}|$ and $supp(f) = \emptyset$ then f is a map in $|\mathbf{X}| \to |\mathbf{Y}|$. Also, $\sigma \cdot f(x) = f(\sigma \cdot x)$ by Theorems 22 and 9.

Theorem 26. - $\Rightarrow$ - is *an exponential in* Ren.

Proof. We show that currying and uncurrying are arrows between $(\mathbf{X} \times \mathbf{Y}) \longrightarrow \mathbf{Z}$ and $\mathbf{X} \longrightarrow (\mathbf{Y} \Rightarrow \mathbf{Z})$. The $\beta\eta$ equations are then inherited.
Currying: Take $F \in (\mathbf{X} \times \mathbf{Y}) \longrightarrow \mathbf{Z}$. For $x \in \mathbf{X}$ we put $F_x = (\lambda y{\in}|\mathbf{Y}|.F(x, y))$. We show that $\lambda x{\in}|\mathbf{X}|.F_x \in \mathbf{X} \longrightarrow (\mathbf{Y} \Rightarrow \mathbf{Z})$:

If $x \in |X|$ we put $S_{F_x} = supp(x)$. If $\sigma|_{supp(x)} = id|_{supp(x)}$ then $\sigma \cdot F_x(y) = F(\sigma \cdot x, \sigma \cdot y) = F(x, \sigma \cdot y) = F_x(\sigma \cdot y)$, so $F_x \in |\mathbf{X} \Rightarrow \mathbf{Y}|$.

To see that $x \mapsto F_x$ is an arrow pick arbitrary σ. We need to show $\sigma \cdot F_x = F_{\sigma \cdot x}$. Appealing to Lemma 18 we choose $y \in \mathbf{Y}$ with $\sigma \cdot y = y$ and $supp(y)$ disjoint from $supp(x) = supp(F_x)$. We then have $(\sigma \cdot F_x)(y) = \sigma \cdot F(x, \sigma \cdot y) = \sigma \cdot F(x, y) = F(\sigma \cdot x, \sigma \cdot y) = F(\sigma \cdot x, y) = F_{\sigma \cdot x}(y)$.
Uncurrying: Take $G \in \mathbf{X} \longrightarrow (\mathbf{Y} \Rightarrow \mathbf{Z})$. We show that $\lambda(x, y){\in}|\mathbf{X} \times \mathbf{Y}|.G(x)(y) \in (\mathbf{X} \times \mathbf{Y}) \longrightarrow \mathbf{Z}$:

It suffices to show that $\sigma \cdot G(x)(y) = G(\sigma \cdot x)(\sigma \cdot y)$. Now $\sigma \cdot G(x)(y) = (\sigma \cdot G(x))(\sigma \cdot y)$ by Theorem 22 and $\sigma \cdot G(x) = G(\sigma \cdot x)$ by Definition 5.

Lemma 27 does for Ren what [13, Lemma 6.3] does for the category of nominal sets. We can develop similar inductive and recursive principles for syntax-with-binding as are exhibited using the Gabbay-Pitts И-quantifier in Theorem 6.5 in [13].

Lemma 27. *There is a bijection between $F : (\mathbf{A} \times \mathbf{X}) \longrightarrow \mathbf{Y}$ such that $a\#F(a, x)$ always, and $G \in (\mathbf{A} \Rightarrow \mathbf{X}) \longrightarrow \mathbf{Y}$.*

Proof. We define mappings as follows:

– F maps to $\lambda f \in |\mathbf{A} \Rightarrow \mathbf{X}|.Иa.\ F(a, fa)$ where (as is standard [13]) $Иa.\ F(a, fa)$ is equal to $F(a, fa)$ for any a such that $a\#f$.
– G maps to $\lambda(a, x) \in |\mathbf{A} \times \mathbf{X}|.G(\lambda y{\in}\mathbf{A}.[a{\mapsto}y] \cdot x)$.

We can prove this is well-defined, and the result follows by calculations.

