Under consideration for publication in Math. Struct. in Comp. Science

Two-level nominal sets and semantic nominal terms: an extension of nominal set theory for handling meta-variables

Murdoch J. Gabbay (gabbay.org.uk)

*Received August 2010; Revised 7 February 2011**

Nominal sets are a sets-based first-order denotation for variables in logic and programming. In this paper we extend nominal sets to *two-level nominal sets*. These preserve much of the behaviour of nominal sets—including notions of variable and abstraction—but they include a denotation for variables *and meta-variables*.

Meta-variables are interpreted as infinite lists of distinct variable symbols. We use two-level sets to define, amongst other things, a denotation for meta-variable abstraction, and nominal style datatypes of syntax-with-binding with meta-variables. We discuss the connections between this and nominal terms and prove a soundness result.

Contents

* This version differs from the publisher's version by including corrections suggested by Claus-Peter Wirth. *23 June 2011*

1. Introduction

This paper is about generalising nominal sets and linking this to the denotational semantics of unknowns (meta-variables) in nominal terms. Most of the paper is devoted to introducing two-level nominal sets and doing concrete sets calculations on them. We then apply the results thus obtained to construct what is essentially a soundness proof for a generalisation of nominal terms. This generalisation includes nominal terms (Urban et al. 2004), permissive-nominal terms (Dowek et al. 2010), and the syntax of predicates in permissive-nominal logic (Dowek and Gabbay 2010).

Thus, this paper presents a new semantic theory for nominal terms and their generalisations as considered by the author over the past several years.

To set this work in its general context, we will take a short detour through meta-programming and logical specification.

1.1. *A word on multi-level syntax*

Meta-programming and logical specification are concerned with specifying computation and logic respectively, and then reasoning about them (this is of course the topic of mathematical computer science). One design question is whether there should be two (or more) distinct kinds of variable in the system—one for the meta-level reasoning system and one for the object-level system—or whether there should be just one kind of variable.

First-order logic is a one-level system; it has only one kind of variable. Axiomatisations of logic and computation within first-order logic include, for instance, combinators, lambda abstraction algebras, and cylindric algebra (Hindley and Seldin 2008; Manzonetto and Salibra 2010; Henkin et al. 1971 and 1985). Arguably, higher-order logic and the λ-calculus are also one-level systems. These underlie, for instance, LCF and the Isabelle theorem prover (Paulson 1990). Having only one level is no barrier to having

unorthodox control of variables; see for example Adbmal (Hendriks and van Oostrom 2003). Having only one level of variable is no barrier to doing meta-level reasoning; consider that ZF set theory is axiomatised in Isabelle using an axiomatisation of first-order logic in higher-order logic.

Yet if we want to do meta-programming or meta-reasoning, having two (or more) levels of variable can be useful; one level for the object-system, the other for the system we use to reason or program about it.

In (Gacek et al. 2009, Section 2) these two levels are called *variables* and *nominal constants*. The authors trace this to a previous paper on LG^{ω} (Tiu 2007), though their ideas and terminology are also closely related to the nominal atoms of nominal techniques introduced in (Gabbay and Pitts 2001), which also underlie the work in this paper. Similarly the ∇-quantifier of Miller and Tiu's work is very similar to the Gabbay-Pitts $\mathcal{N}$-quantifier.[1]

Many other examples of multi-level systems exist. Thus for example we have Talcott's 'theory of binding structures' (Talcott 1993), a family of λ-calculi by Sato *et al* with levels of variable explicitly designed to model meta-variables (Sato et al. 2003); the ? and ! variables of McBride's thesis (McBride 1999); the 'holes with binding power' of Jojgov's thesis and subsequent work (Jojgov 2002); Muñoz's 'variables and place-holders' (noz 1997); Bognar's study of contexts in the λ-calculus (Bognar 2002); the term equational systems of Fiore and Hur (Fiore and Hur 2008); and Cardelli *et al*'s work on trees with hidden labels (Cardelli et al. 2003).

Similar issues have arisen and continue to arise with research into contexts, modules, object-oriented programming, incremental program construction, dynamic binding, proof-search, and more. Arguably, the γ and δ variables which appear in work by Wirth and others are an instance of a two-level system (Wirth 2004); more on this in the Conclusions. Also arguably the multiple levels of variable in MetaML (Moggi et al. 1999) are a multi-level system. These lists are not intended to be exhaustive.

There is a need here, repeatedly manifested over many decades of research and many different fields, for syntax and semantics with 'holes'. There is no consensus for what 'holes' are, nor agreement over what to call them, but there are clearly some thing or things out there and they are not just variables in first- or higher-order logic.

1.2. *Nominal terms*

Nominal terms are a formal syntax that takes a two-level approach very seriously, making it a centrepiece of their design.

Three features make nominal terms special:

— two levels of variable and an unusual capturing substitution,

[1] We believe that a process of convergent evolution may be taking place; the ∇-quantifier and its nominal constants are very similar to the $\mathcal{N}$-quantifier and atoms of this author's work. Perhaps most of the apparent differences between the author's work and others and that of Miller and others is down to differences in emphasis: in the author's case on semantics and staying close to the spirit of first-order logic; in Miller's case on implementation and staying close to the spirit of higher-order logic.

— a non-functional notion of abstraction with α-equivalence based on permutations, and
— a distinctive first-order semantics based on Fraenkel-Mostowski set theory and nominal sets (Gabbay 2011).

We take full mathematical advantage of these features in this paper.

Nominal terms have two levels of variable: *atoms* (level 1 or object-level variables) and *unknowns* (level 2 or meta-level variables[2]). Nominal terms and their unification were introduced in (Urban et al. 2003, 2004). They were designed to formally express questions couched in the informal meta-level of mathematical discourse (the rigorous but informal language of mathematics papers like this one), such as:

"What values of t make '$\lambda x.\lambda y.t$' equal to '$\lambda y.\lambda x.t$'?"

Here we see the same two levels of variable, with x and y having a different level from the 'meta-variable' t. This is modelled almost symbol-for-symbol as the *nominal terms unification* problem

"What values of X make '$\lambda[a]\lambda[b]X$' equal to '$\lambda[b]\lambda[a]X$'?"

Here, λ is just a term-former and $[a]$- and $[b]$- denote the atoms-abstraction from (Gabbay and Pitts 2001). Note that $[a]$- is not a functional abstraction. This is a 'nominal' answer to the issues with function spaces highlighted in the quote above.

The applications of nominal terms go beyond unification. Here, for example, is a formal specification of η-equivalence "if x is fresh for t then $\lambda x.(tx) = t$":

$$a\#X \Rightarrow \lambda[a](\mathsf{app}(X,a)) = X$$

Here $a\#X$ is a *freshness side-condition* which formally models 'if x is not free in t'.

Thus, in a series of logical systems on nominal rewriting, nominal logic programming, nominal algebra, and permissive-nominal logic (Fernández and Gabbay 2007; Cheney and Urban 2008; Gabbay and Mathijssen 2009; Dowek and Gabbay 2010), the author and others have investigated nominal terms' application to rewriting, logic programming, equational specification, and first-order specification. These have been used by the author and others to axiomatise first-order logic, λ-calculus, and arithmetic (Gabbay and Mathijssen 2008b, 2010; Dowek and Gabbay 2010). Cheney and Urban have implemented a logic-programming system αProlog (Cheney and Urban 2008) which is based on nominal terms.

1.3. *Semantics for nominal terms*

There is however no *nominal* semantics for the unknowns in nominal terms; so far, we only have *functional* semantics for them. That question motivates this paper.

What do we mean by this? As already mentioned, nominal terms have a notion of *meta*-variable X, which we call *unknowns*. They also have a notion of atoms-abstraction

[2] I deprecate calling level 2 variables 'meta-level', because when they are modelled in a formal syntax they cease to be meta-level and instead become a *model* of the meta-level. A *thing* is not the same as our mathematical *model* of that thing.

$[a]$- whose semantics is not functional; abstraction is modelled using the Gabbay-Pitts atoms-abstraction in nominal sets, from the author's thesis and subsequent work (Gabbay and Pitts 1999; Gabbay 2001; Gabbay and Pitts 2001).

However, the only interpretation of unknowns in nominal terms is based on functional abstraction. This is usually phrased as a *valuation* mapping unknowns to denotations, but note that a valuation is just one giant simultaneous functional abstraction over all unknowns. See for example the denotation of nominal algebra in (Gabbay and Mathijssen 2009, Definition 4.14).

This is a mathematically reasonable and correct denotation but it is also unsatisfactory that this should be the only one in our toolbox—for a number of reasons:

- Part of nominal techniques is that we get inductive datatypes of syntax-with-binding called *nominal abstract syntax* (Gabbay and Pitts 2001). Because nominal unknowns do not have a nominal semantics, atoms-abstraction in nominal term syntax cannot be directly interpreted as atoms-binding in the style of nominal abstract syntax.[3]
- Thus, nominal inductive reasoning principles cannot be applied to nominal terms in a way corresponding to the syntax-with-binding explored in (Gabbay and Pitts 2001).
- There is no nominal style theory of binding for variables in nominal terms. That is, nominal terms have a notion of variable X, but no accompanying notion of $\forall X$ or λX.

In short, and in spite of their name, nominal terms are closer to first-order terms than to nominal terms: it is hard to define datatypes of nominal term syntax-with-binding, and their variables have denotational semantics using valuations.

So we are led to the following question:

What 'nominal' meaning can be given to the meta-variables of nominal terms, if any?

In this paper we will develop a semantic theory which explains nominal terms purely in terms of an elaboration of the nominal semantics from (Gabbay and Pitts 2001). We call this new semantics *two-level nominal sets*. There is no need to appeal to functions and higher-orders to explain the variables in nominal terms. As a nice corollary of this semantics, we see how to extend nominal terms with binding for its variables; that is, following the notation and terminology of (Urban et al. 2004), we will extend nominal term syntax with unknowns-abstraction $[X]r$ and interpret that in two-level nominal sets.

This work is based on two ideas:

- (Meta-)Variables are modelled as well-orderings on infinite sets of names / atoms / urelemente. In this paper we call these *level 2 atoms*.

[3] The literature has been strangely quiet on this point. The datatype of nominal terms of Definition 2.3 in (Urban et al. 2004) is what we would have called a first-order name-carrying datatype of abstract syntax; that is, an 'ordinary' datatype, not up to α-conversion. This relation has to be defined 'by hand'; see Figure 2 of (Urban et al. 2004).

— Binding is modelled as equivalence classes of permutations of well-orderings.

That is, by the mathematical story told in this paper, level 2 atoms are well-orderings on level 1 atoms, and α-renaming of level 2 atoms is based on reordering those well-orderings. It is not obvious why this should work, but the mathematics to follow will show that it does. We give some intuitions as to why, in Remark 2.10.

The reader familiar with nominal terms can recover the notion of a *moderated unknown* or *variable with suspended permutation* $\pi \cdot X$ by considering a level 2 atom as a pair (π, X) where π is a finite level 1 permutation and X (or $\bar{a}$, in the notation of this paper) is a fixed but arbitrary choice of representative. We make this formal towards the end of the paper, in Definition 6.11.

1.4. *Summary*

There are many multi-level syntaxes in the literature. This is because it is useful to separate 'meta-level' from 'object-level', or to capture ideas of context, modularity, incompleteness, and incrementality.

