Generalised Name Abstraction for Nominal Sets

Ranald Clouston*

Logic and Computation Group, Research School of Computer Science,
The Australian National University, Canberra, ACT 0200, Australia
ranald.clouston@anu.edu.au

Abstract. The Gabbay-Pitts nominal sets model provides a framework for reasoning with names in abstract syntax. It has appealing semantics for name binding, via a functor mapping each nominal set to the 'atom-abstractions' of its elements. We wish to generalise this construction for applications where sets, lists, or other patterns of names are bound simultaneously. The atom-abstraction functor has left and right adjoint functors that can themselves be generalised, and their generalisations remain adjoints, but the atom-abstraction functor in the middle comes apart to leave us with two notions of generalised abstraction for nominal sets. We give new descriptions of both notions of abstraction that are simpler than those previously published. We discuss applications of the two notions, and give conditions for when they coincide.

Keywords: name binding, nominal sets, adjoint functors.

1 Introduction

Programming languages and formal calculi frequently feature syntactic constructs called *names*, or *object-level variables*. When combined with *binding* constructs, rendering names anonymous within their scope, technical and conceptual issues arise that have attracted much research in recent years. Some such constructs bind not just a single name at a time, but a set, list, or *pattern* of names; as yet there is no convincing general theory for such binders [1].

One of the most prominent approaches to names and binding is the *nominal sets* model [10,16]. Nominal sets provide a mathematical model in which names exist as first-class citizens, manipulated by permutations. This model supports reasoning techniques that closely match pen-and-paper practice, in which we often explicitly manipulate bound names. Nominal sets have inspired a literature too extensive to summarise here, with nominal variants offered of everything from equational logic [4], to interactive theorem proving [24], to game semantics [23].

Binding in nominal sets is elegantly captured by a construction called *atom-abstraction*; for each nominal set X we define a new nominal set, written $[\mathbb{A}]X$, which can be thought of in two ways: as a set of pairs of names and X-elements

* The author gratefully acknowledges his discussions with Andrew Pitts, Michael Norrish, and Barry Jay, and the comments of the anonymous reviewers.

F. Pfenning (Ed.): FOSSACS 2013, LNCS 7794, pp. 434–449, 2013.

quotiented by legal renamings, and as a function mapping free names to X-elements. For example, $[\mathbb{A}]\mathbb{A}$ (names abstracted over names) contains pairs such as (a, a), equivalent to (b, b) but not equivalent to (b, a), or alternatively contains functions such as $\lambda x.x$ and $\lambda x.a$.

It is worth dwelling on this dual description of $[\mathbb{A}]X$ as (quotiented) product and (partial) function space. We can use the language of category theory to capture this intuition more firmly. The map $X \mapsto [\mathbb{A}]X$ extends to an endofunctor $[\mathbb{A}]$- on the category of nominal sets with both left and right adjoints:

$$\text{-} \otimes \mathbb{A} \;\dashv\; [\mathbb{A}]\text{-} \;\dashv\; \text{-}_{[\mathbb{A}]} \quad . \tag{1}$$

Attention was first drawn to this chain of adjunctions by [15]. It immediately tells us that atom-abstraction has nice properties, preserving all limits and colimits. More intriguingly, the leftmost functor comes via a sort of product, and the rightmost functor via a sort of function space. This can be compared to the cartesian closure adjunction $\text{-} \times X \dashv X \to \text{-}$, which gives the standard relationship between products and function spaces. Hence $[\mathbb{A}]X$ is somehow amphibious, both a product and a function space. As [6] puts it, atom-abstractions are "constructed like a pair... but destructed like a partial function".

Atom-abstraction nominal sets as a semantics for binding one name at a time are now well established; what of applications involving more complex binding? Such applications, such as the functional programming operator *letrec* that binds lists of names simultaneously, are known to be imperfectly modelled by one-at-a-time binding - see [1,21,24] for varied discussions on this point. This paper explores and critiques the fundamental mathematics that could provide a nominal sets semantics for such general binding. Concretely, we will try to generalise the atoms-abstraction construction $[\mathbb{A}]Y$ by replacing the set of names $\mathbb{A}$ with other nominal sets X, such as the nominal set of lists of names.

The adjunctions of (1) offer one approach to this generalisation, as the functors $\text{-}\otimes\mathbb{A}$ and $\text{-}_{[\mathbb{A}]}$ easily generalise to functors $\text{-}\otimes X$ and $\text{-}_{[X]}$. These generalisations remain left and right adjoints respectively, but an interesting thing happens: the right adjoint of $\text{-} \otimes X$ and left adjoint of $\text{-}_{[X]}$ do not in general coincide. Thus we have two notions of generalised abstraction, one of which is constructed like a pair, and the other of which is destructed like a partial function. Summarising our situation, we have

$$\text{-} \otimes X \;\dashv\; X \mathrel{-\!\!*} \text{-} \quad \text{and} \quad [X]\text{-} \;\dashv\; \text{-}_{[X]} \tag{2}$$

where $X \mathrel{-\!\!*} \text{-}$ and $[X]$- coincide in the case that $X = \mathbb{A}$, but not in general.

The left hand adjunction of (2) is known to exist for general category theoretic reasons, because $\otimes$, called here the *separated product*, is related to the 'Day convolutions' of [5]. We call its closure the *separating function space*, and use the 'magic wand' notation from the logic of Bunched Implications (BI) [18]. However this category theoretic view does not yield an accessible nominal set theoretic construction, and such concrete constructions have been key to the succesful applications of nominal techniques. The closest we have to such a construction is [19, Sec. 3.3], which involves a tricky quotient on partial functions

and selection of canonical members from each equivalence class. In Sec. 3 we will give a considerably simpler and more intuitively appealing construction.

Conversely, the construction $[X]$- was established in [9, Sec. 6] and implemented in the language FreshML [22]. It has been given the name *generalised abstraction* and notation matching that name; we will harmonise with the literature on this, at the risk of implying it is the only generalisation of atom-abstraction worth considering. To our knowledge, the right hand adjunction of (2) was first explicitly observed in the unpublished [17], although it is a corollary of the earlier [19, Prop. 3.3.32]; in Sec. 4 we will sketch this adjunction, along with a novel treatment of generalised abstractions as equalisers.

Sec. 5 will give necessary and sufficient conditions under which these notions of abstraction coincide, as they do for $\mathbb{A}$. The 'sufficient' direction is due to [19, Sec. 10.3]; here we restate that proof in terms of our simple notion of separating functions, and give the 'necessary' direction also. Finally, Sec. 6 will look at applications and limitations of these mathematical developments.

2 Nominal Sets

This section gives us a brief overview of nominal sets; [16] or [3, Cha. 2] provide more leisurely introductions to the area.