4 The atoms-exponential $\mathbf{A} \Rightarrow \mathbf{X}$

Lemma 28. *If $f \in |\mathbf{A} \Rightarrow \mathbf{X}|$ then $f = \lambda y{\in}\mathbf{A}.[a{\mapsto}y] \cdot x$ for some $x \in |\mathbf{X}|$ and $a\#f$.*

Proof. By Definition 20 $f \in |\mathbf{A} \Rightarrow \mathbf{X}|$ when $f \in |\mathbf{A}| \to |\mathbf{X}|$ and there is a finite $S \subseteq \mathbf{A}$ such that $\sigma \in Fin$, $a \in \mathbf{A}$, and $\sigma|_S = id|_S$ imply $\sigma \cdot f(a) = f(\sigma(a))$. Choose $a \notin S$ and $y \in \mathbf{A}$. Then $f(y) = f([a{\mapsto}y] \cdot a) = [a{\mapsto}y] \cdot f(a)$, and we take $x = f(a)$.

Two functions on atoms are equal if they agree for one fresh atom; compare this for example with axiom (Ext^{τ}_{ν}) in the theory of contexts [17, Figure 4] and the extensionality principle for concretion of the Gabbay-Pitts model of atoms-abstraction in nominal sets [13, Proposition 5.5, equation (48)]:

Corollary 29. *Suppose $f, f' \in |\mathbf{A} \Rightarrow \mathbf{X}|$ and suppose $c\#f$ and $c\#f'$. Then $f(c) = f'(c)$ if and only if $f = f'$.*

Proof. The right-to-left implication is easy. Assume $f(c) = f'(c)$. By Lemma 28

- there exist $x \in |\mathbf{X}|$ and $a \in \mathbf{A}$ such that $a\#f$ and $f = \lambda y{\in}\mathbf{A}.[a{\mapsto}y] \cdot x$, and
- there exist $x' \in |\mathbf{X}|$ and $a' \in \mathbf{A}$ such that $a'\#f'$ and $f' = \lambda y{\in}\mathbf{A}.[a'{\mapsto}y] \cdot x'$.

We assumed that $f(c) = f'(c)$ so $[a{\mapsto}c] \cdot x = [a'{\mapsto}c] \cdot x'$. By Lemma 14 $a'\#x$ and $x' = [a{\mapsto}a'] \cdot x$. Choose any $y \in \mathbf{A}$. We reason as follows:

$$\begin{array}{ll} f'(y) = [a'{\mapsto}y] \cdot x' & f' = \lambda y{\in}\mathbf{A}.[a'{\mapsto}y] \cdot x' \\ \quad = [a'{\mapsto}y] \cdot [a{\mapsto}a'] \cdot x. & x' = [a{\mapsto}a'] \cdot x \\ \quad = [a{\mapsto}y] \cdot x & \text{Theorem 9, } a'\#x \end{array}$$

Lemma 30. *Suppose $f \in |\mathbf{A} \Rightarrow \mathbf{Y}|$. Then $a\#f$ if and only if $a\#f(b)$, for any $b\#f$ (by our permutative convention, $b \neq a$).*

Equivalently, $supp(f) = supp(f(b)) \setminus \{b\}$ for any $b\#f$.

Proof. We prove two implications. Choose any $b\#f$.

If $a\#f$ then by Corollary 24 $a\#fb$ and we are done. Suppose that $a\#f(b)$. We have assumed $b\#f$ so by Corollary 12 it suffices to prove $[a{\mapsto}b] \cdot f = f$. Choose a fresh c (so $c\#f$ and $c\#[a{\mapsto}b] \cdot f$). By Corollary 29 it suffices to check that $f(c) = ([a{\mapsto}b] \cdot f)(c)$. Note that by Corollary 24 $a\#f(c)$. We reason as follows:

$$\begin{array}{ll} fc = [a{\mapsto}b] \cdot f(c) & \text{Theorem 9 and Corollary 24} \\ \quad = ([a{\mapsto}b] \cdot f)([a{\mapsto}b] \cdot c) & \text{Theorem 22} \\ \quad = ([a{\mapsto}b] \cdot f)c & [a{\mapsto}b] \cdot c = c \end{array}$$