Nominal sets (or Fraenkel-Mostowski sets, to use an earlier and scarier name) are a sets-based denotation with a marked first-order flavour.[4] Nominal terms are a syntax with two levels of variable. Nominal terms have well-understood denotations in nominal sets; notably those developed in previous work by the author and others on nominal algebra and permissive-nominal logic (Gabbay and Mathijssen 2009; Dowek and Gabbay 2010). However, these denotations are based on valuations and so are functional. Since one motivation for nominal techniques is "names and binding without functions" we are led to ask whether an alternative, less functional and more nominal, answer exists; this question also has practical repercussions if we want to reason inductively on nominal-terms-up-to-binding or extend nominal terms with abstraction for level 2 variables.

In this paper we will construct a new semantic answer by generalising nominal sets, and verify its soundness with respect to nominal terms generalised to include level 2 abstraction.

Note that this paper concentrates on semantics. The reader interested in unpacking the applications of nominal terms can find plenty of material elsewhere: notably the nominal rewriting and algebraic frameworks (Fernández and Gabbay 2007; Gabbay and Mathijssen 2009), αProlog (Cheney and Urban 2008), and axiomatisations using nominal terms and proofs of correctness of first-order logic, λ-calculus, and arithmetic (Gabbay and Mathijssen 2008b, 2010; Dowek and Gabbay 2010). A first application to incremental program construction is (Gabbay and Mathijssen 2008b), with more in preparation.

[4] The first 'nominal' denotation as considered in (Gabbay and Pitts 1999) was models of Fraenkel-Mostowski set theory. Nominal sets are equivariant Fraenkel-Mostowski sets; so nominal sets are a special case of Fraenkel-Mostowski sets. However, elements of nominal sets need not be equivariant and in that sense we still get back to Fraenkel-Mostowski sets; indeed since the universe of all Fraenkel-Mostowski sets is itself equivariant, in a certain sense they are a special case of a nominal set. It does not matter for this paper, but this brief footnote might be of benefit to an interested reader.

2. Two-level nominal sets

2.1. *Atoms and permutatations*

We start as usual in nominal techniques by postulating a set of atoms $\mathbb{A}$. This is Definition 2.1.

Unlike some previous work, $\mathbb{A}$ is split into two countably infinite halves, $\mathbb{A}^<$ and $\mathbb{A}^>$. This makes the development *permissive*, following terminology from (Dowek et al. 2010): Our treatment of atoms is based not on finite and cofinite sets of atoms like (Gabbay and Pitts 2001), but on *permission sets* which are sets of atoms differing finitely from $\mathbb{A}^<$.[5] This is Definition 2.3.

The next idea is a notion of *level 2 atom*. A level 2 atom is an *ordering on a list of (level 1) atoms*. This is Definition 2.5. We also introduce the notion of the orbit of a level 2 atom.

Finally, we consider a fundamental property of the interaction between level 2 atoms and permutations. This is Proposition 2.9.

Definition 2.1. Fix two disjoint countably infinite sets of **(level 1) atoms** $\mathbb{A}^<$ and $\mathbb{A}^>$ and write $\mathbb{A} = \mathbb{A}^< \cup \mathbb{A}^>$.

$a, b, c, \ldots$ will range over distinct atoms; we call this the **permutative** convention.

Definition 2.2. A **level 1 permutation** is a bijection π on $\mathbb{A}$ such that
$$nontriv(\pi) = \{a \mid \pi(a) \neq a\} \quad \text{is finite.}$$
π will range over level 1 permutations.

Definition 2.3. A **permission set** S has the form
$$S = (\mathbb{A}^< \backslash A) \cup A' \quad \text{where} \quad A \subseteq \mathbb{A}^< \text{ and } A' \subseteq \mathbb{A}^> \text{ are finite.}$$
S, T, U will range over permission sets.

Notation 2.4. i will range over strictly positive natural numbers $1, 2, 3, \ldots$

Definition 2.5. A **level 2 atom** $\bar{a}$ is an ω-tuple (a stream, or infinite list) $(a_i)_i$ of distinct level 1 atoms such that
$$atoms(\bar{a}) = \{a_i \mid i \in \omega\} \quad \text{is a permission set.}$$
Write $\bar{\mathbb{A}}$ for the set of level 2 atoms.

Definition 2.6. Define $\pi{\cdot}\bar{a}$ and $orb(\bar{a})$ by:
$$\pi{\cdot}\bar{a} = (\pi(a_i))_i \qquad \in \bar{\mathbb{A}}$$
$$orb(\bar{a}) = \{\pi{\cdot}\bar{a} \mid \text{all } \pi\} \subseteq \bar{\mathbb{A}}$$

Remark 2.7. $orb(\bar{a})$ in Definition 2.6 is particularly important.

[5] One nice way of looking at the difference between permissive nominal terms from (Dowek et al. 2010) and the nominal terms from (Urban et al. 2004), is that in nominal terms we are particularly interested in sets of atoms that differ finitely from $\varnothing$, whereas in permissive nominal terms we are particularly interested in sets of atoms that differ finitely from $\mathbb{A}^<$.

$orb(\bar{a})$ is the orbit of $\bar{a}$ under the action of level 1 permutations π such that π permutes only atoms in $atoms(\bar{a})$. Thus $orb(\bar{a})$ is the equivalence class of level 2 atoms obtained from $\bar{a}$ by re-ordering finitely but unboundedly many of the atoms in $\bar{a}$.

Note that $atoms(\bar{a}) = atoms(\bar{b})$ does not imply $orb(\bar{a}) = orb(\bar{b})$. This is because two infinite lists may mention exactly the same atoms, but in orders which differ infinitely from each other.

We will see these same ideas when we define level 2 atoms-abstraction $[\bar{a}]x$ in Definition 4.2.

Notation 2.8. Henceforth $\bar{a}, \bar{b}, \bar{a}', \bar{c}, \ldots$ will range over level 2 atoms *in distinct orbits*. That is, '$\bar{a}$ and $\bar{b}$' means "any two $\bar{a} \in \bar{\mathbb{A}}$ and $\bar{b} \in \bar{\mathbb{A}}$ such that $orb(\bar{a}) \neq orb(\bar{b})$".

Proposition 2.9. *$\pi{\cdot}\bar{a} = \pi'{\cdot}\bar{a}$ if and only if $\pi(a) = \pi'(a)$ for every $a \in atoms(\bar{a})$.*

Proof. By an easy calculation. □

Remark 2.10. ω-tuples have infinitely many level 1 atoms and satisfy Proposition 2.9. They are the *simplest* non-trivial structure with these two properties, that this author can think of. ω-tuples can also be swapped: if the atoms in $\bar{b}$ and $\bar{a}$ are equal then intuitively $(\bar{b}\ \bar{a})$ swaps the ith atom of $\bar{a}$ 'pointwise' for the ith atom of $\bar{b}$.

There is a close connection between Proposition 2.9 and an axiomatic property of nominal terms (see e.g. Lemma 2.8 of (Urban et al. 2004)). In the terminology used in (Tzevelekos 2007), Proposition 2.9 states that level 2 atoms are *strongly supported* at level 1. So in that terminology, ω-tuples are the simplest strongly-supported elements with infinite support.

Other structures with these properties are possible (for instance, binary trees with ordered daughters), and we make no claim that Definition 2.5 is unique. We discuss some possible alternatives in the Conclusions. Definition 2.5 seems *canonical* in the sense that it is *minimal* amongst possible definitions in a sense we do not make formal.

Later on in Definition 2.21 we will define level 2 swapping $(\bar{b}\ \bar{a})$. As mentioned, one further benefit of using ω-tuples is that they make the level 2 swapping action easy to imagine: intuitively $(\bar{b}\ \bar{a})$ swaps the ith atom of $\bar{a}$ 'pointwise' for the ith atom of $\bar{b}$. Having given this intuition we should qualify it: Definition 2.21 is more subtle than that; it may be that $(\bar{b}\ \bar{a})(\bar{c}) = \bar{c}$ even if atoms in $\bar{b}$ and $\bar{a}$ occur in $\bar{c}$, depending on whether $\bar{c}$ differs *finitely* from $\bar{a}$ or $\bar{b}$.

The headline is this: ω-lists are a concise mathematical structure with infinitely many atoms, strong support, and determining a level 2 swapping action.

2.2. *Sets with a two-level permutation action*

Our ultimate goal is to define a notion of two-level nominal set (Definition 2.33, for the impatient) but before we do this it is useful to consider a more primitive notion of a set with permutation actions for level 1 and for level 2.

In this short subsection we introduce the idea of a level 2 permutation (Definition 2.11) and a set with a two-level permutation action (Definition 2.15).

As the name suggests, a 'level 2 permutation' is like a 'level 1 permutation', only it permutes level 2 atoms instead of level 1 atoms.

However, whereas level 1 atoms are atomic elements (urelemente), level 2 atoms are not atomic at all. Level 2 atoms are lists, and as such they have internal structure.[6]

Nominal techniques from (Gabbay and Pitts 2001) are based on the idea of *atoms being atomic*. It is not obvious that $\bar{a}$ should display enough of the behaviour which makes atoms a useful, to be similarly useful—but it does, just.

Definition 2.11. A **level 2 permutation** is a bijection $\bar{\pi}$ on $\bar{\mathbb{A}}$ such that:

1. $atoms(\bar{\pi}(\bar{a})) = atoms(\bar{a})$ for all $\bar{a}$.
2. $non\bar{triv}(\bar{\pi})$ is finite.
3. $\bar{\pi}(\pi{\cdot}\bar{a}) = \pi{\cdot}(\bar{\pi}(\bar{a}))$ for all π and all $\bar{a}$.

Write

$$non\bar{triv}(\bar{\pi}) = \{orb(\bar{a}) \mid \bar{\pi}(\bar{a}) \neq \bar{a}\} \subseteq orb(\bar{\mathbb{A}}).$$

$\bar{\pi}$ will range over level 2 permutations.

Remark 2.12. $\bar{\pi}$ rearranges the *order* of the atoms in $\bar{a}$, not necessarily finitely. Thus $atoms(\bar{\pi}(\bar{a})) = atoms(\bar{a})$ but $orb(\bar{\pi}(\bar{a}))$ is not necessarily equal to $orb(\bar{a})$.

$\bar{\pi}$ must also be finitely-supported in that it only affects finitely many orbits of level 2 atoms.

Finally, $\bar{\pi}$ must commute with the level 1 permutation action. In the terminology of Definition 3.14, the level 2 action must be level 1 equivariant; see Lemma 3.16.

Definition 2.13. As is standard, we write -^{-1} for inverse, $\circ$ for functional composition, and id for the identity. Thus for example $(f \circ g)(x) = f(g(x))$, and $id(x) = x$.

Lemma 2.14. *Level 1 permutations form a group with $\circ$, -^{-1}, and id. Level 2 permutations form a group similarly.*

Definition 2.15. A **set with a two-level permutation action** X is a triple $(|\mathsf{X}|, \cdot_1, \cdot_2)$ of

— an **underlying set** $|\mathsf{X}|$,
— a **level 1 permutation action** $\cdot_1 : \mathbb{P} \times |\mathsf{X}| \to |\mathsf{X}|$, we write it infix $\pi{\cdot}x$, and
— a **level 2 permutation action** $\cdot_2 : \bar{\mathbb{P}} \times |\mathsf{X}| \to |\mathsf{X}|$, we write it infix $\bar{\pi}{\cdot}x$

such that $\cdot_1$ and $\cdot_2$ are group actions of $\mathbb{P}$ and $\bar{\mathbb{P}}$ respectively on $|\mathsf{X}|$.
We will normally write $\pi{\cdot_1}x$ as $\pi{\cdot}x$ and $\bar{\pi}{\cdot_2}x$ as $\bar{\pi}{\cdot}x$.