Definition 2.1. *Fix a countably infinite set $\mathbb{A}$ of* atoms. *The set* Perm *of (finite)* permutations *consists of all bijections $\pi : \mathbb{A} \to \mathbb{A}$ whose non-trivial domain*

$$supp(\pi) \triangleq \{a \mid \pi(a) \neq a\} \tag{3}$$

is finite.

Perm *is a group, with multiplication as permutation composition, $\pi'\pi(a) = \pi'(\pi(a))$, and identity as the permutation ι leaving all atoms unchanged.*

Example 2.2. The *transpositions* $(a\ b)$ map $a \mapsto b$, $b \mapsto a$ and leave all other atoms unchanged. Now let

$$\mathbb{A}^{(n)} \triangleq \{(a_1, \ldots, a_n) \in \mathbb{A}^n \mid a_i \neq a_j \text{ for } 1 \leq i < j \leq n\} \ . \tag{4}$$

All the tuples of atoms we use in this paper will be so disjoint. Take $\vec{a} = (a_1, \ldots, a_n), \vec{a}' = (a'_1, \ldots, a'_n) \in \mathbb{A}^{(n)}$ with mutually disjoint underlying sets. Then their *generalised transposition* is

$$(\vec{a}\ \vec{a}') \triangleq (a_1\ a'_1) \cdots (a_n\ a'_n) \ .$$

Definition 2.3. *A* Perm-set *is a set X equipped with a function, or* Perm-action, *$(\pi, x) \mapsto \pi \cdot x$ from $\mathsf{Perm} \times X$ to X such that $\iota \cdot x = x$ and $\pi \cdot (\pi' \cdot x) = \pi\pi' \cdot x$.*

Given such a Perm*-set X we say that a set of atoms $\overline{a} \subseteq \mathbb{A}$* supports *$x \in X$ if for all $\pi \in \mathsf{Perm}$, $supp(\pi) \cap \overline{a} = \emptyset$ implies that $\pi \cdot x = x$.*

Definition 2.4. *A* nominal set *is a* Perm*-set X with the* finite support property*: for each $x \in X$ there exists some finite $\overline{a} \subseteq \mathbb{A}$ supporting x.*

If an element x is finitely supported then there is a unique least such support set [10, Prop. 3.4], which we write $supp(x)$ and call the support of x*.*

Given nominal set elements $x \in X, y \in Y$, if $supp(x) \cap supp(y) = \emptyset$ then we write $x \# y$, and say that x is fresh *for y, or equivalently that y is fresh for x.*

Remark 2.5. The atoms $\mathbb{A}$ can be seen as a set of *names*, and the support of a nominal set element as its set of *free names*. The finite support condition reflects that for most notions of syntax, terms may have only finitely many free names. It is a useful condition because it allows us to uniquely define *the* support of an element, and hence always find names that are fresh for, or *not free in*, that element.

Example 2.6. (i) Any set is a nominal set under the trivial Perm-action $\pi \cdot x = x$; then $supp(x) = \emptyset$.

(ii) $\mathbb{A}$ is a nominal set given $\pi \cdot a = \pi(a)$; $supp(a) = \{a\}$.

(iii) Perm is a nominal set given the conjugation action $\pi \cdot \pi' = \pi\pi'\pi^{-1}$; support is as (3).

(iv) $\mathcal{P}_{fin}(\mathbb{A})$, the set of finite sets of atoms, is nominal given the element-wise Perm-action; $supp(\overline{a}) = \overline{a}$.

(v) Given nominal sets X, Y, the usual product $X \times Y$ is nominal given the element-wise Perm-action; $supp((x, y)) = supp(x) \cup supp(y)$. We write the n-fold product of X as X^n.

(vi) Define a subset of $X \times Y$ by

$$X \otimes Y \triangleq \{(x, y) \in X \times Y \mid x \# y\} \ .$$

This is nominal with the same element-wise action and supports as $X \times Y$, and is called the *separated product* of X and Y. The n-fold separated product of X is written $X^{(n)}$, as with (4).

(vii) The usual disjoint union $X + Y$, with typical members $(x, 1)$ or $(y, 2)$, is nominal given the Perm-actions and supports inherited from X, Y. Where there is no confusion we will omit the indices.

Definition 2.7. *We can define a* Perm*-action on the functions $f : X \to Y$ by applying the evident action to their graphs, i.e.*

$$(\pi \cdot f)(x) \triangleq \pi \cdot (f(\pi^{-1} \cdot x)) \ . \tag{5}$$

Functions between nominal sets are not necessarily finitely supported under this action; we call the functions that are so the finitely supported functions*.*

A function that has empty support under this action is called an equivariant function*; this property has the equivalent formulation*

$$\pi \cdot (f(x)) = f(\pi \cdot x) \ .$$

The category of nominal sets*, written $\mathcal{N}om$, has as objects, nominal sets, and as arrows, equivariant functions between them.*

Example 2.8. (i) The permutations of Perm are all finitely supported functions $\mathbb{A} \to \mathbb{A}$: compare Ex. 2.6(iii) and (5).
(ii) For each nominal set X, the map $x \mapsto supp(x)$ is an equivariant function $X \to \mathcal{P}_{fin}(\mathbb{A})$.
(iii) If X is a subset of the nominal set Y, and X is closed under Y's Perm-action, then we call X a *nominal subset* of Y; the inclusion function is obviously equivariant. For example, $X \otimes Y$ is a nominal subset of $X \times Y$.

Remark 2.9. $\mathcal{N}om$ has much categorial structure; it is a Grothendiek topos. We in particular note, without proof, that it has initial object $\emptyset$ and terminal object $1 = \{\bullet\}$ under the trivial Perm-actions (Ex. 2.6(i)), that Ex. 2.6(v) and (vii) define its binary product and coproducts, and that its exponential from X to Y is the nominal set of functions $X \to Y$ finitely supported under (5).

Lemma 2.10. *The following are useful basic facts about nominal sets. We will use them often in this paper, usually without specific reference:*

(i) Given an atom $a \in \mathbb{A}$, $a \# x$ if and only if for ***some*** *atom $a' \# (a, x)$, we have $(a\ a') \cdot x = x$, if and only if for* ***any*** *atom $a' \# (a, x)$, we have $(a\ a') \cdot x = x$*
(ii) If f is equivariant then $supp(f(x)) \subseteq supp(x)$.
(iii) If the permutations π, π' coincide on their restrictions to $supp(x)$, then $\pi \cdot x = \pi' \cdot x$.