Lemma 31. *$\lambda y{\in}\mathbf{A}.[a{\mapsto}y] \cdot x$ is supported by $supp(x)$; thus $\lambda y{\in}\mathbf{A}.[a{\mapsto}y] \cdot x \in |\mathbf{A} \Rightarrow \mathbf{X}|$.*

Proof. The corollary is immediate given the first part, by Definition 15. We now prove that $supp(x)$ supports $\lambda y{\in}\mathbf{A}.[a{\mapsto}y] \cdot x$. By Corollary 23 it suffices to show $\sigma|_{supp(x)} = id|_{supp(x)}$ implies $\sigma \cdot \lambda y{\in}\mathbf{A}.[a{\mapsto}y] \cdot x = \lambda y{\in}\mathbf{A}.[a{\mapsto}y] \cdot x$.

So suppose $\sigma|_{supp(x)} = id|_{supp(x)}$. By Corollary 29 it suffices to check

$$(\sigma \cdot \lambda y{\in}\mathbf{A}.[a{\mapsto}y] \cdot x)c = (\lambda y{\in}\mathbf{A}.[a{\mapsto}y] \cdot x)c$$

for fresh c. Choose c such that $c\#\sigma \cdot \lambda y{\in}\mathbf{A}.[a{\mapsto}y] \cdot x$, $c\#\lambda y{\in}\mathbf{A}.[a{\mapsto}y] \cdot x$, $\sigma(c) = c$ and $c\#x$. By Theorem 22 since $\sigma(c) = c$ we have that

$$\sigma \cdot ((\lambda y{\in}\mathbf{A}.[a{\mapsto}y] \cdot x)c) = (\sigma \cdot \lambda y{\in}\mathbf{A}.[a{\mapsto}y] \cdot x)c.$$

So it suffices to check that $\sigma \cdot [a{\mapsto}c] \cdot x = [a{\mapsto}c] \cdot x$. We assumed $\sigma|_{supp(x)} = id|_{supp(x)}$ and $c\#x$. It follows that $(\sigma \circ [a{\mapsto}c])|_{supp(x)} = [a{\mapsto}c]|_{supp(x)}$. By Theorem 9 the result follows.

Corollary 32. $supp(\lambda y{\in}\mathbf{A}.[a{\mapsto}y]\cdot x) = supp(x)\setminus\{a\}$.

Proof. Choose some fresh b (so $b\#\lambda y{\in}\mathbf{A}.[a{\mapsto}y]\cdot x$ and $b\#x$). By Lemma 31 $\lambda y{\in}\mathbf{A}.[a{\mapsto}y]\cdot x \in |\mathbf{A}\Rightarrow\mathbf{X}|$. Therefore by Lemma 30 we know that

$$supp(\lambda y{\in}\mathbf{A}.[a{\mapsto}y]\cdot x) = supp([a{\mapsto}b]\cdot x)\setminus\{b\}.$$

Since $b\#x$ by part 2 of Lemma 11 $supp([a{\mapsto}b]\cdot x) = (supp(x)\setminus\{a\})\cup\{b\}$.

Lemma 33. *Suppose* $x, x' \in |\mathbf{X}|$, *and* $a, a' \in \mathbf{A}$. *Then* $\lambda y{\in}\mathbf{A}.[a{\mapsto}y]\cdot x = \lambda y{\in}\mathbf{A}.[a'{\mapsto}y]\cdot x'$ *if and only if* $a'\#x$ *and* $x' = [a{\mapsto}a']\cdot x$.

Proof. Using Lemma 14 and Corollary 29.