Definition 2.15 states that $|\mathsf{X}|$ is acted on by two different groups, with no further specification of their interaction except that, as observed in Remark 2.12, level 2 and level 1 permutations will by Definition 2.11 commute.

Example 2.16. Here are examples of sets with a two-level permutation action:

[6] If a is an atom then $\bar{a}$ is a 'molecule' or 'polymer' of level 1 atoms. In spite of its name, a level 2 atom is not an atomic structure. What *is* atomic, in a certain sense, is the order of the atoms within $\bar{a}$.

1. $\mathbb{A}$ with $\pi{\cdot}a = \pi(a)$ and $\bar{\pi}{\cdot}a = a$.
2. $\bar{\mathbb{A}}$ with $\pi{\cdot}\bar{a} = (\pi(a_i))_i$ and $\bar{\pi}{\cdot}\bar{a} = \bar{a}$.
3. The set of permission sets with $\pi{\cdot}S = \{\pi(a) \mid a \in S\}$ and $\bar{\pi}{\cdot}S = S$.
4. The set of all sets of atoms *powerset*$(\mathbb{A})$ with the *pointwise* action $\pi{\cdot}A = \{\pi(a) \mid a \in A\}$ and $\bar{\pi}{\cdot}A = A$.
5. The set of all sets of level 2 atoms *powerset*$(\bar{\mathbb{A}})$ with the pointwise action $\pi{\cdot}B = \{\pi{\cdot}\bar{a} \mid \bar{a} \in B\}$ and $\bar{\pi}{\cdot}B = \{\bar{\pi}(\bar{b}) \mid \bar{b} \in B\}$.
6. $orb(\bar{\mathbb{A}})$ with $\pi{\cdot}orb(\bar{a}) = orb(\bar{a})$ and $\bar{\pi}{\cdot}orb(\bar{a}) = orb(\bar{\pi}(\bar{a}))$.
7. Suppose X and Y are sets with a two-level permutation action. Then $\mathsf{X} \to \mathsf{Y}$ with underlying set $|\mathsf{X}| \to |\mathsf{Y}|$ (functions on the underlying sets) with the **conjugation action**

$$(\pi{\cdot}f)x = \pi{\cdot}(f(\pi^{-1}{\cdot}x)) \quad \text{and} \quad (\bar{\pi}{\cdot}f)x = \bar{\pi}{\cdot}(f(\bar{\pi}^{-1}{\cdot}x))$$

is a set with a two-level permutation action.

Remark 2.17. We take a moment to check in detail that the action $\bar{\pi}{\cdot}orb(\bar{a}) = orb(\bar{\pi}(\bar{a}))$ of part 6 of Example 2.16, is well-defined. Suppose $\pi{\cdot}atoms(\bar{a}) = atoms(\bar{a})$ so that $orb(\bar{a}) = orb(\pi{\cdot}\bar{a})$. We need to show that $orb(\bar{\pi}(\pi{\cdot}\bar{a})) = orb(\bar{\pi}(\bar{a}))$.

By assumption $\bar{\pi}(\pi{\cdot}\bar{a}) = \pi{\cdot}\bar{\pi}(\bar{a})$. Also by assumption $atoms(\bar{\pi}(\bar{a})) = atoms(\bar{a})$, so that $\pi{\cdot}atoms(\bar{\pi}(\bar{a})) = atoms(\bar{\pi}(\bar{a}))$. It follows that $orb(\pi{\cdot}\bar{\pi}(\bar{a})) = orb(\bar{\pi}(\bar{a}))$.

Remark 2.18. Another way to characterise the conjugation action from part 7 of Example 2.16 is that

$$\pi{\cdot}f(x) = (\pi{\cdot}f)(\pi{\cdot}x) \quad \text{and} \quad \bar{\pi}{\cdot}f(x) = (\bar{\pi}{\cdot}f)(\bar{\pi}{\cdot}x).$$

Remark 2.19. The condition $atoms(\bar{\pi}(\bar{a})) = atoms(\bar{a})$ in Definition 2.11 is necessary. Suppose we drop it. Now consider $\bar{\pi}$ such that $atoms(\bar{\pi}(\bar{a})) \neq atoms(\bar{a})$. Suppose $\pi{\cdot}\bar{a} = \pi'{\cdot}\bar{a}$, so that by Proposition 2.9 π and π' agree on atoms in $atoms(\bar{a})$. Now it does not follow that $\pi{\cdot}\bar{\pi}(\bar{a}) = \pi'{\cdot}\bar{\pi}(\bar{a})$.

The condition $\pi{\cdot}\bar{\pi}(\bar{a}) = \bar{\pi}(\pi{\cdot}\bar{a})$ is also necessary. It is in the terminology of Subsection 3.2 an *equivariance* condition; see also Lemma 3.16. To examine one more example of where it is used, see Lemma 4.20.

2.3. *Level 2 swappings*

Intuitively a swapping $(\bar{a}\ \bar{b})$ maps $\bar{a}$ to $\bar{b}$ and vice-versa. But the definition of a swapping needs to be a little more elaborate, to account for the coherence condition in Definition 2.11 that $(\bar{a}\ \bar{b}){\cdot}\pi{\cdot}x = \pi{\cdot}(\bar{a}\ \bar{b}){\cdot}x$. So we have to be just a little careful.

We introduce level 1 swappings and level 2 swappings in Definitions 2.20 and 2.21. In Lemma 2.25 and Proposition 2.26 we check that relevant parts of the theory of permutations remain true in the more elaborate level 2 case.

Definition 2.20 is standard from (Gabbay and Pitts 2001):

Definition 2.20. Suppose a and b are level 1 atoms. Define a level 1 **swapping** $(b\ a)$ by:

$$\begin{array}{l}(b\ a)(a) = b\\(b\ a)(b) = a\\(b\ a)(c) = c\end{array}$$

Definition 2.21 is new:

Definition 2.21. Suppose $\bar{a}$ and $\bar{b}$ are level 2 atoms (and $orb(\bar{a}) \neq orb(\bar{b})$). Suppose $atoms(\bar{a}) = atoms(\bar{b})$.

Suppose also that π is a level 1 permutation and $atoms(\pi{\cdot}\bar{a}) = atoms(\bar{a})$.

Define level 2 **swappings** $(\bar{b}\ \bar{a})$ and $(\pi{\cdot}\bar{a}\ \bar{a})$ by:

$$\begin{array}{ll}(\bar{b}\ \bar{a})(\pi'{\cdot}\bar{a}) = \pi'{\cdot}\bar{b} & (\pi{\cdot}\bar{a}\ \bar{a})(\pi'{\cdot}\bar{a}) = (\pi' \circ \pi){\cdot}\bar{a}\\(\bar{b}\ \bar{a})(\pi'{\cdot}\bar{b}) = \pi'{\cdot}\bar{a} & (\pi{\cdot}\bar{a}\ \bar{a})(\bar{b}) = \bar{b}\\(\bar{b}\ \bar{a})(\bar{c}) = \bar{c} &\end{array}$$

Remark 2.22. Recall from Notation 2.8 the permutative convention that $\bar{a}$, $\bar{b}$, and $\bar{c}$ range over level 2 atoms in distinct orbits.

The three cases in the definition of the action of $(\bar{b}\ \bar{a})$ in Definition 2.21 come from the fact that for any level 2 atom z, exactly on of the following three must hold: $orb(z) = orb(\bar{a})$, $orb(z) = orb(\bar{b})$, or $orb(z) = orb(\bar{c})$ for some $\bar{c}$ with $orb(\bar{c}) \notin \{orb(\bar{a}), orb(\bar{b})\}$.

Similarly for the definition of the action of $(\pi{\cdot}\bar{a}\ \bar{a})$.

Remark 2.23. The reader familiar with nominal techniques with expect that $(\bar{b}\ \bar{a}) = (\bar{a}\ \bar{b})$, since $(b\ a) = (a\ b)$. This is correct.

However, $(\pi{\cdot}\bar{a}\ \bar{a}) \neq (\bar{a}\ \pi{\cdot}\bar{a})$ in general. To see why, choose any $\bar{a}$ and suppose $a, b, c \in atoms(\bar{a})$. Let $\pi{=}(b\ c) \circ (c\ a)$ (so $\pi(a){=}b$, $\pi(b){=}c$, and $\pi(c){=}a$). Then $(\pi{\cdot}\bar{a}\ \bar{a})(\bar{a}) = \pi{\cdot}\bar{a}$ and $(\bar{a}\ \pi{\cdot}\bar{a})(\bar{a}) = \pi^{-1}{\cdot}\bar{a} \neq \pi{\cdot}\bar{a}$.

It is true that $(\pi{\cdot}\bar{a}\ \bar{a}) = (\bar{a}\ \pi^{-1}{\cdot}\bar{a})$ always: this is a special case of Lemma 2.25.

Lemma 2.24. *Definition 2.21 is well-defined, defines level 2 permutations, and* $\overline{non}triv((\bar{b}\ \bar{a})) = \{orb(\bar{b}), orb(\bar{a})\}$ *and* $\overline{non}triv((\pi{\cdot}\bar{a}\ \bar{a})) = \{orb(\bar{a})\}$.

Proof. The only slightly non-trivial part is to check is that $(\pi{\cdot}\bar{a}\ \bar{a}){\cdot}\pi'{\cdot}\bar{a} = \pi'{\cdot}(\pi{\cdot}\bar{a}\ \bar{a}){\cdot}\bar{a}$. This is not hard. □

Lemma 2.25. *Suppose $\bar{a}$ and $\bar{b}$ are level 2 atoms (and by convention $orb(\bar{a}) \neq orb(\bar{b})$). Suppose $atoms(\bar{a}) = atoms(\bar{b})$. Suppose π is a level 1 permutation. Suppose π' is a level 1 permutation such that $\pi'{\cdot}atoms(\bar{a}) = atoms(\bar{a})$. Then*

$$(\pi{\cdot}\bar{b}\ \pi{\cdot}\bar{a}) = (\bar{b}\ \bar{a}) \quad \textit{and} \quad (\pi{\cdot}\pi'{\cdot}\bar{a}\ \pi{\cdot}\bar{a}) = (\pi'{\cdot}\bar{a}\ \bar{a}).$$

Proof. By routine calculations unpacking Definition 2.21. □

Proposition 2.26. *The set of all level 2 permutations is generated as a group by the level 2 swappings.*

Proof. This is not immediately obvious because $\{\bar{a} \mid \bar{\pi}(\bar{a}) \neq \bar{a}\}$ is not in general finite (the generators of a group should generate that group *finitely*). However, by assumption $\bar{nontriv}(\bar{\pi})$ *is* finite, and we can induct on its size.

Suppose $\bar{\pi}$ is a level 2 permutation. There are three cases:

- $\bar{\pi} = id$. There is nothing to prove.
- There exists some $\bar{a}$ such that $orb(\bar{\pi}(\bar{a})) \neq orb(\bar{a})$. Write $\bar{b} = \bar{\pi}(\bar{a})$. It is a fact, proved much as Lemma 2.25, that $\bar{nontriv}(\bar{\pi} \circ (\bar{b}\ \bar{a})) = \bar{nontriv}(\bar{\pi}) \backslash \{orb(\bar{b})\}$. By inductive hypothesis $\bar{\pi} \circ (\bar{b}\ \bar{a})$ is generated by swappings. The result follows.
- There exists some $\bar{a}$ such that $orb(\bar{\pi}(\bar{a})) = orb(\bar{a})$ and $\bar{\pi}(\bar{a}) = \pi{\cdot}\bar{a} \neq \bar{a}$. It is a fact that $\bar{nontriv}(\bar{\pi} \circ (\bar{a}\ \pi{\cdot}\bar{a})) = \bar{nontriv}(\bar{\pi}) \backslash \{orb(\bar{a})\}$. By inductive hypothesis $\bar{\pi} \circ (\bar{a}\ \pi^{-1}{\cdot}\bar{a})$ is generated by swappings. The result follows. □

2.4. *Two-level nominal sets*

We are now ready to extend the ideas in (Gabbay and Pitts 2001) to develop a notion of two-level nominal set. In the terminology we are about to develop, a two-level nominal set is a set with a two-level permutation action that has small support at levels 1 and 2.