We now present the standard atom-abstraction construction, which gives semantics for name binding operations such as λ-abstraction:

Definition 2.11. *Given a nominal set X we define a relation on $\mathbb{A} \times X$ by*

$$(a, x) \sim (a', x') \ \Leftrightarrow\ (a\ b) \cdot x = (a'\ b) \cdot x'$$

for some atom $b \# (a, a', x, x')$. This defines an equivalence relation; write the class containing (a, x) as $\langle a \rangle x$ and call such a class the atom-abstraction of a on x. *This relation is equivariant, i.e. $(a, x) \sim (a', x')$ implies $(\pi(a), \pi \cdot x) \sim (\pi(a'), \pi \cdot x')$.*

The set of atom-abstractions of a on x as a ranges over $\mathbb{A}$ and x over X hence forms a nominal set under the action $\pi \cdot \langle a \rangle x \triangleq \langle \pi(a) \rangle (\pi \cdot x)$. Write this nominal set $[\mathbb{A}]X$, and call its members the atom-abstractions on X.

This construction extends to a functor $[\mathbb{A}]\text{-} : \mathcal{N}om \to \mathcal{N}om$ by, given equivariant $f : X \to Y$ and atom-abstraction $\langle a \rangle x \in [\mathbb{A}]X$, the map $([\mathbb{A}]f)(\langle a \rangle x) = \langle a \rangle (f(x))$.

Definition 2.12. *The separated product construction $X \otimes Y$ of Ex. 2.6(vi) extends to a monoidal operation on $\mathcal{N}om$ under the evident action on equivariant functions. This gives rise to a functor $\text{-} \otimes X : \mathcal{N}om \to \mathcal{N}om$ for any X.*

Definition 2.13. *For any nominal sets X, Y define the nominal set of* freshening functions *from X to Y by*

$$Y_{[X]} \triangleq \{f : X \to Y \mid f \text{ is finitely supported, and } \forall x \in X.\, x \# f(x)\}\ .$$

Equivalently, $Y_{[X]}$ is the set of functions whose graphs draw elements from $X \otimes Y$.

This extends to a functor $\text{-}_{[X]} : \mathcal{N}om \to \mathcal{N}om$ in the evident manner: given equivariant $g : Y \to Z$ and freshening function $f \in Y_{[X]}$, we have $g_{[X]}(f) = g \circ f$.

Theorem 2.14. *We have the adjunctions*

$$\text{-} \otimes \mathbb{A} \ \dashv \ [\mathbb{A}]\text{-} \ \dashv \ \text{-}_{[\mathbb{A}]}$$

Proof. By Thms. 3.6, 4.5, and 5.5 below. We note here only that the co-unit of the left hand adjunction, with components $\varepsilon_X : [\mathbb{A}]X \otimes \mathbb{A} \to X$, is called *concretion*, and is defined by

$$\varepsilon_X(\langle a \rangle x, b) = (a\ b) \cdot x \ .$$

It is in this sense that atom-abstraction is destructed like a function space.

3 Separating Functions

In this section we will define the *closure* of the separated product $\otimes$: a binary connective $\multimap\!\!*$ with the adjoint property $\text{-} \otimes X \ \dashv \ X \multimap\!\!* \text{-}$ for any nominal set X. Such a closure condition can be compared to the standard cartesian closure relating products and function spaces. As $\otimes$ is a sort of product, we will not be surprised to find that $\multimap\!\!*$ forms a sort of function space.

Definition 3.1. *Given sets X, Y, the* partial functions *$f : X \rightharpoonup Y$ are the functions $f : dom(f) \to Y$ for $dom(f) \subseteq X$. We say $f(x) \downarrow$, and that $f(x)$* converges*, if $x \in dom(f)$. We say $f(x) \uparrow$, and that $f(x)$* diverges*, if $x \in X - dom(f)$.*

Given nominal sets X, Y and a partial function $f : X \rightharpoonup Y$, we can define a Perm*-action on f by the* Perm*-action on its graph, i.e.*

$$(\pi \cdot f)(x) = \begin{cases} \pi \cdot f(\pi^{-1} \cdot x) & \text{if } f(\pi^{-1} \cdot x) \downarrow \\ \uparrow & \text{if } f(\pi^{-1} \cdot x) \uparrow \end{cases} \tag{6}$$

The nominal set $X \rightharpoonup_{fs} Y$ is the set of partial functions $X \rightharpoonup Y$ finitely supported under (6)*.*

Definition 3.2. *Given nominal sets X, Y a* separating function *f from X to Y is a finitely supported partial function satisfying*

(i) $f(x) \downarrow$ if and only if $f \# x$;
(ii) $supp(f) = \bigcup_{x \in dom(f)} supp(f(x)) - supp(x)$;

This defines a nominal subset of $X \rightharpoonup_{fs} Y$, which we write $X \multimap\!\!* Y$ and call the separating function space *from X to Y.*

Lemma 3.3. *For any finitely supported partial function $f : X \rightharpoonup Y$, Def. 3.2(ii) is equivalent to*

$$supp(f) \subseteq \bigcup_{x \in dom(f)} supp(f(x)) - supp(x) \ . \tag{7}$$

Proof. The converse holds for any finitely supported f: Say $a \in supp(f(x)) - supp(x)$ for some $x \in dom(f)$. Take $a' \# (a, x, f)$, so $(a\ a') \cdot f(x) \neq f(x)$. To see that $a \in supp(f)$ we will show that $f, (a\ a') \cdot f$ disagree on x.

$a, a' \# x$ implies that $(a\ a') \cdot x = x$, so $f(x) \downarrow$ implies that $((a\ a') \cdot f)(x)$ converges to $(a\ a') \cdot f(x)$ by (6). This is not equal to $f(x)$, as required.

Remark 3.4. How is Def. 3.2 motivated? The proposed adjunction requires that we evaluate $f(x)$ only in the case that $(f, x) \in (X \multimap Y) \otimes X$; that is, where $f \# x$. It therefore must be that f is partial and that its domain be entirely determined by its support; otherwise we would not have total evaluation, or would have non-identical functions that evaluate identically. This restriction (here, Def. 3.2(i)) is also found in [19, Sec. 3.3].

A counter-example helps to motivate Def. 3.2(ii). Following the 'sharing interpretation' of the logic BI [18, Cha. 9], if we interpret names as resources, and supports as the resources claimed by each element, then $\otimes$ is a 'non-sharing' product, and we would expect its closure to consist of functions that cannot access their arguments' supports. But consider the partial function $\uparrow_a : \mathbb{A} \rightharpoonup 1$ that diverges on a and converges elsewhere. This obeys Def. 3.2(i), as $supp(\uparrow_a) = \{a\}$, yet it needs to access its argument's support to determine divergence. However the right hand side of (7) is empty, so $\uparrow_a \notin \mathbb{A} \multimap 1$, so Def. 3.2(ii) fails.