The reader may recognise these results from the theory of the Gabbay-Pitts model of atoms-abstraction in nominal sets [13, Definition 5.4]. A nominal renaming set is also a nominal permutation set (by 'forgetting' the action for non-bijective σ); using Corollary 12 it can be proved [11, Theorem 4.8] that the notions of support coincide. In the spirit of [13, (35)] we can write[5] $[a]x = \{(a,x)\}\cup\{(c,[a{\mapsto}c]\cdot x) \mid c\#x\}$ and $|[\mathbf{A}]\mathbf{X}| = \{[a]x \mid a\in\mathbf{A},\ x\in|\mathbf{X}|\}$. The proof of Theorem 34 is now routine [11, Theorem 5.7]:

Theorem 34. $|\mathbf{A}\Rightarrow\mathbf{X}|$ *is in bijective correspondence with* $|[\mathbf{A}]\mathbf{X}|$. *Inverse mappings are given by:*

– α *maps* $z\in|[\mathbf{A}]\mathbf{X}|$ *to* $\lambda y{\in}\mathbf{A}.[a{\mapsto}y]\cdot x$, *for* $(a,x){\in}z$.
– β *maps* $f\in|\mathbf{A}\Rightarrow\mathbf{X}|$ *to* $[a](fa)$, *for* $a\#f$.

5 Presheaf and topos structure of Ren

For convenience, take the finite sets in $\mathbb{I}$ and $\mathbb{F}$ from the Introduction to be finite $S\subseteq\mathbb{A}$.

We could now go on and exhibit the subobject classifier in Ren, thus establishing that Ren is a topos, and hence is a model of higher-order logic. This is not hard to do: a subobject classifier is the renaming set Ω with underlying set $|\Omega|$ those $U\subseteq Fin$ such that:

– If $\sigma\in U$ then $\mu\circ\sigma\in U$ for all $\mu\in Fin$.
– There exists finite $S\subseteq\mathbf{A}$ such that $\mu\in U$ and $\sigma|_S = id|_S$ imply $\mu\circ\sigma\in U$.

The renaming action is given by $\sigma\cdot U = \{\mu \mid \mu\circ\sigma\in U\}$. The proof that this makes Ren into a topos is by standard calculations. We also remark that Ren is a Grothendieck topos for a topology on the opposite of the category of finite sets and functions and thus a full subcategory of the functor category $\mathsf{Set}^{\mathbb{F}}$. The required topology has for basic covers of an object X nonempty families of monos $f_i : X \longrightarrow X_i$. The interested reader is referred to [24] where a proof of a similar result is worked out.

We now characterise Ren as a category of presheaves.

[5] The use of $[a{\mapsto}c]$ instead of the swapping $(a\ c)$ (swapping defined in [13, (3)]) is immaterial because $c\#x$ and so the actions coincide by Theorem 9.

Definition 35. *Let* PBM *be the category of presheaves in* $\mathsf{Set}^{\mathbb{F}}$ *that preserve pullbacks of pairs of monos, and natural transformations between them.*

If $f \in X \to Y$ is a function let its *image* $img(f)$ be $\{f(x) \mid x \in X\} \subseteq Y$. Note that monos in $\mathbb{F}$ and Ren are injections, and that a pullback of a pair of monos is given by set intersection with the natural inclusion maps. Write '$\iota : S \subseteq S'$' for "$S \subseteq S'$ and we write ι for the subset inclusion arrow in $\mathbb{F}$".

Definition 36. *Fix* $F \in$ PBM. *Let* (S, x) *range over pairs where* $S \subseteq \mathbf{A}$ *is finite and* $x \in F(S)$. *Write* $\sim$ *for the least equivalence relation such that* $\iota : S \subseteq S'$ *implies* $(S, x) \sim (S', F(\iota)(x))$.