This is what the reader familiar with nominal techniques would expect, except that our notion of support at level 2 has to be based on *orbits* of level 2 atoms (under finitely-supported permutations π). Thus in Definition 2.28 $fix(\bar{\pi})$ is a set of $orb(\bar{a})$ and not a set of $\bar{a}$.[7]

Remark 2.27. The reader might at this point begin to question our design decision to let level 2 atoms be ω-tuples of atoms. Since orbits $orb(\bar{a})$ are used in the coming definitions so much, why do we not take level 2 atoms to be what here we write $orb(\bar{a})$ instead of $\bar{a}$? The answer is this: suppose $b, a \in atoms(\bar{a})$; we do not want $(b\ a){\cdot}\bar{a}$ to be equal to $\bar{a}$ and it is a fact that $orb((b\ a){\cdot}\bar{a}) = orb(\bar{a})$.

Put another way, Proposition 2.9 would fail.

More on this in Remark 4.19, including an intuition for why $(b\ a){\cdot}\bar{a} \neq \bar{a}$ and Proposiiton 2.9 are so desirable.

Definition 2.28. Suppose $A \subseteq \mathbb{A}$ and $B \subseteq orb(\bar{\mathbb{A}})$. Define $fix(A)$ and $\bar{fix}(B)$ by:

$$\begin{array}{ll} fix(A) = \{\pi \mid \forall a \in A.\pi(a) = a\} & \subseteq \mathbb{P} \\ \bar{fix}(B) = \{\bar{\pi} \mid \forall o \in B.\forall \bar{b} \in o.\bar{\pi}(\bar{b}) = \bar{b}\} & \subseteq \bar{\mathbb{P}} \end{array}$$

Remark 2.29. The 'o' in the definition of $\bar{fix}(B)$ in Definition 2.28 is there because B is a set of *orbits* under the action of level 1 permutations π. Since $\bar{\pi}(\pi{\cdot}\bar{b}) = \pi{\cdot}(\bar{\pi}(\bar{b}))$ is assumed in Definition 2.11, the value of $\bar{\pi}$ on *one* $\bar{b} \in o$ determines that on *every* $\bar{b} \in o$. Thus another way to define $\bar{fix}(B)$ would use $\exists \bar{b} \in o$ instead of $\forall \bar{b} \in o$.

Perhaps simplest of all, using the permutation action in part 6 of Example 2.16, is to define $\bar{fix}(B) = \{\bar{\pi} \mid \forall o \in B.\bar{\pi}{\cdot}o = o\}$. These definitions are all equivalent.

[7] It could almost never be finite otherwise.

Definition 2.30. Suppose X is a set with a two-level permutation action.

Say that $A \subseteq \mathbb{A}$ **1-supports**, and say that $B \subseteq \mathit{orb}(\bar{\mathbb{A}})$ **2-supports** $x \in |\mathsf{X}|$ when
$$\forall \pi.\ \pi \in \mathit{fix}(A) \Rightarrow \pi{\cdot}x{=}x \quad \text{and}$$
$$\forall \bar{\pi}.\ \bar{\pi} \in \bar{\mathit{fix}}(B) \Rightarrow \bar{\pi}{\cdot}x{=}x$$
respectively.

Definition 2.31. Call a set $A \subseteq \mathbb{A}$ **small** when $A \subseteq S$ for some permission set S. Call a set $B \subseteq \mathit{orb}(\bar{\mathbb{A}})$ **small** when for every permission set S, the set $\{\mathit{orb}(\bar{a}) \in B \mid \mathit{supp}(\bar{a}) = S\}$ is finite.

Remark 2.32. Another way to characterise 'small' from Definition 2.31 brings out a symmetry between the two levels: A is small when $A \backslash \mathbb{A}^{<}$ is finite; B is small when $B \backslash \varnothing$ is finite. More on the design of Definition 2.31 in Remark 2.35.

Definition 2.33. Call a set with a two-level permutation action X a **two-level nominal set** when:

- For every $x \in |\mathsf{X}|$ there exist small $A \subseteq \mathbb{A}$ and $B \subseteq \mathit{orb}(\bar{\mathbb{A}})$ that support x.
- $\bar{\pi}{\cdot}(\pi{\cdot}x) = \pi{\cdot}(\bar{\pi}{\cdot}x)$.

Henceforth X, Y, and Z will range over two-level nominal sets.

Example 2.34. Recall the example sets with a two-level permutation action from Example 2.16. $\mathbb{A}$, $\bar{\mathbb{A}}$, and the set of permission sets are two-level nominal sets. $\mathit{orb}(\bar{\mathbb{A}})$ is a two-level nominal set.

The set of sets of atoms $\mathit{powerset}(\mathbb{A})$ (number 4 in Example 2.16) is not a two-level nominal set, because not all sets of atoms have small support to level 1. Neither are $\mathit{powerset}(\bar{\mathbb{A}})$ and $|\mathsf{X} \to \mathsf{Y}|$, for similar reasons.

Remark 2.35. A purpose of *small* support is to guarantee an infinite supply of fresh names; it is a feature of names that 'we can always find a fresh one' and this is also a technical requirement of the mathematics to follow, e.g. the construction of atoms-abstraction in Section 4.

There is design freedom in what we take 'small' to mean. Definition 2.33 does exactly what is convenient:

- At level 2 we take small to be finite. This is consistent with (Gabbay and Pitts 2001).
- At level 1 we cannot do this, because we want level 2 atoms $\bar{a}$ to have small support. So we use $\mathbb{A}^{<}$ instead.
  As remarked, we can view small at level 1 as 'small'='only finitely larger than $\mathbb{A}^{<}$'. Small sets at level 1 are closed under the axioms of a support ideal (Cheney 2006) (permission sets are not, because they do not include finite sets). In view of the fact that level 2 atoms are well-orderings of small sets of level 1 atoms, we can also view small at level 1 as being 'small'='well-orderable'. This is consistent with (Gabbay 2002, 2007).

Other design choices exist. For instance, we could take uncountably many atoms and take 'small'='countable' at both levels.

3. Support and equivariance; functions and the category NOM2

It is well-known from nominal techniques that a 'nominal' element x has a *supporting set* of atoms $supp(x)$, which is the least set of atoms such that if $\pi(a) = \pi'(a)$ for all $a \in supp(x)$ then $\pi{\cdot}x = \pi'{\cdot}x$. Permutations are to α-renaming, as support is to 'free variables in'.

In Subsection 3.1 we extend this story to the level 2 atoms. Note that $s\bar{u}pp(x)$ is a set of *orbits*; e.g. $s\bar{u}pp(\bar{a}) = \{orb(\bar{a})\}$. We emphasise here that $orb(\bar{a})$ (Definition 2.6) is not equal to $atoms(\bar{a})$ (Definition 2.5). The notion of level 2 support is actually quite subtle—but it seems to be what is required to make important 'nominal' results valid for the level 2 case, such as Theorem 3.9 and the property of having small support.

In Subsection 3.2 we explore notions of *equivariance* for elements and functions. Equivariance is important, because an equivariant element is in some sense 'global' or 'generic'. Equivariance manifests itself in several equivalent ways: $supp(x) = \varnothing$, $\pi{\cdot}x = x$ for all π, and (if x is a function) $\pi{\cdot}(x(y)) = x(\pi{\cdot}y)$, and similarly at level 2. We state and prove these properties. We then discuss some useful examples of equivariant elements in Remark 3.17 and the subsequent results. In Corollary 3.10 we exploit one of these examples to give one way of detecting the support of an element; it will be useful later.

3.1. *Support at levels 1 and 2*

Definition 3.1. Call a permutation π / $\bar{\pi}$ **self-inverse** when $\pi = \pi^{-1}$ / $\bar{\pi} = \bar{\pi}^{-1}$.

Lemma 3.2. *1. If $A', A \subseteq \mathbb{A}$ are small and 1-support $x \in |\mathsf{X}|$ then so does $A' \cap A$.*
2. If $B', B \subseteq orb(\bar{\mathbb{A}})$ are small and 2-support $x \in |\mathsf{X}|$ then so does $B' \cap B$.

Proof. We prove only the second part; the first part is similar.

Suppose B' and B are small and 2-support $x \in |\mathsf{X}|$. Suppose $\bar{\pi} \in \bar{fix}(B \cap B')$. Now $B' \backslash B$ is finite and $orb(\bar{\mathbb{A}}) \backslash (B \cup B')$ is infinite, so we can find a self-inverse permutation $\bar{\pi}' \in \bar{fix}(B)$ such that $(\bar{\pi}'{\cdot}B') \cap B' = B \cap B'$. It can be verified that $\bar{\pi}' \circ \bar{\pi} \circ \bar{\pi}' \in \bar{fix}(B')$, so $(\bar{\pi}' \circ \bar{\pi} \circ \bar{\pi}'){\cdot}x = x$. It follows from Definition 2.28 that $\bar{\pi}{\cdot}x = x$, as required. □

Lemma 3.3. — *A supports x if and only if $\pi{\cdot}A$ supports $\pi{\cdot}x$.*
— *B supports x if and only if $\bar{\pi}{\cdot}B$ supports $\bar{\pi}{\cdot}x$.*

Proof. We consider only the second part; the first is exactly similar. Since permutations are invertible, it suffices to show that if B supports x then $\bar{\pi}{\cdot}B$ supports $\bar{\pi}{\cdot}x$.

Suppose B supports x and suppose $\bar{\pi}' \in \bar{fix}(\bar{\pi}{\cdot}B)$. It is a fact that $\bar{\pi}^{-1} \circ \bar{\pi}' \circ \bar{\pi} \in \bar{fix}(B)$. So $(\bar{\pi}^{-1} \circ \bar{\pi}' \circ \bar{\pi}){\cdot}x = x$. It follows that $\bar{\pi}'{\cdot}(\bar{\pi}{\cdot}x) = \bar{\pi}{\cdot}x$. □

Definition 3.4. Define $supp(x)$ and $s\bar{u}pp(x)$ by:

$$\begin{array}{l} supp(x) = \bigcap\{A \subseteq \mathbb{A} \mid A \text{ is small and } A \text{ 1-supports } x\} \\ s\bar{u}pp(x) = \bigcap\{B \subseteq orb(\bar{\mathbb{A}}) \mid B \text{ is small and } B \text{ 2-supports } x\} \end{array}$$

Notation 3.5. Following (Gabbay and Pitts 2001), we write $a\#x$ for '$a \notin supp(x)$'. Similarly we write $\bar{a}\#x$ for '$\bar{a} \notin s\bar{u}pp(x)$'.