Example 3.5. (i) If all elements of Y have empty support, then $X \multimap Y$ contains exactly the equivariant total functions $X \to Y$. In particular, $X \multimap 1 \cong 1$.

(ii) The separating functions $\mathbb{A} \multimap \mathbb{A}$ are the identity and, for each $a \in \mathbb{A}$, the partial functions f_a defined by $f_a(b) = a$ if $a \neq b$, and diverging on a.

(iii) The separating functions $(1 + \mathbb{A}) \multimap \mathbb{A}$ are defined, for each $a \in \mathbb{A}$, by mapping $\bullet \mapsto a$ and then (1) as the identity on $\mathbb{A}$ except diverging on a, (2) sending all atoms to a except diverging on a, or (3) for any $b \neq a$, sending all atoms to b except diverging on $\{a, b\}$.

Theorem 3.6. *The definition of separating function spaces extends to a functor $X \multimap - : \mathcal{N}om \to \mathcal{N}om$ for any nominal set X, with the adjoint property*

$$- \otimes X \ \dashv \ X \multimap -$$

Proof. The bijection $\mathcal{N}om(Z \otimes X, Y) \cong \mathcal{N}om(Z, X \multimap Y)$ is given via the co-unit, whose components $\varepsilon_Y : (X \multimap Y) \otimes X \to Y$ are the usual evaluation functions:

$$\varepsilon_Y(f, x) = f(x)$$

Each ε_Y is straightforwardly total and equivariant. We must show that for any $f : Z \otimes X \to Y$ there is a unique $\hat{f} : Z \to (X \multimap Y)$ such that

$$\begin{array}{ccc} Z & & Z \otimes X \\ \downarrow{\scriptstyle \hat{f}} & & \downarrow{\scriptstyle \hat{f}\otimes X} \searrow^{f} \\ X \multimap Y & & (X \multimap Y) \otimes X \xrightarrow[\varepsilon_Y]{} Y \end{array} \tag{8}$$

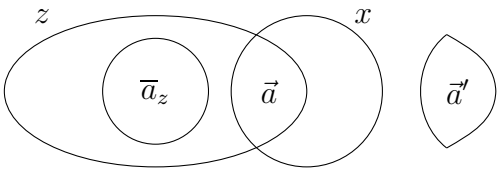
commutes. For each $z \in Z$ let

$$\overline{a}_z \triangleq \bigcup_{(x \in X)\#z} supp(f(z,x)) - supp(x) \ . \tag{9}$$

The map $z \mapsto \overline{a}_z$ is straightforwardly equivariant, so $\overline{a}_z \subseteq supp(z)$ by Lem. 2.10(ii). Now let $\hat{f} : Z \to (X \multimap Y)$ be

$$\hat{f}(z)(x) \triangleq \begin{cases} f((\vec{a}\ \vec{a}') \cdot z, x) & \text{if } \overline{a}_z \# x \\ \uparrow & \text{otherwise.} \end{cases} \tag{10}$$

where $\vec{a}$ is an ordering of $supp(z) \cap supp(x)$ and $\vec{a}'$ is a tuple of the same size fresh for (z, x). Then where $\hat{f}(z)(x) \downarrow$ we may picture the elements' supports as



Evidently $(\vec{a}\ \vec{a}') \cdot z \# x$, so $f((\vec{a}\ \vec{a}') \cdot z, x)$ is well-defined. To see that $\hat{f}$ is **well-defined** we must check that it does not depend on our choice of fresh $\vec{a}'$. To do this we first prove

$$\vec{a} \# f(z, (\vec{a}\ \vec{a}') \cdot x) \ . \tag{11}$$

Suppose b is in the support of both sides of (11). Now $(b, z) \# (\vec{a}\ \vec{a}') \cdot x$, so $b \in \overline{a}_z$ by (9). But $\overline{a}_z \# x$ and $b \in supp(x)$ by definition, so by contradiction (11) holds. Because f is equivariant we can then apply $(\vec{a}\ \vec{a}')$ to both sides of (11) to get $\vec{a}' \# f((\vec{a}\ \vec{a}') \cdot z, x)$, which is sufficient to conclude that our choice of fresh $\vec{a}'$ is arbitrary. $\hat{f}$ is hence a function; equivariance is straightforward.

The diagram (8) **commutes** because if we start at $(z, x) \in Z \otimes X$ then $\overline{a}_z \# x$, and $\vec{a}$ in (10) is empty, so $\hat{f}(z)(x)$ converges to $f(z, x)$.

We next confirm that $\hat{f}(z)$ is a **separating function**. $\hat{f}(z)(x) \downarrow$ iff $\overline{a}_z \# x$ by (10), so Def. 3.2(i) holds if $supp(\hat{f}(z)) = \overline{a}_z$. Def. 3.2(ii) asks further that $supp(\hat{f}(z))$ equals

$$\bigcup_{x \in dom(\hat{f}(z))} supp(\hat{f}(z)(x)) - supp(x) \ . \tag{12}$$

We start by showing that $\overline{a}_z$ equals (12). Taking $a \in \overline{a}_z$, there exists $x \in X$ such that $x \# z$ and $a \in \mathit{supp}(f(z,x)) - \mathit{supp}(x)$. Then $\hat{f}(z)(x) = f(z,x)$, so a is in (12). Conversely, take a in (12). There exists $x \in \mathit{dom}(\hat{f}(z))$ such that $a \in \mathit{supp}(\hat{f}(z)(x)) - \mathit{supp}(x)$. Now $\hat{f}(z)(x) = f((\vec{a}\ \vec{a}') \cdot z, x)$, and $a \# x$ implies $a \# \vec{a}$, and f is equivariant, so we can apply $(\vec{a}\ \vec{a}')$ to both sides of $a \in \mathit{supp}(\hat{f}(z)(x)) - \mathit{supp}(x)$ to yield $a \in \mathit{supp}(f(z, (\vec{a}\ \vec{a}') \cdot x)) - \mathit{supp}((\vec{a}\ \vec{a}') \cdot x)$. But $(\vec{a}\ \vec{a}') \cdot x \# z$, so $a \in \overline{a}_z$ by (9).

By Lem 3.3 we need only now show (by contrapositive) that $\mathit{supp}(\hat{f}(z)) \subseteq \overline{a}_z$. The equivariance of $\hat{f}$ and Lem. 2.10(ii) tell us that $a \# z$ implies $a \# \hat{f}(z)$, so we need only consider $a \in \mathit{supp}(z) - \overline{a}_z$. We will show that, given fresh atom a',

$$\hat{f}(z) = (a\ a') \cdot \hat{f}(z)\ .$$

First, convergence: for any $x \in X$, $\hat{f}(z)(x) \downarrow$ iff $\overline{a}_z \# x$ by (10), iff $\overline{a}_z \# (a\ a') \cdot x$ because $a, a' \# \overline{a}_z$, iff $\hat{f}(z)((a\ a') \cdot x) \downarrow$, iff $((a\ a') \cdot \hat{f}(z))(x) \downarrow$ by (6).