Lemma 37. *If* $\iota_1 : S_1 \subseteq S'$ *and* $\iota_2 : S_2 \subseteq S'$ *and* $(S_1, x_1) \sim (S', x') \sim (S_2, x_2)$ *then there exists some* $x \in F(S_1 \cap S_2)$ *such that* $(S_1 \cap S_2, x) \sim (S', x')$. *Thus, each* $\sim$*-equivalence class has a unique least representative* (S, x), *'least' in the sense that if* $(S, x) \sim (S', x')$ *then* $\iota : S \subseteq S'$ *and* $x' = F(\iota)(x)$.

Theorem 38. Ren *is equivalent to* PBM.

Proof. $\mathbf{X}$ maps to $F_{\mathbf{X}}$ mapping $S \in \mathbb{F}$ to $\{x \in |\mathbf{X}| \mid supp(x) \subseteq S\}$ and mapping $\tau : S \to S'$ to the renaming action of τ extended to a total function which is the identity off S. This is a presheaf by part 1 of Lemma 11 and we can prove that it preserves pullbacks of monos using Lemma 8. F maps to the set of unique least representative elements of $\sim$-equivalence classes, as constructed in Lemma 37 with action given by $\tau \cdot (S, x)$ is the representative of the $\sim$-equivalence class of $(\tau \cdot S, F(\tau|_S)(x))$. The result follows by routine calculations.

Preserving pullbacks of monos, and the corollary that we have 'least representatives', makes possible the sets-based presentation of nominal sets and nominal renaming sets, contrasting with the purely presheaf-based presentations of [7,16].

6 Tripos structure on Ren

As argued in [16] the topos logic of $\mathcal{F}$, thus also that of Ren, is unsuited to reasoning about syntax with binding. For example equality of atoms is not decidable in that logic. In fact, in the topos logic of Ren and $\mathcal{F}$, the proposition $\forall x, y{\in}\mathbf{A}.\neg\neg(x{=}y)$ holds.

It was proposed in [16] to use the tripos obtained by pulling back the logic from the Schanuel topos instead. In particular, the tripos so obtained is needed to justify the theory of contexts [17] (well — *almost*; the restriction to pullback-preserving functors had not been made explicit in that paper.)

In keeping with our goal of making these constructions concrete and usable for direct calculations, we make this construction explicit at the level of Ren.

Definition 39. *For a nominal renaming set* $\mathbf{X}$ *let* $Pred(\mathbf{X})$ *be the set of those subsets* U *of* $|\mathbf{X}|$ *that are preserved by bijective renamings, i.e., for which* $x \in U$ *implies* $\sigma \cdot x \in U$ *provided that* $\sigma \in Fin$ *is bijective, i.e., a permutation.*

$Pred(\mathbf{X})$ *is ordered by subset inclusion and* $Pred(-)$ *extends to a functor* $\mathsf{Ren}^{op} \to$ *Poset where* $Pred(f)(U) = \{x \mid f(x) \in U\}$ *when* $f : \mathbf{X} \to \mathbf{Y}$ *and* $U \in Pred(\mathbf{Y})$.

We remark that in the topos logic of Ren a predicate is a subset preserved by all renamings, not just the bijective ones.

Theorem 40 is a consequence of the folklore result stated in [16] which asserts that triposes can be pulled back along finite-limit-preserving functors with a right adjoint ('geometric morphisms'). Rather than detailing this argument we illustrate the tripos structure by concrete constructions below.

Theorem 40. *The pair* $(\mathsf{Ren}, Pred)$ *forms a tripos.*

This means there is enough structure to interpret higher-order logic. In particular, predicates are closed under boolean operations, universal quantification over renaming sets, and there is a renaming set $\mathbf{O}$ of propositions such that morphisms $\mathbf{X} \to \mathbf{O}$ are in 1-1 correspondence with elements of $Pred(\mathbf{X})$. We explicitly construct these ingredients in order to demonstrate that the logic for Ren is quite close to classical set-theoretic reasoning in contrast to the functorial reasoning that was used in [17].