Example 3.6. Recall the two-level permutation actions from Example 2.34. With these actions:

$$\begin{array}{lllll} \text{In } \mathbb{A}: & supp(a) = \{a\} & \text{and} & s\bar{u}pp(a) = \varnothing \\ \text{In } \bar{\mathbb{A}}: & supp(\bar{a}) = atoms(\bar{a}) & \text{and} & s\bar{u}pp(\bar{a}) = \{orb(\bar{a})\} \\ \text{In } orb(\bar{\mathbb{A}}): & supp(orb(\bar{a})) = \varnothing & \text{and} & s\bar{u}pp(orb(\bar{a})) = \{orb(\bar{a})\} \end{array}$$

Lemma 3.7. *$supp(\pi{\cdot}x) = \pi{\cdot}supp(x)$ and $s\bar{u}pp(\bar{\pi}{\cdot}x) = \bar{\pi}{\cdot}s\bar{u}pp(x)$.*

Proof. Routine from Lemma 3.3 and Definition 3.4. □

Definition 3.8. Define $diff(\pi, \pi')$ and $\bar{diff}(\bar{\pi}, \bar{\pi}')$ by:

$$\begin{array}{ll} diff(\pi, \pi') = \{a \mid \pi(a) \neq \pi'(a)\} & \subseteq \mathbb{A} \\ \bar{diff}(\bar{\pi}, \bar{\pi}') = \{orb(\bar{a}) \mid \bar{\pi}(\bar{a}) \neq \bar{\pi}'(\bar{a})\} & \subseteq orb(\bar{\mathbb{A}}) \end{array}$$

Theorem 3.9. *1. If $x \in |\mathsf{X}|$ has a small 1-supporting set A then $supp(x)$ is well-defined, is small, and is the unique least set of level 1 atoms that 1-supports x.*

2. *If $x \in |\mathsf{X}|$ has a small 2-supporting set B then $s\bar{u}pp(x)$ is well-defined, is small, and is the unique least set of orbits of level 2 atoms that 2-supports x.*

As a particular corollary,

3. *if $diff(\pi, \pi') \cap supp(x) = \varnothing$ then $\pi{\cdot}x = \pi'{\cdot}x$, and*
4. *if $\bar{diff}(\bar{\pi}, \bar{\pi}') \cap s\bar{u}pp(x) = \varnothing$ then $\bar{\pi}{\cdot}x = \bar{\pi}'{\cdot}x$.*

Proof. For part 1, we reason as follows:

— *$supp(x)$ is small (Definition 2.31).* By construction $supp(x) \subseteq A$ and A is small.
— *$supp(x)$ is well-defined.* By assumption A a small supporting set for x, exists.
— *$supp(x)$ supports x.* Assume $\pi \in fix(supp(x))$. It suffices to show that $\pi{\cdot}x = x$. $nontriv(\pi)$ is finite, so write $nontriv(\pi) = \{a_1, \ldots, a_n\}$. By assumption $nontriv(\pi) \cap \bigcap\{A \text{ small and 1-supports } x\} = \varnothing$, so for every a_i there exists a small A_i that 1-supports x and such that $a_i \notin A_i$. By Lemma 3.2 $\bigcap_i A_i$ supports x. By construction $\pi \in fix(\bigcap_i A_i)$, and it follows that $\pi{\cdot}x = x$.
— *$supp(x)$ is least.* Suppose $A \subseteq supp(x)$. Suppose there exists $a \in supp(x) \backslash A$. Then choose b fresh (so $b \notin supp(x)$). By Lemma 3.7 $(b\ a){\cdot}supp(x) \neq supp(x)$; it follows that $(b\ a){\cdot}x \neq x$, so A does not support x. Therefore, $A = supp(x)$.

The case of part 2 is similar, but easier.

Parts 3 and 4 follow by unpacking the definition of what it is to support x (Definition 2.30) and considering $\pi^{-1} \circ \pi'$ and $\bar{\pi}^{-1} \circ \bar{\pi}'$. □

Corollary 3.10. — *$a \in supp(x)$ if and only if for fresh b (so $b\#x$) $(b\ a){\cdot}x \neq x$.*
— *$\bar{a} \in s\bar{u}pp(x)$ if and only if for fresh $\bar{b}$ (so $\bar{b}\#x$) such that $atoms(\bar{b}) = atoms(\bar{a})$, $(\bar{b}\ \bar{a}){\cdot}x = x$.*

3.2. *Functions and equivariance*

Definition 3.11. — Call $x \in |\mathsf{X}|$ **level 1 equivariant** when $\pi{\cdot}x = x$ for all level 1 permutations π.
— Call $x \in |\mathsf{X}|$ **level 2 equivariant** when $\bar{\pi}{\cdot}x = x$ for all level 2 permutations $\bar{\pi}$.

Example 3.12. Recall Examples 2.16 and 2.34.

1. In $\mathbb{A}$, all elements are level 2 equivariant and no elements are level 1 equivariant.
2. In $\bar{\mathbb{A}}$, no elements are level 1 or level 2 equivariant.
3. In $orb(\bar{\mathbb{A}})$, all elements are level 1 equivariant and no elements are level 2 equivariant.

Proposition 3.13. *$x \in |\mathsf{X}|$ is level 1 / level 2 equivariant if and only if $supp(x) = \varnothing$ / $s\bar{u}pp(x) = \varnothing$ respectively.*

Proof. By routine calculations. □

Recall the definition of $\mathsf{X} \to \mathsf{Y}$ from number 7 of Example 2.16).

Definition 3.14. Suppose X and Y are sets with a two-level permutation action. Suppose $f \in |\mathsf{X} \to \mathsf{Y}|$ is a function.

— Call f **level 1 equivariant** when $f(\pi{\cdot}x) = \pi{\cdot}f(x)$ for all level 1 permutations π.
— Call f **level 2 equivariant** when $f(\bar{\pi}{\cdot}x) = \bar{\pi}{\cdot}f(x)$ for all level 2 permutations $\bar{\pi}$.
— Call f **equivariant** when it is level 1 and level 2 equivariant.

Lemma 3.15. *Suppose $f \in |\mathsf{X} \to \mathsf{Y}|$. Then f is equivariant in the sense of Definition 3.11 ($supp(f) = \varnothing = s\bar{u}pp(f)$) if and only if f is equivariant in the sense of Definition 3.14 ($\pi{\cdot}f(x) = f(\pi{\cdot}x)$ and $\bar{\pi}{\cdot}f(x) = f(\bar{\pi}{\cdot}x)$).*

Proof. By routine calculations. □

Lemma 3.16. *Suppose X is a two-level nominal set. Then the following three conditions are equivalent:*

— *$\bar{\pi}{\cdot}(\pi{\cdot}x) = \pi{\cdot}(\bar{\pi}{\cdot}x)$ (this is the condition on two-level nominal sets from Definition 2.33).*
— *The function $\lambda x.\pi{\cdot}x \in |\mathsf{X} \to \mathsf{X}|$ is level 2 equivariant.*
— *The function $\lambda x.\bar{\pi}{\cdot}x \in |\mathsf{X} \to \mathsf{X}|$ is level 1 equivariant.*

Proof. Direct from the definitions. □

Remark 3.17. In nominal techniques, 'natural' functions tend to be equivariant. Examples are Lemma 2.25, Lemma 3.16, and Lemma 3.18. See also part 2 of Lemma 5.10.

Lemma 3.18. *Suppose X and Y are two-level nominal sets. The maps*

$$\lambda x.supp(x) \in |\mathsf{X} \to powerset(\mathbb{A})| \quad \text{and} \quad \lambda x.s\bar{u}pp(x) \in |\mathsf{X} \to powerset(\bar{\mathbb{A}})|$$

(with the conjugation action from Example 2.16) are equivariant.

Proof. A reformulation of Lemma 3.7. □

Lemma 3.19. *$supp(f(x)) \subseteq supp(f) \cup supp(x)$ and similarly for $s\bar{u}pp$.*

Proof. By routine calculations using Remark 2.18 and Theorem 3.9. □

3.3. *The cartesian product and exponential*

Definition 3.20. Suppose X and Y are two-level nominal sets. Define $\mathsf{X} \times \mathsf{Y}$ by:

- $|\mathsf{X} \times \mathsf{Y}|$ is $|\mathsf{X}| \times |\mathsf{Y}|$.
- $\pi\cdot(x,y) = (\pi\cdot x, \pi\cdot y)$ and $\bar{\pi}\cdot(x,y) = (\bar{\pi}\cdot x, \bar{\pi}\cdot y)$.

Define $\mathsf{X} \Rightarrow \mathsf{Y}$ by:

- $f \in |\mathsf{X} \Rightarrow \mathsf{Y}|$ when $f \in |\mathsf{X} \to \mathsf{Y}|$ (Example 2.16) and f has small (Definition 2.31) support to level 1 and level 2.
- The permutation action is the conjugation action, inherited from $\mathsf{X} \to \mathsf{Y}$.

Lemma 3.21. *If* X *and* Y *are two-level nominal sets then so are* $\mathsf{X} \times \mathsf{Y}$ *and* $\mathsf{X} \Rightarrow \mathsf{Y}$.

Proof. Direct from the constructions. □

Lemma 3.22. *A bijection between equivariant functions* $f \in |\mathsf{X}| \to |\mathsf{Y} \Rightarrow \mathsf{Z}|$ *and* $g \in |\mathsf{X} \times \mathsf{Y}| \to |\mathsf{Z}|$ *is given by currying and uncurrying. That is,*

- f *maps to* $\lambda x, y.f(x)(y)$.
- g *maps to* $\lambda x.\lambda y.g(x,y)$.

3.4. *The category of two-level nominal sets*

Definition 3.23. Define a category NOM2 by the following data:

- Objects are two-level nominal sets X.
- Arrows $f : \mathsf{X} \longrightarrow \mathsf{Y}$ are equivariant functions $f \in |\mathsf{X} \to \mathsf{Y}|$.

It is easy to check that NOM2 is indeed a category.

Definition 3.24. Let $\mathbb{B}$ be the two-level nominal set with $|\mathbb{B}| = \{0,1\}$ and the *trivial* permutation action.

That is, $\forall\pi.\forall x{\in}\{0,1\}.\pi\cdot x = x$ and $\forall\bar{\pi}.\forall x{\in}\{0,1\}.\bar{\pi}\cdot x = x$.

Proposition 3.25. NOM2 *is a topos:*

- *An initial object is the one-element set with trivial permutation actions.*
- *The cartesian product and exponential are* $\mathsf{X} \times \mathsf{Y}$ *and* $\mathsf{X} \Rightarrow \mathsf{Y}$ *from Definition 3.20.*
- *A subobject classifier is* $\mathbb{B}$.

Proof. The first two parts are routine given the results already proven. For the subobject classifier, it suffices to note that $X \subseteq |\mathsf{X}|$ is the underlying set of an object in NOM2 (so $x{\in}X$ implies $\forall\pi.\pi\cdot x{\in}X$ and $\forall\bar{\pi}.\bar{\pi}\cdot x{\in}X$) if and only if $\lambda x.$*if* $x{\in}X$ *then* 1 *else* 0 is an equivariant function in the sense of Definition 3.14. □

4. Atoms-abstraction

We now come to the theory of abstractions by level 1 and level 2 atoms.

Nominal techniques introduced the idea of an atoms-abstraction $[a]x$ in (Gabbay and Pitts 2001). This is a notion of α-abstraction that generalises beyond syntax, though in

the special case that x is an abstract syntax tree it coincides with what we would normally call 'real' α-abstraction.

In Definition 4.2 we reprise the nominal definition of $[a]x$ and extend it with a new definition $[\bar{a}]x$ of abstraction by a level 2 atom $\bar{a}$. The specific design of the definitions does not follow (Gabbay and Pitts 2001) and is based on a decomposition of abstraction into pairing and *permutation orbits*, following (Gabbay 2007, 2011).

In Subsection 4.2 we check that the permutation action we give atoms-abstraction matches up with the permutation action we obtain pointwise from atoms-abstractions as sets (permutation orbits) in Definition 4.1. This is interesting because it helps us to prove later results, but it is relevant also for another reason: One useful aspect of nominal techniques is its connection with sets foundations of mathematics, and Subsection 4.2 verifies that this connection remains sound.