Now, taking any x on which they converge, we will show that $\hat{f}(z)(x) = ((a\ a') \cdot \hat{f}(z))(x)$. By $\hat{f}$'s equivariance and (10), this asks that

$$f((\vec{a}\ \vec{a}') \cdot z, x) = f((\vec{b}\ \vec{b}')(a\ a') \cdot z, x) \tag{13}$$

where $\vec{a}$ orders $\mathit{supp}(z) \cap \mathit{supp}(x)$, $\vec{b}$ orders $\mathit{supp}((a\ a') \cdot z) \cap \mathit{supp}(x)$, and $\vec{a}', \vec{b}'$ are chosen fresh. If $a \in \mathit{supp}(z) \cap \mathit{supp}(x)$ then $(\vec{a}\ \vec{a}')$ and $(\vec{b}\ \vec{b}')(a\ a')$ coincide (given an appropriate choice of fresh variables), so (13) holds. Otherwise, say $a \# x$, in which case $\vec{a} = \vec{b}$. Now because $a \notin \overline{a}_z$ and $x \in \mathit{dom}(\hat{f}(z))$, we have $a \notin \mathit{supp}(f((\vec{a}\ \vec{a}') \cdot z, x)) - \mathit{supp}(x)$ by (9), and so $a \# f((\vec{a}\ \vec{a}') \cdot z, x)$. Hence $f((\vec{a}\ \vec{a}') \cdot z, x) = (a\ a') \cdot f((\vec{a}\ \vec{a}') \cdot z, x) = f((a\ a')(\vec{a}\ \vec{a}') \cdot z, (a\ a') \cdot x)$. This is the right hand side of (13) because $a \# (x, \vec{a})$.

Finally, confirmation that $\hat{f}$ is **uniquely determined** is routine.

Corollary 3.7. *$X \multimap -$ extends to a bifunctor $- \multimap - : \mathcal{N}om^{op} \times \mathcal{N}om \to \mathcal{N}om$.*

Proof. Given equivariant $g : X' \to X, h : Y \to Y'$, apply the adjunction to

$$(X \multimap Y) \otimes X' \xrightarrow{id \otimes g} (X \multimap Y) \otimes X \xrightarrow{\varepsilon_Y} Y \xrightarrow{h} Y'$$

Theorem 3.8. *$X \multimap -$ has no right adjoint in general.*

Proof. It suffices to find some nominal set X, and some colimit in $\mathcal{N}om$, such that this colimit is not preserved by $X \multimap -$. Now $\mathbb{A}^2 \multimap 1 \cong 1$, so $(\mathbb{A}^2 \multimap 1) + (\mathbb{A}^2 \multimap 1)$ has two elements. $\mathbb{A}^2 \multimap (1 + 1)$, on the other hand, contains four elements: the maps $(a, b) \mapsto (\bullet, i)$ for $i = 1$ or 2, the map

$$(a, b) \mapsto \begin{cases} (\bullet, 1) & \text{if } a = b \\ (\bullet, 2) & \text{otherwise} \end{cases}$$

and its converse.

4 Generalised Abstraction

Lemma 4.1. *Given nominal sets X, Y, we can define an equivalence relation $\sim$ on $X \times Y$ by setting, for all permutations $\pi \# (supp(y) - supp(x))$,*

$$(x, y) \sim \pi \cdot (x, y)$$

recalling that $\pi \cdot (x, y) = (\pi \cdot x, \pi \cdot y)$.

Proof. For reflexivity, set $\pi = \iota$. For symmetry, $(\pi \cdot x, \pi \cdot y) \sim \pi^{-1} \cdot (\pi \cdot x, \pi \cdot y)$ by the equivariance of *supp* and set minus. Transitivity is similarly straightforward.

Definition 4.2. *Write $X \times Y$ modulo $\sim$ as $[X]Y$. Call its members the X-*abstractions on Y. *Write the equivalence class containing (x, y) as $\langle x \rangle y$. Call the construction, as X ranges across all nominal sets,* generalised abstraction.

Lemma 4.3. *$[X]Y$ is a nominal set under the action $\pi \cdot \langle x \rangle y = \langle \pi \cdot x \rangle (\pi \cdot y)$, and*

$$supp(\langle x \rangle y) = supp(y) - supp(x)$$

Proof. Standard nominal techniques.

Example 4.4. (i) $[\mathbb{A}]X$ is the familiar notion of atom-abstraction from Def. 2.11. For example, $[\mathbb{A}]\mathbb{A}$ contains the emptily supported element $\langle a \rangle a = \langle b \rangle b = \cdots$ and, for each atom a, the element $\langle b \rangle a = \langle c \rangle a = \cdots$ for $b, c, \ldots \neq a$, supported by $\{a\}$. Compare with Ex. 3.5(ii).

(ii) $[\mathbb{A} + 1]\mathbb{A}$ contains all the elements of $[\mathbb{A}]\mathbb{A}$ plus, for each $a \in \mathbb{A}$, the element $\langle \bullet \rangle a$ supported by $\{a\}$. Contrast with Ex. 3.5(iii), which had no emptily supported element.

Theorem 4.5. *Generalised abstraction extends to a functor $[X]\text{-} : \mathcal{N}om \to \mathcal{N}om$ for any nominal set X, with the adjoint property*

$$[X]\text{-} \dashv \text{-}_{[X]}$$

where $\text{-}_{[X]}$ is the freshening function *functor of Def. 2.13*

Proof. In the case $X = \mathbb{A}$, the proof is [8, Thm. 9.6.6], although the language of adjunctions is not explicitly used there. The general situation [17] holds similarly: the bijection $\mathcal{N}om(Y, Z_{[X]}) \cong \mathcal{N}om([X]Y, X)$ is defined by mapping equivariant $f : Y \to Z_{[X]}$ to $g : [X]Y \to Z$, where $g(\langle x \rangle y) = f(y)(x)$. The unit of the adjunction has components $\eta_Y : Y \to ([X]Y)_{[X]}$ defined by $\eta_Y(y)(x) = \langle x \rangle y$.

Theorem 4.6. *$[X]-$ has no left adjoint in general.*

Proof. It suffices to find some limit in $\mathcal{N}om$ not preserved by $[X]\text{-}$ for some X. $[\mathbb{A}^2]1$ has two distinct objects:

(i) $\langle (a, a) \rangle \bullet$;
(ii) $\langle (a, b) \rangle \bullet$ (where $a \neq b$).