Proposition 41. *The object of propositions* $\mathbf{O}$ *in the tripos* $(\mathsf{Ren}, Pred)$ *has underlying set those* $U \subseteq Fin$ *such that:*

– *If* $\sigma \in U$ *and* $\pi \in Fin$ *is a permutation then* $\pi \circ \sigma \in U$

– *There exists finite* $S \subseteq \mathbf{X}$ *such that if* $\sigma|_S = \sigma'|_S$ *then* $\sigma \in U$ *iff* $\sigma' \in U$.

The renaming action in $\mathbf{O}$ *is given by* $\sigma \cdot U = \{\mu \mid \mu \circ \sigma \in U\}$.

Proof (Sketch). If $P \in Pred(\mathbf{X})$ then a morphism $\chi_P : \mathbf{X} \to \mathbf{O}$ is defined by $\chi_P(a) = \{\sigma \mid \sigma \cdot a \in P\}$. Conversely, if $m : \mathbf{X} \to \mathbf{O}$ is a morphism then we associate the predicate $P_m = \{a \mid m(a) = \top\}$ where $\top \in \mathbf{O}$ is given by $\top = Fin$.

Proposition 42. *The logic of* $(\mathsf{Ren}, Pred)$ *is classical.*

Proof. It suffices to show that if $P \in Pred(\mathbf{X})$ is a predicate then its set-theoretic complement $\{x \mid x \notin P\}$ is again a predicate. But if $\pi \in Fin$ is a permutation and $x \notin P$ then $\pi \cdot x \notin P$ for otherwise $x = \pi^{-1} \cdot \pi \cdot x \in P$.

Proposition 43 (universal quantification). *Let* $P \in Pred(\mathbf{X} \times \mathbf{Y})$ *(possibly given by a morphism* $\mathbf{X} \times \mathbf{Y} \to \mathbf{O}$*). Its universal quantification* $\forall P \in Pred(\mathbf{X})$ *is given by* $\forall P = \{x \mid \forall y \in \mathbf{Y}.(x, y) \in P\}$.

Proof. Suppose that $Q \in Pred(\mathbf{X})$. We have to show that $\forall P$ is a predicate and that the following are equivalent:

– for all $x \in \mathbf{X}$ and $y \in \mathbf{Y}$ if $x \in Q$ then $(x, y) \in P$

– for all $x \in \mathbf{X}$ if $x \in Q$ then $x \in \forall P$

The equivalence being obvious from the definition it only remains to show that $\forall P$ is indeed a predicate. So assume that $\pi \in Fin$ is a permutation and consider that $x \in \forall P$. If $y \in \mathbf{Y}$ then $(x, \pi^{-1} \cdot y) \in P$ since $x \in \forall P$. So, $(\pi \cdot x, y) \in P$ since P is a renaming set. Hence, since y was arbitrary, $\pi \cdot x \in \forall P$ as required.

Logical operations in $(\mathsf{Ren}, Pred)$ are given by their standard meanings in classical set theory. Propositions are elements of $\mathbf{O}$ rather than of $\{0, 1\}$. Note that the axiom

of unique choice (which identifies functional relations with morphisms) is not valid in $(\mathsf{Ren}, \mathit{Pred})$. Indeed, a functional relation, i.e., a predicate P on $\mathbf{X} \times \mathbf{Y}$ such that for all $x \in |\mathbf{X}|$ there is a unique $y \in |\mathbf{Y}|$ such that $(x, y) \in P$, does not in general give rise to a morphism from $\mathbf{X}$ to $\mathbf{Y}$. Indeed, in [16] it was shown that the Axiom of Unique Choice is an inconsistent extension of the theory of contexts. There is a unique function f such that $(x, f(x)) \in P$ for all $x \in |\mathbf{X}|$ but in general it will be invariant under permutations, not all renamings.

7 Conclusions

Nominal renaming sets are a natural evolution of nominal sets. The connection between Ren and the category of nominal sets [13] is evident. However, the details are not entirely straightforward; proofs have to be changed and sometimes they take on a very different character. In particular the definition of the action for function spaces in the permutative case $(\sigma \cdot f)(x) = \sigma^{-1} \cdot f(\sigma \cdot x)$ does not carry over, because σ^{-1} need not exist. It is replaced by an *a priori* partial definition (Definition 20) which is totalised with a unique extension theorem (Lemma 19).