Subsection 4.3 develops the theory of support and equality for atoms-abstraction. In spirit, the theorems follow (Gabbay and Pitts 2001) but the level 2 case has its own unique character and things do not play out entirely as one might expect. Perhaps the most important point is hidden in part 3 of Theorem 4.11, where we prove that $supp([\bar{a}]x) = supp(x) \cup atoms(\bar{a})$. In words, this expresses an important distinction that $[\bar{a}]x$ abstracts the *order* of the atoms in $\bar{a}$ in x, but it does not abstract the atoms themselves. This is a recurring theme in this paper: our notion of a level 2 atom is an order on an infinite list of level 1 atoms, rather than the (infinite collection of) level 1 atoms in the list.

Subsection 4.4 is very brief but makes the observation of the previous paragraph formal. $abs([\bar{a}]x)$ identifies the atoms in $\bar{a}$, but does not (and cannot) recover their order. This technical definition is useful later on in Section 4.6.

Finally, Subsections 4.5 and 4.6 develop and explore *atoms-concretion* and the theory of functions out of atoms-abstraction.

4.1. *Basic definition*

Definition 4.1. Suppose X is a two-level nominal set. Suppose $x \in |\mathsf{X}|$, and suppose $A \subseteq \mathbb{A}$ and $B \subseteq orb(\bar{\mathbb{A}})$ are small (Definition 2.31). Define **permutation orbits** $x\!\uparrow_A$ and $x\!\uparrow_B$ by:

$$x\!\uparrow_A = \{\pi{\cdot}x \mid \pi \in fix(A)\}$$
$$x\!\uparrow_B = \{\bar{\pi}{\cdot}x \mid \bar{\pi} \in \bar{fix}(B)\}$$

Definition 4.2. Suppose X is a two-level nominal set. Define **level 1 and level 2 atoms abstraction** $[\mathbb{A}]\mathsf{X}$ and $[\bar{\mathbb{A}}]\mathsf{X}$, which are sets with a two-level permutation action, as fol-

lows:

$$\begin{array}{ll} [a]x = (a,x)\fatslash_{supp(x)\backslash\{a\}} & |[\mathbb{A}]\mathsf{X}| = \{[a]x \mid a{\in}\mathbb{A},\ x{\in}|\mathsf{X}|\} \\ [\bar{a}]x = (\bar{a},x)\fatslash_{s\bar{u}pp(x)\backslash\{orb(\bar{a})\}} & |[\bar{\mathbb{A}}]\mathsf{X}| = \{[\bar{a}]x \mid \bar{a}{\in}\bar{\mathbb{A}},\ x{\in}|\mathsf{X}|\} \\ \\ \pi{\cdot}[a]x = [\pi(a)]\pi{\cdot}x & \bar{\pi}{\cdot}[a]x = [a]\bar{\pi}{\cdot}x \\ \pi{\cdot}[\bar{a}]x = [\pi{\cdot}\bar{a}]\pi{\cdot}x & \bar{\pi}{\cdot}[\bar{a}]x = [\bar{\pi}(\bar{a})]\bar{\pi}{\cdot}x \end{array}$$

Remark 4.3. We will prove a number of things about Definition 4.2:

- $[\mathbb{A}]\mathsf{X}$ and $[\bar{\mathbb{A}}]\mathsf{X}$ are two-level nominal sets. This is Corollary 4.7.
- The permutation action on $[a]x$ and $[\bar{a}]x$ given in Definition 4.2 coincides with the pointwise permutation action on $(a,x)\fatslash_{supp(x)\backslash\{a\}}$ and $(\bar{a},x)\fatslash_{supp(x)\backslash\{\bar{a}\}}$ respectively (Definition 4.4). This can be viewed as a kind of 'sanity check', but it turns out to be an independently useful result. This is Proposition 4.5.
- We describe the theory of equality on abstractions. This is Propositions 4.9 and 4.10.
- We describe the theory of support of abstractions. This is Theorem 4.11.

4.2. *The pointwise action*

Definition 4.4. Suppose X is a set with a two-level permutation action. Then the set of subsets $U \subseteq |\mathsf{X}|$ inherits the two-level permutation action *pointwise*, defined by:

$$\pi{\cdot}U = \{\pi{\cdot}u \mid u \in U\} \qquad \bar{\pi}{\cdot}U = \{\bar{\pi}{\cdot}u \mid u \in U\}$$

Proposition 4.5. *Suppose $A \subseteq \mathbb{A}$ and $B \subseteq \bar{\mathbb{A}}$ are small. Suppose X is a two-level nominal set and $x \in |\mathsf{X}|$. Then:*

- $\pi{\cdot}(x\fatslash_A) = (\pi{\cdot}x)\fatslash_{\pi\cdot A}$. *As a corollary, $\pi{\cdot}[a]x$ in the sense of Definition 4.2 is equal to $\pi{\cdot}[a]x$ in the sense of Definition 4.4.*
- $\bar{\pi}{\cdot}(x\fatslash_A) = (\bar{\pi}{\cdot}x)\fatslash_{\bar{\pi}\cdot A}$. *As a corollary, $\bar{\pi}{\cdot}[a]x$ in the sense of Definition 4.2 is equal to $\bar{\pi}{\cdot}[a]x$ in the sense of Definition 4.4.*
- *Similarly for $\pi{\cdot}[\bar{a}]x$ and $\bar{\pi}{\cdot}[\bar{a}]x$.*

4.3. *Support and equality*

Lemma 4.6. *Suppose X is a two-level nominal set and $x \in |\mathsf{X}|$. Suppose $a \in \mathbb{A}$ and $\bar{a} \in \bar{\mathbb{A}}$. Then:*

1. *$supp([a]x) \subseteq supp(x)\backslash\{a\}$ and $s\bar{u}pp([a]x) \subseteq s\bar{u}pp(x)$.*
2. *$supp([\bar{a}]x) \subseteq atoms(\bar{a}) \cup supp(x)$ and $s\bar{u}pp([\bar{a}]x) \subseteq s\bar{u}pp(x)\backslash\{orb(\bar{a})\}$.*

Proof. By routine arguments on the group action using Proposition 4.5. □

Corollary 4.7. *$[\mathbb{A}]\mathsf{X}$ and $[\bar{\mathbb{A}}]\mathsf{X}$ are two-level nominal sets.*

Proof. We need to check the properties in Definition 2.33. $[\mathbb{A}]\mathsf{X}$ and $[\bar{\mathbb{A}}]\mathsf{X}$ have a two-level permutation action by construction. The existence of small supporting sets follows from Lemma 4.6.

It remains to check that $\bar{\pi}\cdot\pi\cdot[a]x = \pi\cdot\bar{\pi}\cdot[a]x$ and $\bar{\pi}\cdot\pi\cdot[\bar{a}]x = \pi\cdot\bar{\pi}\cdot[\bar{a}]x$. This is by routine calculations using the fact that $\bar{\pi}\cdot\pi\cdot x = \pi\cdot\bar{\pi}\cdot x$ and (by Definition 2.11) $\bar{\pi}\cdot\pi\cdot\bar{a} = \pi\cdot\bar{\pi}\cdot\bar{a}$. □

Lemma 4.8. *Suppose* X *is a two-level nominal set and* $x \in |\mathsf{X}|$. *Suppose* $a \in \mathbb{A}$ *and* $\bar{a} \in \bar{\mathbb{A}}$.
Then $[a]x$ *and* $[\bar{a}]x$ *are graphs of partial functions.*
That is, $(y', y) \in [\bar{a}]x$ *and* $(y', z) \in [\bar{a}]x$ *imply* $y = z$, *and similarly for* $[a]x$.

Proof. We present only the proof for $[\bar{a}]x$.
Suppose $\bar{\pi}$ is such that $\mathit{fix}(\bar{\pi}) \subseteq \mathit{su\bar{p}p}(x)\backslash\{\mathit{orb}(\bar{a})\}$ and similarly for $\bar{\pi}'$. Suppose $\bar{\pi}(\bar{a}) = \bar{\pi}'(\bar{a})$. Then $\mathit{diff}(\bar{\pi}, \bar{\pi}') \cap \mathit{su\bar{p}p}(x) = \varnothing$. By Theorem 3.9 $\bar{\pi}\cdot x = \bar{\pi}'\cdot x$. □

Proposition 4.9. *Suppose* X *is a two-level nominal set and* $x \in |\mathsf{X}|$. *Suppose* $a, b \in \mathbb{A}$ *and* $\bar{a}, \bar{b} \in \bar{\mathbb{A}}$.

1. $[a]x = [b]y$ *if and only if* $b\#x$ *and* $(b\ a)\cdot x = y$.
2. $[\bar{a}]x = [\bar{b}]y$ *if and only if* $\mathit{atoms}(\bar{b}) = \mathit{atoms}(\bar{a})$, $\bar{b}\#x$, *and* $(\bar{b}\ \bar{a})\cdot x = y$.

Proof. We present only the proof for part 2. We prove two implications.

— *Left-to-right.* Suppose $[\bar{a}]x = [\bar{b}]y$.
By Definition 4.2 $(\bar{a}, x) \in [\bar{b}]y$ and it follows that $\bar{a} = \bar{\pi}\cdot\bar{b}$ for some $\bar{\pi}$. By Definition 2.11 $\mathit{atoms}(\bar{a}) = \mathit{atoms}(\bar{b})$.
By Definition 4.2 $(\bar{b}, (\bar{b}\ \bar{a})\cdot x) \in [\bar{b}]y$ and $(\bar{b}, y) \in [\bar{b}]y$. It follows by Lemma 4.8 that $(\bar{b}\ \bar{a})\cdot x = y$. We deduce that $\bar{b}\#x$ using Lemma 4.6.

— *Right-to-left.* Suppose $\mathit{atoms}(\bar{a}) = \mathit{atoms}(\bar{b})$, $\bar{b}\#x$, and $(\bar{b}\ \bar{a})\cdot x = y$. Then $[\bar{b}](\bar{b}\ \bar{a})\cdot x = [\bar{b}]y$. It is a fact that $(\bar{b}\ \bar{a}) \in \bar{\mathit{fix}}(\mathit{su\bar{p}p}(x)\backslash\{\mathit{orb}(\bar{a})\})$. Using Lemma 4.6 and Theorem 3.9 and some easy calculations we deduce that $[\bar{a}]x = [\bar{b}]y$. □

Proposition 4.10. — $[a]x = [a]x'$ *if and only if* $x = x'$.
— $[\bar{a}]x = [\pi\cdot\bar{a}]x'$ *if and only if* $\pi\cdot\mathit{atoms}(\bar{a}) = \mathit{atoms}(\bar{a})$ *and* $(\pi\cdot\bar{a}\ \bar{a})\cdot x = x'$.