They are not equal because there is no permutation π such that $\pi \cdot ((a,a),\bullet) = ((a,b),\bullet)$. $[\mathbb{A}^2]1$ is therefore not terminal.

The next theorem gives a novel description of generalised abstractions as (isomorphic to) certain nominal subsets of finite atom-set abstractions.

Theorem 4.7. *Given nominal sets X, Y, $[X]Y$ is the equaliser of functions $\top, f$ from $[\mathcal{P}_{fin}(\mathbb{A})](X \times Y)$ to $2 = \{\top, \bot\}$, defined as:*

(i) The constant function $\top$;

(ii) $f(\langle\overline{a}\rangle(x,y)) = \begin{cases} \top & \text{if } supp(x) = \overline{a} \\ \bot & \text{otherwise.} \end{cases}$

Proof. The forgetful functor $\mathcal{N}om \to \mathcal{S}et$ reflects equalisers, so we need only observe that $[X]Y$ is isomorphic to the usual equaliser via the bijective map $\langle x\rangle y \mapsto \langle supp(x)\rangle(x,y)$.

5 When Do Separating Functions and Generalised Abstraction Coincide?

From Exs. 3.5(iii) and 4.4(ii) and Thms. 3.8 and 4.6, we see that $X \multimap Y$ and $[X]Y$ are not always isomorphic. Thm. 5.5 will specify when they do coincide.

Definition 5.1. *A nominal set X is* transitive *if for all $x, y \in X$, there exists $\pi \in \mathsf{Perm}$ such that $\pi \cdot x = y$.*

X is strong[1] *if, for all $\pi \in \mathsf{Perm}$, $\pi \cdot x = x$ implies $\pi \# x$.*

Example 5.2. (i) $\mathbb{A}$ is transitive and strong.

(ii) $\mathcal{P}_2(\mathbb{A})$, the nominal set of unordered pairs of atoms with the element-wise Perm-action, is transitive but not strong: $(a\ b) \cdot \{a,b\} = \{a,b\}$, but it is not the case that $(a\ b) \# \{a,b\}$.

(iii) $\mathbb{A}^2$ is strong but not transitive: (a,a) and (a,b), where $a \neq b$, occupy different orbits.

(iv) $\mathcal{P}_{fin}(\mathbb{A})$ is neither transitive nor strong.

Lemma 5.3. *Say $X \multimap Y \cong [X]Y$ for all Y. Then*

(i) X is non-empty;

(ii) X is transitive;

(iii) X is strong.

Proof. (i) $\emptyset \multimap Y$ contains the empty function, while $[\emptyset]Y$ is empty always.

(ii) Say we have $x, x' \in X$ with no permutation between them. $\langle x\rangle\bullet$ and $\langle x'\rangle\bullet$ are therefore distinct elements of $[X]1$, which then differs from $X \multimap 1 \cong 1$.

[1] This property was introduced by [19] and called *essentially simple*; we use the more widely used *strong* from [23].

(iii) Say X is not strong, so we have $\pi \cdot x = x$ with $supp(\pi) \cap supp(x) \neq \emptyset$. Say $supp(x) - supp(\pi)$ has n elements, and suppose for contradiction that there existed a total equivariant function $f : X \to \mathbb{A}^{(n+1)}$. $supp(f(x)) \subseteq supp(x)$ by Lem. 2.10(ii), and because $supp(f(x))$ has $n+1$ elements it is too big to contain just $supp(x) - supp(\pi)$; it must contain some $a \in supp(x) \cap supp(\pi)$. Now $\pi \cdot f(x) = f(\pi \cdot x) = f(x)$, but this Perm-action is just the element-wise action on lists, so $\pi(a) = a$. This contradicts $a \in supp(\pi)$, so there are no equivariant functions $X \to \mathbb{A}^{(n+1)}$. But the total equivariant functions are the only emptily supported elements of separating function spaces, so $X \twoheadrightarrow \mathbb{A}^{(n+1)}$ has no emptily supported elements.

Conversely, let $\vec{a}$ be an ordering of $(supp(x) - supp(\pi)) \cup \{a\}$, where $a \in supp(\pi) \cap supp(x)$. Then $\langle x \rangle \vec{a}$ is an emptily supported element of $[X](\mathbb{A}^{(n+1)})$ by Lem. 4.3.

Lemma 5.4. *Say X is transitive and strong. Then we can define an equivariant function, called* generalised concretion, *$[X]Y \otimes X \to Y$ sending $(\langle x \rangle y, \pi \cdot x) \mapsto \pi \cdot y$, where $\pi \# \langle x \rangle y$.*

Proof. We first check that for any $(\langle x \rangle y, x') \in [X]Y \otimes X$ there exists such a permutation π. Because X is transitive there exists a permutation π' such that $\pi' \cdot x = x'$. Let $\vec{a}$ order $supp(\pi') \cap supp(\langle x \rangle y)$, so $\vec{a} \# (x, x')$, and let $\vec{a}'$ be a fresh copy. Then $(\vec{a}\ \vec{a}')\pi'(\vec{a}\ \vec{a}') \cdot x = (\vec{a}\ \vec{a}')\pi' \cdot x = (\vec{a}\ \vec{a}') \cdot x' = x'$. For any $a \in supp(\langle x \rangle y)$ either $a \# \pi'$ or $a \in supp(\vec{a})$; either way $a \# (\vec{a}\ \vec{a}')\pi'(\vec{a}\ \vec{a}')$.

Concretion does not depend on choice of permutation: say $(\pi, \pi') \# \langle x \rangle y$ and $\pi \cdot x = \pi' \cdot x$. Then $\pi^{-1}\pi' \cdot x = x$, and so, because X is strong, $\pi^{-1}\pi' \# x$. But $supp(\pi^{-1}\pi') \subseteq supp(\pi, \pi')$, and this is fresh for $\langle x \rangle y$, so $\pi^{-1}\pi' \# y$, so $\pi \cdot y = \pi' \cdot y$. Concretion also does not depend on choice of representative: say $(x, y) \sim \pi \cdot (x, y)$, and $(\langle x \rangle y, \pi' \cdot x) \mapsto \pi' \cdot y$. Applying π to both sides of $\pi^{-1}\pi' \# \langle x \rangle y$ yields $\pi'\pi^{-1} \# \pi \cdot \langle x \rangle y$, and so concretion maps $(\pi \cdot \langle x \rangle y, \pi'\pi^{-1} \cdot (\pi \cdot x)) \mapsto \pi'\pi^{-1} \cdot (\pi \cdot y) = \pi' \cdot y$. Concretion is then well-defined; equivariance is straightforward.