Nominal renaming sets are related to $\mathsf{Set}^{\mathbb{F}}$ as featured in [7] and [16]; they correspond to the pullback-preserving presheaves (Theorem 38). Ren has appeared before; the first author discussed the idea in his thesis [9, Section 11.1] and studied a generalisation of nominal renaming sets as an algebraic theory over nominal sets in [12]. Meanwhile, a category equivalent to Ren was independently presented, also as an algebraic theory over nominal sets, in [8, Definition 2.4]. The constructions presented in this paper, notably the exponential and the tripos structure, are new and do not follow directly from these earlier results. It is thanks to these explicit constructions that Ren can serve as a sets-based semantic underpinning for weak HOAS, i.e. that we can do calculations entirely within Ren, without switching to a functor category. To our knowledge, we are the first to suggest applying renaming sets as a semantic basis for weak HOAS in the sense of Despeyroux [5] and in particular the theory of contexts [17].

Future work.

It will be interesting to give details of a proof of validity of the theory of contexts using nominal renaming sets. In this conference paper we have not done that, but such a proof could be based on the one in [17], and should be simplified thanks to the use of set-theoretic language (most of the time). Note that no notion of forcing is required.

Nominal sets have been generalised to 'nominal domains' [22] thus giving access to fresh names in denotational semantics. It would be interesting to generalise renaming sets in this way as well, giving access to fresh names with substitution within a functional metalanguage. Here, our explicitation of the exponential (Section 3) may be particularly useful.

Nominal renaming sets belong to a family of structures with a finitely supported substitution action. Probably they are the simplest, but others may also be interesting. For example, define a *nominal substitution set* to be a pair of a nominal renaming set $\mathbf{X}$ and a function $\mathit{sub} : ((\mathbf{A} \Rightarrow \mathbf{X}) \times \mathbf{X}) \Rightarrow \mathbf{X}$ such that

– If $a \# z$ then $z[a \mapsto x] = z$.
– If $a \# y$ then $z[a \mapsto x][b \mapsto y] = z[b \mapsto y][a \mapsto x[b \mapsto y]]$.

Here we write $z[a{\mapsto}x]$ for $sub(\lambda y{\in}\mathbf{A}.[a{\mapsto}y]\cdot z,\ x)$. A category of nominal substitution sets has arrows maps $F : \mathbf{X} \longrightarrow \mathbf{Y}$ such that $F(z[a{\mapsto}x]) = (F(z))[a{\mapsto}F(x)]$.

(This can also be phrased directly using 'normal' sets.) We conjecture that this definition or a refinement of it will be useful to give semantics to typings like $(tm \to tm) \to tm$ used for binders, for example in Twelf [19], for which hitherto only purely presheaf semantics are available [16].