Proof. Routine using Lemma 4.8 and Theorem 3.9. □

Theorem 4.11. $[\mathbb{A}]\mathsf{X}$ *and* $[\bar{\mathbb{A}}]\mathsf{X}$ *are two-level nominal sets. Furthermore:*

1. $\mathit{supp}([a]x) = \mathit{supp}(x)\backslash\{a\}$
2. $\mathit{su\bar{p}p}([a]x) = \mathit{su\bar{p}p}(x)$
3. $\mathit{supp}([\bar{a}]x) = \mathit{supp}(x) \cup \mathit{atoms}(\bar{a})$
4. $\mathit{su\bar{p}p}([\bar{a}]x) = \mathit{su\bar{p}p}(x)\backslash\{\mathit{orb}(\bar{a})\}$

Proof. 1. Suppose $b\#[a]x$. Choose b' fresh (so $b'\#x, [a]x$). By Corollary 3.10 $b\#[a]x$ if and only if $(b'\ b)\cdot[a]x = [a]x$. By Proposition 4.10 $(b'\ b)\cdot[a]x = [a]x$ if and only if $(b'\ b)\cdot x = x$. By Corollary 3.10 $(b'\ b)\cdot x = x$ if and only if $b\#x$.
2. Like the proof of part 1, using the fact that $(\bar{b}\ \bar{a})\cdot[a]x = [a](\bar{b}\ \bar{a})\cdot x$.
3. Suppose $b\#[\bar{a}]x$. Choose b' fresh (so $b'\#x, \bar{a}, [\bar{a}]x$). By Corollary 3.10 $b\#[\bar{a}]x$ if and only if $(b'\ b)\cdot[\bar{a}]x = [\bar{a}]x$. By Proposition 4.10 $(b'\ b)\cdot[\bar{a}]x = [\bar{a}]x$ if and only if $(b'\ b)\cdot\mathit{atoms}(\bar{a}) = \mathit{atoms}(\bar{a})$ and $((b'\ b)\cdot\bar{a}\ \bar{a})\cdot x = (b'\ b)\cdot x$. This happens if and only if $b\#\bar{a}$ and, by Corollary 3.10, $b\#x$.
4. Like the proof of part 1. □

4.4. *Abstracted level 1 atoms of a level 2 abstraction*

Note in Theorem 4.11 that $supp([\bar{a}]x) = atoms(\bar{a}) \cup supp(x)$ and $supp([\bar{a}]x) \neq supp(x) \backslash atoms(\bar{a})$. In $[\bar{a}]x$ we do not abstract the atoms in $\bar{a}$—we abstract the *order* in which they appear in $\bar{a}$.

With this in mind, the following definition will be useful later:

Definition 4.12. Suppose X is a two-level nominal set, $x \in |\mathsf{X}|$, and $\bar{a} \in \bar{\mathbb{A}}$. Define

$$abs([\bar{a}]x) = atoms(\bar{a}).$$

Lemma 4.13. *abs is well-defined.*

Proof. Suppose $[\bar{a}]x = [\bar{b}]y$. By Proposition 4.9 $atoms(\bar{a}) = atoms(\bar{b})$. □

4.5. *Concretion*

Definition 4.14. Suppose $x \in |[\mathbb{A}]\mathsf{X}|$. Write $x@a$ for the unique (by Lemma 4.8) element of $|\mathsf{X}|$ such that $(a, x@a) \in x$, when this element exists.
Suppose $x \in |[\bar{\mathbb{A}}]\mathsf{X}|$. Write $x@\bar{a}$ for the unique element of $|\mathsf{X}|$ such that $(\bar{a}, x@\bar{a}) \in x$, when this element exists.
We call $x@a$ / $x@\bar{a}$ **level 1 / 2 concretion**.

Lemma 4.15. — *If $x \in |[\mathbb{A}]\mathsf{X}|$ then $x@a$ exists if and only if $a\#x$.*
— *If $x \in |[\bar{\mathbb{A}}]\mathsf{X}|$ then $x@\bar{a}$ exists if and only if $\bar{a}\#x$.*

Proof. By construction and Theorem 4.11. □

Lemma 4.16. *Suppose X is a two-level nominal set.*

— *Suppose $x \in |[\mathbb{A}]\mathsf{X}|$ and $b\#x$.*
Then $([a]x)@b = (b\ a)\cdot x$ and $([a]x)@a = x$.
— *Suppose $x \in |[\bar{\mathbb{A}}]\mathsf{X}|$, $\bar{b}\#x$, and $\pi\cdot atoms(\bar{a}) = atoms(\bar{a})$.*
Then $([\bar{a}]x)@\bar{b} = (\bar{b}\ \bar{a})\cdot x$ and $([\bar{a}]x)@\pi\cdot\bar{a} = (\pi\cdot\bar{a}\ \bar{a})\cdot x$.
— *Suppose $x \in |[\mathbb{A}]\mathsf{X}|$ and $a\#x$.*
Then $[a](x@a) = x$.
— *Suppose $x \in |[\bar{\mathbb{A}}]\mathsf{X}|$, $\bar{a}\#x$, and $\pi\cdot atoms(\bar{a}) = atoms(\bar{a})$.*
Then $[\bar{a}](x@\pi\cdot\bar{a}) = (\pi\cdot\bar{a}\ \bar{a})\cdot x$.

Proof. By routine calculations using Lemma 4.8 and Theorem 3.9. □

It is now convenient to pause for a moment and explore some fine detail of abstractions at both levels:

Proposition 4.17. *Suppose $\bar{a}$ is a level 2 atom and suppose $a \in atoms(\bar{a})$ and $b \in atoms(\bar{a})$ are two distinct atoms. Then:*

— $[\bar{a}]a \neq [\bar{a}]b$.
— $[\bar{a}][a]a = [\bar{a}][b]b$.

— $[a][\bar{a}]a \neq [b][\bar{a}]b$.
— $[\bar{a}]\bar{a} = \pi{\cdot}[\bar{a}]\bar{a}$ *if and only if* $nontriv(\pi) \subseteq atoms(\bar{a})$ *or* $nontriv(\pi) \cap atoms(\bar{a}) = \varnothing$.

Proof. — By part 2 of Lemma 4.16 $([\bar{a}]a)@\bar{a} = a \neq b = ([\bar{a}]b)@\bar{a}$.
— By part 1 of Proposition 4.9 $[a]a = [b]b$. The result follows.
— By contradiction: by Proposition 4.9 if $[a][\bar{a}]a = [b][\bar{a}]b$ then $b\#[a][\bar{a}]a$. This is impossible by Theorem 4.11.
— By definition $\pi{\cdot}[\bar{a}]\bar{a} = [\pi{\cdot}\bar{a}]\pi{\cdot}\bar{a}$. We use Proposition 4.9 and some routine calculations to reduce to the question of whether $\pi{\cdot}atoms(\bar{a}) = atoms(\bar{a})$. The result follows by properties of sets.

□

Remark 4.18. Proposition 4.17 is designed to highlight some of the (perhaps less obvious) aspects of level 2 and level 1 abstraction.

First, we note that $[\bar{a}]a \neq [\bar{a}]b$ for two distinct atoms $a, b \in atoms(\bar{a})$. This emphasises that the index of an atom—where it appears in $\bar{a}$—really counts. *The order of the atoms in* $\bar{a}$ *matters, even under a* $\bar{a}$*-abstraction;* shades here of de Bruijn indexes (de Bruijn 1972).

Second, we note that this does not affect atoms-abstraction inside the level 2 atoms-abstraction. $[a]a = [b]b$ still holds no matter what.

Third, we note that the index of a and b in $\bar{a}$ (where they occur in $\bar{a}$) continues to matter, even under atoms-abstraction by a and b. *From* outside *the* $\bar{a}$*-abstraction,* a *and* b *remain visible by their index.*

Fourth, we note that the order of the atoms in $\bar{a}$ is nevertheless abstracted by a $\bar{a}$-abstraction.

Remark 4.19. In Remark 2.27 which opened Subsection 2.4, we asked why we use $\bar{a}$ to build atoms-abstraction and why in general we take $\bar{a}$ as our model of meta-variables, instead of using orbits under finite permutations $orb(\bar{a})$ (Definition 2.6). After all, $\bar{supp}$ and $\bar{fix}$ both return sets of orbits of level 2 atoms and not sets of level 2 atoms.

We could do that. We would obtain an alternative theory in which there is no distinction between $\bar{a}$ and $orb(\bar{a})$.

But then, $(b\ a){\cdot}\bar{a} = \bar{a}$ would hold and (using an appropriately updated version of Proposition 4.9) so would $[a][b]\bar{a} = [b][a]\bar{a}$, for $a, b \in atoms(\bar{a})$. In this author's opinion, this is wrong for the following reason: the context '$\lambda x.\lambda y.t$' is not generally taken to be equal to '$\lambda y.\lambda x.t$'.

Note however that if we know of our language that the order of binding will not matter, as in first-order logic where $\forall x.\forall y.\phi$ is always logically equivalent with $\forall y.\forall x.\phi$, then this might not matter.

Lemma 4.20. — *Suppose* $x \in |[\mathbb{A}]\mathbb{X}|$ *and* $a\#x$. *Then* $\pi{\cdot}(x@a) = (\pi{\cdot}x)@\pi(a)$.
— *Suppose* $x \in |[\mathbb{A}]\mathbb{X}|$ *and* $a\#x$. *Then* $\bar{\pi}{\cdot}(x@a) = (\bar{\pi}{\cdot}x)@a$.
— *Suppose* $x \in |[\bar{\mathbb{A}}]\mathbb{X}|$ *and* $\bar{a}\#x$. *Then* $\pi{\cdot}(x@\bar{a}) = (\pi{\cdot}x)@\pi{\cdot}\bar{a}$.
— *Suppose* $x \in |[\bar{\mathbb{A}}]\mathbb{X}|$ *and* $\bar{a}\#x$. *Then* $\bar{\pi}{\cdot}(x@\bar{a}) = (\bar{\pi}{\cdot}x)@\bar{\pi}(\bar{a})$.

Proof. We consider only the first and the third case and use Lemma 4.16.
— If $x = [a]x'$ then $x@a = x'$ and $(\pi{\cdot}x)@\pi(a) = \pi{\cdot}x'$.
If $x = [b]x'$ then $x@a = (b\ a){\cdot}x'$ and $(\pi{\cdot}x)@\pi(a) = (\pi(b)\ \pi(a)){\cdot}\pi{\cdot}x' = \pi{\cdot}((b\ a){\cdot}x')$.

– If $x = [\pi'\cdot\bar{a}]x'$ then $x@\bar{a} = (\bar{a}\ \pi'\cdot\bar{a})\cdot x'$ and $(\pi\cdot x)@\pi\cdot\bar{a} = (\pi\cdot\bar{a}\ \pi\cdot\pi'\cdot\bar{a})\cdot\pi\cdot x'$. By Lemma 2.25 $(\pi\cdot\bar{a}\ \pi\cdot\pi'\cdot\bar{a}) = (\bar{a}\ \pi'\cdot\bar{a})$ and by equivariance of the permutation actions (Definition 2.33) $(\bar{a}\ \pi'\cdot\bar{a})\cdot\pi\cdot x' = \pi\cdot(\bar{a}\ \pi'\cdot\bar{a})\cdot x'$. The result follows.
If $x = [\bar{b}]x'$ then $x@\bar{a} = (\bar{b}\ \bar{a})\cdot x'$ and $(\pi\cdot x)@\pi\cdot\bar{a} = (\pi\cdot\bar{b}\ \pi\cdot\bar{a})\cdot\pi\cdot x'$. By Lemma 2.25 $(\pi\cdot\bar{b}\ \pi\cdot\bar{a}) = (\bar{b}\ \bar{a})$ and by equivariance of the permutation actions (Definition 2.33) $(\bar{b}\ \bar{a})\cdot\pi\cdot x' = \pi\cdot(\bar{b}\ \bar{a})\cdot x'$. The result follows. □

4.6. *Arrows out of atoms-abstractions*

We know how to build atoms-abstractions; given a and x we build $[a]x$ and similarly for $\bar{a}$ and x. It is just as important, if not more important, to know how to *destruct* atoms-abstractions. That is, how do we build *functions on* atoms-abstractions?

Recall the definition of $abs(x)$ from Definition 4.12 and the definition of the exponential $X \Rightarrow Y$ from Definition 3.20.