Theorem 5.5. *$X \twoheadrightarrow Y \cong [X]Y$ iff X is non-empty, transitive and strong.*

Proof. The left-to-right direction is Lem. 5.3. Conversely, the map $g : [X]Y \to (X \twoheadrightarrow Y)$ is got by applying the adjunction of Thm. 3.6 to the generalised concretion of Lem. 5.4. Unpacking the adjunction, $\overline{a}_{\langle x \rangle y} = supp(\langle x \rangle y)$, and so

$$g(\langle x \rangle y)(\pi \cdot x) = \begin{cases} \pi \cdot y & \text{if } \pi \cdot x \# \langle x \rangle y \\ \uparrow & \text{otherwise.} \end{cases} \tag{14}$$

The converse map $h : (X \twoheadrightarrow Y) \to [X]Y$ is $h(f) \triangleq \langle x \rangle (f(x))$ for $x \# f$. Such an x exists because X is non-empty. Equivariance of h is easy, but we must confirm that h does not depend on our choice of x: given $x, x' \# f$ there exists π such that $\pi \# f$ and $\pi \cdot x = x'$. Then $\langle x' \rangle (f(x')) = \pi \cdot \langle x \rangle (f(x))$, which equals $\langle x \rangle f(x)$ because $\pi \# f$ implies $\pi \# supp(f(x)) - supp(x)$ by Lem. 3.3.

$h \circ g(\langle x \rangle y) = \langle \pi \cdot x \rangle (g(\langle x \rangle y)(\pi \cdot x))$ for $\pi \cdot x \# \langle x \rangle y$ and, without loss of generality, $\pi \# \langle x \rangle y$. This yields $\pi \cdot (\langle x \rangle y) = \langle x \rangle y$. Conversely,

$(g \circ h(f))(x) = g(\langle \pi \cdot x \rangle f(\pi \cdot x))(x)$ for $\pi \cdot x \# f$. Say $x \# f$; then by Def. 3.2, $x \# supp(f(\pi \cdot x)) - supp(\pi \cdot x)$, so $x \# \langle \pi \cdot x \rangle f(\pi \cdot x)$. Hence by (14), $(g \circ h(f))(x) = \pi^{-1} \cdot f(\pi \cdot x)$. But without loss of generality $\pi \# f$, so $\pi^{-1} \cdot f(\pi \cdot x) = f(x)$. If x is not fresh for f then $(g \circ h(f))(x) \uparrow$ as required.

Remark 5.6. The proof given under Thm. 5.5 is presented in more abstract form by [19, Prop. 10.3.7], which shows that non-empty, transitive, strong nominal sets - called there *name-like objects* - produce a 'category with binding structure'. The converse (here, Lem. 5.3) is, however, new to this paper. We also note that [19, Lem. 10.3.6] gives a succinct description of name-like objects in $\mathcal{N}om$: a nominal set is non-empty, transitive and strong iff it is isomorphic to $\mathbb{A}^{(n)}$ for $n \geq 0$. This gives an easy criterion for which our generalisations of name binding coincide, so that they may be constructed like a pair and destructed like a partial function.

6 Applications and Further Work

The logic of Bunched Implications (BI). The categorial structure specified in Sec. 2, and by Thm. 3.6, makes $\mathcal{N}om$ a bi-cartesian doubly closed category, and hence a model of BI [18]. BI is a logic where additive intuitionistic logic sits alongside multiplicative (substructural) intuitionistic logic. The additive logic is interpreted by cartesian closure and finite coproducts, while the multiplicative logic is interpreted by $\otimes$, $-\!\!*$, and $\otimes$'s identity. Now $\otimes$ has as identity the terminal object 1, which makes $\mathcal{N}om$ a model of *affine* BI, where we have *weakening* via canonical 'projection' functions $X \leftarrow X \otimes Y \rightarrow Y$, but do not have *contraction*, as for example there is no equivariant function $\mathbb{A} \rightarrow \mathbb{A} \otimes \mathbb{A}$.

We have thus described an appealingly concrete model for (affine) BI. It is, in fact, very close to the functor category $\mathcal{S}et^{\mathcal{I}}$ discussed in [18, Sec. 9.3], where $\mathcal{I}$ is the category of finite sets (without loss of generality, finite sets of atoms), and injections between them. We can think of such a functor as mapping each finite $\overline{a} \subseteq \mathbb{A}$ to the subset of elements it supports. $\mathcal{N}om$ is known to be equivalent to the category of *pullback-preserving* functors $\mathcal{I} \rightarrow \mathcal{S}et$, called the *Schanuel topos*. Pullback preservation means that elements' supports are closed under intersection, necessary for the definition of freshness. The inclusion functor $cell : \mathcal{I} \rightarrow \mathcal{S}et$ of [18, Sec. 9.3] then corresponds to the nominal set of atoms $\mathbb{A}$. We hope that the concrete constructions of this paper will facilitate the application of BI to reasoning about names as resources, continuing the work of [20].

$\mathcal{N}om$-enriched Categories. $\mathcal{N}om$ has two monoidal products $\times, \otimes$ with which it might make sense to explore *enriched category theory* [14]. For example, it is $\otimes$ that is used in [7, Sec. 5] to define their 'freshness environments'. As both these monoidal products are closed, we have two different notions of *internal hom* induced, for which we now have two concrete descriptions.

Nominal Isabelle. The Nominal Isabelle package for interactive theorem proving over abstract syntax now supports various notions of generalised binding [24]. Perhaps surprisingly, this work is not explicitly based on the established

notion of generalised abstraction. Nonetheless it is clear that [24]'s set-binding and list-binding are the abstractions of $\mathcal{P}_{fin}(\mathbb{A})$, and the nominal set $\mathbb{A}^*$ of finite lists of atoms, respectively. It is hoped this paper will bring the concept of generalised abstraction back into view, and that our new Thm. 4.7 will help unify this notion. Nominal Isabelle's notion of 'set+-binding' goes further than $\mathcal{P}_{fin}(\mathbb{A})$-abstraction by ignoring 'vacuous binders', so for example $\langle\{a\}\rangle b \approx_\alpha \langle\emptyset\rangle b$ for $a \neq b$. This is a quotient of the $\mathcal{P}_{fin}(\mathbb{A})$-abstraction.