References

1. Nick Benton and Benjamin Leperchey. Relational reasoning in a nominal semantics for storage. In Pawel Urzyczyn, editor, *TLCA*, volume 3461 of *Lecture Notes in Computer Science*, pages 86–101. Springer, 2005.
2. Norbert Brunner. 75 years of independence proofs by Fraenkel-Mostowski permutation models. *Mathematica Japonica*, 43:177–199, 1996.
3. Anna Bucalo, Furio Honsell, Marino Miculan, Ivan Scagnetto, and Martin Hofmann. Consistency of the theory of contexts. *Journal of Functional Programming*, 16(3):327–395, 2006.
4. Joelle Despeyroux. A higher-order specification of the π–calculus. In *IFIP TCS*, pages 425–439, 2000.
5. Joëlle Despeyroux, Amy P. Felty, and André Hirschowitz. Higher-order abstract syntax in COQ. In *TLCA'95*, volume 3461 of *Lecture Notes in Computer Science*, pages 124–138, 1995.
6. Joëlle Despeyroux and André Hirschowitz. Higher-order abstract syntax with induction in COQ. In *LPAR'94*, volume 822 of *Lecture Notes in Computer Science*, pages 159–173. Springer, 1994.
7. Marcelo P. Fiore, Gordon D. Plotkin, and Daniele Turi. Abstract syntax and variable binding. In *LICS'99*, pages 193–202. IEEE, 1999.
8. Marcelo P. Fiore and Sam Staton. A congruence rule format for name-passing process calculi from mathematical structural operational semantics. In *LICS'06*, pages 49–58. IEEE, 2006.
9. Murdoch J. Gabbay. A Theory of Inductive Definitions with alpha-Equivalence. PhD thesis, Cambridge, UK, 2000.
10. Murdoch J. Gabbay. A General Mathematics of Names. *Information and Computation*, 205:982–1011, July 2007.
11. Murdoch J. Gabbay. Nominal renaming sets. Technical Report HW-MACS-TR-0058, Heriot-Watt University, 2007.
12. Murdoch J. Gabbay and Aad Mathijssen. Capture-avoiding Substitution as a Nominal Algebra. In *ICTAC*, volume 4281 of *Lecture Notes in Computer Science*, pages 198–212, 2006.
13. Murdoch J. Gabbay and A. M. Pitts. A New Approach to Abstract Syntax with Variable Binding (journal version). *Formal Aspects of Computing*, 13(3–5):341–363, 2001.
14. Murdoch J. Gabbay and Andrew M. Pitts. A new approach to abstract syntax involving binders. In *14th Annual Symposium on Logic in Computer Science*, pages 214–224. IEEE Computer Society Press, 1999.
15. Daniel Hirschkoff. A full formalization of pi-calculus theory in the Calculus of Constructions. In E. Gunter and A. Felty, editors, *Proceedings of TPHOLs'97*, pages 153–169, Murray Hill, 1997.
16. Martin Hofmann. Semantical analysis of higher-order abstract syntax. In *14th Annual Symposium on Logic in Computer Science*, pages 204–213. IEEE, 1999.
17. Furio Honsell, Marino Miculan, and Ivan Scagnetto. An axiomatic approach to metareasoning on nominal algebras in HOAS. In *ICALP'01*, volume 2076, pages 963–978. Lecture Notes in Computer Science, 2001.
18. James McKinna and Robert Pollack. Some lambda calculus and type theory formalized. *Journal of Automated Reasoning*, 23(3-4):373–409, 1999.
19. Frank Pfenning and Carsten Schürmann. System description: Twelf - a meta-logical framework for deductive systems. In *CADE'16*, volume 1632 of *Lecture Notes in Computer Science*, pages 202–206. Springer, 1999.
20. M. R. Shinwell. *The Fresh Approach: Functional Programming with Names and Binders*. PhD thesis, Computer Laboratory, University of Cambridge, December 2004.
21. M. R. Shinwell, A. M. Pitts, and Murdoch J. Gabbay. FreshML: Programming with binders made simple. In *ICFP'2003*, volume 38(9) of *SIGPLAN Not.*, pages 263–274. ACM Press, 2003.
22. Mark R. Shinwell and Andrew M. Pitts. On a monadic semantics for freshness. *Theoretical Computer Science*, 342(1):28–55, 2005.
23. M.R. Shinwell and A.M. Pitts. Fresh objective Caml user manual. Technical Report UCAM-CL-TR-621, University of Cambridge, 2005.
24. Sam Staton. *Name-passing process calculi: operational models and structural operational semantics*. PhD thesis, University of Cambridge, 2007.
25. Christian Urban and Christine Tasson. Nominal techniques in Isabelle/HOL. In *CADE'05*, volume 3632 of *Lecture Notes in Artificial Intelligence*, pages 38–53, 2005.

Acknowledgements. Supported by grant RYC-2006-002131 at the Polytechnic University of Madrid.