The Pure Pattern Calculus and Other Pattern Binding. [12] presents a formal calculus for pattern matching[2], with term syntax

$$t \ ::= \ a \ \mid \ t\,t \ \mid \ t \to_{\overline{a}} t$$

where $a \in \mathbb{A}$ and $\overline{a} \in \mathcal{P}_{fin}(\mathbb{A})$. We call a term $p \to_{\overline{a}} t$ a *case*, linking the *pattern* p to the *body* t via the *binding variables* $\overline{a}$. Free atoms are defined by

$$fa(a) \ \triangleq \ a, \quad fa(t\,u) \ \triangleq \ fa(t) \cup fa(u), \quad fa(p \to_{\overline{a}} t) \ \triangleq \ (fa(p) \cup fa(t)) - \overline{a} \ . \tag{15}$$

α-conversion is the congruence generated by

$$p \to_{\overline{a}} t \ \approx_\alpha \ \{a \leftarrow b\}p \to_{\{a \leftarrow b\}\overline{a}} \{a \leftarrow b\}t$$

where $b \notin fa(p) \cup fa(t) \cup \overline{a}$ and where $\{a \leftarrow b\}$ is the substitution of b for a. As b is chosen fresh we can use permutations $(a\ b)$ instead of substitutions, and hence prove that the nominal set of pure pattern calculus terms quotiented by α-equivalence is the initial algebra for this endofunctor on $\mathcal{N}om$:

$$F \ \triangleq \ \mathbb{A} + (\text{-} \times \text{-}) + [\mathcal{P}_{fin}(\mathbb{A})](\text{-} \times \text{-}) \ .$$

Other inductive definitions in [12], such as the free atom function (15) above, are also F-algebras, with their actions on terms defined in the usual way as the unique homomorphism from the initial algebra. The final coalgebra for F gives, as we would expect, a sensible notion of infinitary terms (following e.g. [11]).

This calculus also offers a convenient piece of syntactic sugar:

$$p \to t \ \triangleq \ p \to_{fa(p)} t \ .$$

We could ask what would happen if we took this sort of construction as basic, rather than sugar, as is quite common - see for example the ρ-calculus [2]. As [1] says, "binding all of the distinct names of a term in another term seems to be a common enough case to deserve special attention and notation". Generalised abstraction provides exactly this notion. However induction over this construction becomes problematic. We would like to have a functor that acts on nominal sets by

$$G\,X \ = \ [X]X \ .$$

[2] A more recent formulation of pattern calculus [13] makes a distinction between *variables* and *matchables* which adds complexity we do not attempt to discuss here.

However it is not clear to us that such a bifunctor [-]- can be defined on the arrows of $\mathcal{N}om$; we therefore do not have a generalised abstraction variant of Cor. 3.7. The naive definition $G\,f(\langle x\rangle x') = \langle f(x)\rangle f(x')$ fails because equivariant functions can shrink supports of the abstracted elements. For example, given $\bullet + id : \mathbb{A} + \mathbb{A} \to 1 + \mathbb{A}$, we would have $G(\bullet + id)$ mapping $\langle (a, 1)\rangle (a, 2) \mapsto \langle \bullet \rangle a$, and this sends an emptily supported element to an non-emptily supported element, violating Lem. 2.10(ii). Hence even though standard category theory gives us a notion of induction on $\mathcal{P}_{fin}(\mathbb{A})(Tm \times Tm)$, where Tm is the set of terms modulo α-equivalence, we do not yet have such a notion for its nominal subset $[Tm]Tm$. Finding a robust induction principle for such constructions is a topic for future research.

References

1. Cheney, J.: Towards a general theory of names, binding and scope. In: MERLIN, pp. 33–40. ACM (2005)
2. Cirstea, H., Kirchner, C.: The rewriting calculus - part I. Log. J. IGPL 9(3), 339–375 (2001)
3. Clouston, R.: Equational Logic for Names and Binders. Ph.D. thesis, University of Cambridge (2009)
4. Clouston, R., Pitts, A.M.: Nominal equational logic. ENTCS 172, 223–257 (2007)
5. Day, B.: On closed categories of functors. Lecture Notes in Math. 137, 1–38 (1970)
6. Dowek, G., Gabbay, M.J.: From nominal sets binding to functions and λ-abstraction: connecting the logic of permutation models with the logic of functions, arXiv (2011)
7. Fiore, M., Hur, C.K.: Term equational systems and logics. In: MFPS. ENTCS, vol. 218, pp. 171–192 (2008)
8. Gabbay, M.J.: A Theory of Inductive Definitions with Alpha-Equivalence. Ph.D. thesis, Cambridge University (2001)
9. Gabbay, M.J.: FM-HOL, a higher-order theory of names. In: 35 Years of Automath (2002)
10. Gabbay, M.J., Pitts, A.M.: A new approach to abstract syntax with variable binding. Formal Aspects Comput. 13, 341–363 (2002)
11. Jacobs, B., Rutten, J.: A tutorial on (co)algebras and (co)induction. BEATCS 62, 222–259 (1997)
12. Jay, B., Kesner, D.: Pure Pattern Calculus. In: Sestoft, P. (ed.) ESOP 2006. LNCS, vol. 3924, pp. 100–114. Springer, Heidelberg (2006)
13. Jay, B., Kesner, D.: First-class patterns. J. Funct. Program. 19, 191–225 (2009)
14. Kelly, G.M.: Basic concepts of enriched category theory, LMS Lecture Note Series, vol. 64. Cambridge University Press (1982)
15. Menni, M.: About $\mathcal{U}$-quantifiers. Appl. Categor. Struct. 11(5), 421–445 (2003)
16. Pitts, A.M.: Nominal Logic: A First Order Theory of Names and Binding. In: Kobayashi, N., Babu, C. S. (eds.) TACS 2001. LNCS, vol. 2215, pp. 219–242. Springer, Heidelberg (2001)
17. Pitts, A.M.: Nominal sets, the metamathematics of names (2011) (unpublished manuscript)
18. Pym, D.J.: The Semantics and Proof Theory of the Logic of Buunched Implications. Applied Logic Series, vol. 26. Kluwer Academic Publishers (2002)

19. Schöpp, U.: Names and Binding in Type Theory. Ph.D. thesis, University of Edinburgh (2006)
20. Schöpp, U., Stark, I.: A Dependent Type Theory with Names and Binding. In: Marcinkowski, J., Tarlecki, A. (eds.) CSL 2004. LNCS, vol. 3210, pp. 235–249. Springer, Heidelberg (2004)
21. Sewell, P., Nardelli, F.Z., Owens, S., Peskine, G., Ridge, T., Sarkar, S., Strnisa, R.: Ott: Effective tool support for the working semanticist. J. Funct. Program. 20(1), 71–122 (2010)
22. Shinwell, M.R., Pitts, A.M., Gabbay, M.J.: FreshML: Programming with binders made simple. In: ICFP. SIGPLAN Notices, vol. 38, pp. 263–274 (2003)
23. Tzevelekos, N.: Nominal Game Semantics. Ph.D. thesis, University of Oxford (2008)
24. Urban, C., Kaliszyk, C.: General Bindings and Alpha-Equivalence in Nominal Isabelle. In: Barthe, G. (ed.) ESOP 2011. LNCS, vol. 6602, pp. 480–500. Springer, Heidelberg (2011)