Definition 4.21. Define maps between $f \in |(\bar{\mathbb{A}} \times \mathsf{X}) \Rightarrow \mathsf{Y}|$ such that $\bar{a}\#f(\bar{a},x)$ for all $\bar{a}$ and $x \in |\mathsf{X}|$, and $g \in |[\bar{\mathbb{A}}]\mathsf{X} \Rightarrow \mathsf{Y}|$ as follows:

– We map $f \in |(\bar{\mathbb{A}} \times \mathsf{X}) \Rightarrow \mathsf{Y}|$ to $g \in |[\bar{\mathbb{A}}]\mathsf{X} \Rightarrow \mathsf{Y}|$ such that $g(x') = f(\bar{a}, x'@\bar{a})$ for $\bar{a}$ such that $\bar{a}\#x'$ and $\bar{a}\#f$ and $atoms(\bar{a}) = abs(x)$.
– We map $g \in |[\bar{\mathbb{A}}]\mathsf{X} \Rightarrow \mathsf{Y}|$ to f such that $f(\bar{a},x) = g([\bar{a}]x)$.

Lemma 4.22. *The maps in Definition 4.21 are well-defined. The map from g to f and back is the identity. The map from f to g and back is the identity provided that f is level 2 equivariant.*

As a corollary, the maps define a bijection on arrows in NOM2.

Proof. The non-trivial part of well-definedness is to check that the choice of fresh $\bar{a}$ in the map from f to g, does not matter. Suppose $x' \in |[\bar{\mathbb{A}}]\mathsf{X}|$. Suppose $atoms(\bar{a}) = atoms(\bar{b}) = abs(x)$ and suppose $\bar{a}\#x'$ and $\bar{b}\#x'$. We need to check that $f(\bar{a}, x@\bar{a}) = f(\bar{b}, x@\bar{b})$. By assumption $\bar{a}\#f(\bar{a}, x@\bar{a})$ and $\bar{b}\#f(\bar{b}, x@\bar{b})$. The result follows using Theorem 3.9, properties of the conjugation action (Remark 2.18), and Lemma 4.20.

To check that the maps are inverse and so define a bijection, it suffices to check two things:

– Suppose $\bar{a}$ and $x \in |\mathsf{X}|$. Suppose $\bar{b}\#x$, and $atoms(\bar{b}) = atoms(\bar{a})$. Suppose $\bar{a}\#f$ and $\bar{b}\#f$. Then $f(\bar{b}, ([\bar{a}]x)@\bar{b}) = f(\bar{a}, x)$.
– Suppose $x' \in |[\bar{\mathbb{A}}]\mathsf{X}|$, $\bar{a}\#x'$, and $atoms(\bar{a}) = abs(x')$.
Then $g([\bar{a}](x'@\bar{a})) = g(x')$.

Both of these facts follow using Theorem 3.9 and Lemma 4.16. □

The case of level 1 atoms-abstractions is known from (Gabbay and Pitts 2001). See also (Gabbay 2011).

Definition 4.23. Define maps between $f \in |(\mathbb{A} \times \mathsf{X}) \Rightarrow \mathsf{Y}|$ such that $a\#f(a,x)$ for all a and $x \in |\mathsf{X}|$, and $g \in |[\mathbb{A}]\mathsf{X} \Rightarrow \mathsf{Y}|$ as follows:

– We map $f \in |(\mathbb{A} \times \mathsf{X}) \Rightarrow \mathsf{Y}|$ to $g \in |[\mathbb{A}]\mathsf{X} \Rightarrow \mathsf{Y}|$ such that $g(x') = f(a, x'@a)$ for a such that $a\#x'$ and $a\#f$.
– We map $g \in |[\mathbb{A}]\mathsf{X} \Rightarrow \mathsf{Y}|$ to f such that $f(a,x) = g([a]x)$.

Lemma 4.24. *The maps in Definition 4.23 are well-defined. The map from g to f and back is the identity. The map from f to g and back is the identity provided that f is level 1 equivariant.*

As a corollary, the maps define a bijection on arrows in NOM2.

Proof. Like the proof of Lemma 4.22, but simpler. □

5. Semantic nominal terms

We can now exploit what we have built, to construct datatypes of syntax-with-binding containing level 2 atoms. This extends nominal abstract syntax from (Gabbay and Pitts 2001) to datatypes with level 2 atoms and abstraction of level 2 atoms.

We do a little more than build a nominal abstract syntax style presentation of nominal-terms-up-to-binding, because there is binding for level 1 atoms but also for level 2 atoms. There is also a little more to this than just building the datatype, because we also give it a substitution action for level 2 atoms.

The similarity with nominal terms unknowns is of course deliberate and is developed in Section 6.

5.1. *The basic definition*

Definition 5.1. Fix some countably infinite set of **term-formers**. f, g, h will range over distinct term-formers.

Definition 5.2. Define **semantic nominal terms** inductively by:

$$r ::= a \mid \bar{a} \mid \mathsf{f}(r,\dots,r) \mid [a]r \mid [\bar{a}]r$$

We make these into a set with a two-level permutation action Sem as follows:

$$\begin{array}{lll} \pi{\cdot}a = \pi(a) & \pi{\cdot}\bar{a} = (\pi(a_i))_i & \pi{\cdot}\mathsf{f}(r_1,\dots,r_n) = \mathsf{f}(\pi{\cdot}r_1,\dots\pi{\cdot}r_n) \\ \pi{\cdot}[a]r = [\pi(a)]\pi{\cdot}r & \pi{\cdot}[\bar{a}]r = [\pi{\cdot}\bar{a}]\pi{\cdot}r & \end{array}$$

$$\begin{array}{lll} \bar{\pi}{\cdot}a = a & \bar{\pi}{\cdot}\bar{a} = \bar{\pi}(\bar{a}) & \bar{\pi}{\cdot}\mathsf{f}(r_1,\dots,r_n) = \mathsf{f}(\bar{\pi}{\cdot}r_1,\dots\bar{\pi}{\cdot}r_n) \\ \bar{\pi}{\cdot}[a]r = [a]\bar{\pi}{\cdot}r & \bar{\pi}{\cdot}[\bar{a}]r = [\bar{\pi}(\bar{a})]\bar{\pi}{\cdot}r & \end{array}$$

Lemma 5.3. Sem *is a two-level nominal set.*

Proof. Routine using Theorem 4.11. □

5.2. *Substitutions*

Definition 5.4. A **(semantic) level 2 substitution** is an element $\sigma \in |\bar{\mathbb{A}} \Rightarrow \mathsf{Sem}|$ (Proposition 3.25) such that $supp(\sigma) = \varnothing$ and $s\bar{u}pp(\sigma)$ is finite.

σ will range over semantic level 2 substitutions.

In words: σ is *equivariant at level 1 and finitely-supported at level 2*.

Lemma 5.5. *Suppose σ is a level 2 substitution. Then $supp(\sigma(\bar{a})) \subseteq atoms(\bar{a})$ for all $\bar{a} \in \bar{\mathbb{A}}$.*

Proof. Suppose $\pi \in \mathit{fix}(\mathit{atoms}(\bar{a}))$. By Lemma 3.15 $\pi{\cdot}\sigma(\bar{a}) = \sigma(\pi{\cdot}\bar{a})$, and by assumption $\sigma(\pi{\cdot}\bar{a}) = \sigma(\bar{a})$. It follows that $\mathit{atoms}(\bar{a})$ supports $\sigma(\bar{a})$. □

Definition 5.6. Given a substitution σ define a level 2 **substitution action** on Sem by:

$$a\sigma = a \qquad \bar{a}\sigma = \sigma(\bar{a}) \qquad \mathsf{f}(r_1,\ldots,r_n)\sigma = \mathsf{f}(r_1\sigma,\ldots,r_n\sigma)$$
$$([a]r)\sigma = [a](r\sigma) \qquad ([\bar{a}]r)\sigma = [\bar{a}](r\sigma)$$

In the clause for $[\bar{a}]r$ we choose $\bar{a}$ such that $\bar{a}\#\sigma$.[8]

Well-definedness of the action in Definition 5.6 follows using Lemmas 4.21 and 4.23, and Theorem 4.11.

Lemma 5.7. *Suppose $r \in |\mathsf{Sem}|$. Then $\pi{\cdot}(r\sigma) = (\pi{\cdot}r)\sigma$. That is, the substitution action is level 1 equivariant.*

Proof. By a routine induction on r. We briefly consider most cases; they are all just by definitions, except in the case of $\pi{\cdot}(\bar{a}\sigma)$:

- $\pi{\cdot}(a\sigma) = \pi(a)$ and $(\pi{\cdot}a)\sigma = \pi(a)$.
- $\pi{\cdot}(\bar{a}\sigma) = \pi{\cdot}\sigma(\bar{a})$. By Lemma 3.15 $\pi{\cdot}\sigma(\bar{a}) = \sigma(\pi{\cdot}\bar{a})$. Then by definition $\sigma(\pi{\cdot}\bar{a}) = (\pi{\cdot}\bar{a})\sigma$.
- $\pi{\cdot}(([a]r)\sigma) = [\pi(a)](\pi{\cdot}(r\sigma)) = [\pi(a)]((\pi{\cdot}r)\sigma) = (\pi{\cdot}[a]r)\sigma$.
- $\pi{\cdot}(([\bar{a}]r)\sigma) = [\pi{\cdot}\bar{a}](\pi{\cdot}(r\sigma)) = [\pi{\cdot}\bar{a}]((\pi{\cdot}r)\sigma) = (\pi{\cdot}[\bar{a}]r)\sigma$. □

Lemma 5.8. *$\mathit{supp}(r\sigma) \subseteq \mathit{supp}(r)$ and $\bar{\mathit{supp}}(r\sigma) \subseteq \bar{\mathit{supp}}(r) \cup \bar{\mathit{supp}}(\sigma)$.*

Proof. By Lemmas 5.7 and 3.19. □

Definition 5.9. Suppose $\bar{a}$ is a level 2 atom. Suppose X is a two-level nominal set and $x \in |\mathsf{X}|$. Suppose $\mathit{supp}(x) \subseteq \mathit{atoms}(\bar{a})$. Define an **atomic level 2 substitution** $[\bar{a}::=x]$ by:

$$[\bar{a}::=x](\pi{\cdot}\bar{a}) = \pi{\cdot}x \qquad [\bar{a}::=x](\bar{b}) = \bar{b}$$

Lemma 5.10. *1. $[\bar{a}::=x]$ is well-defined.*
2. $\mathit{supp}([\bar{a}::=x]) = \varnothing$ and $\bar{\mathit{supp}}([\bar{a}::=x]) \subseteq \{\mathit{orb}(\bar{a})\} \cup \bar{\mathit{supp}}(x)$.

Proof. We continue the notation of Definition 5.9.

For well-definedness, the slightly non-trivial part is to show that if $\pi{\cdot}\bar{a} = \pi'{\cdot}\bar{a}$ then $\pi{\cdot}x = \pi'{\cdot}x$. Suppose $\pi{\cdot}\bar{a} = \pi'{\cdot}\bar{a}$. We use Proposition 2.9, our assumption that $\mathit{supp}(x) \subseteq \mathit{atoms}(\bar{a})$, and Theorem 3.9.

The rest of the proof is by routine calculations. □

Lemma 5.11. *Suppose $a, b \in \mathit{atoms}(\bar{a})$, so that $[\bar{a}::=a]$ and $[\bar{a}::=b]$ are defined. Then:*

- $([a]\bar{a})[\bar{a}::=a] = [a]a$.
- $([a]\bar{a})[\bar{a}::=b] = [a]b$.

[8] The clause for $[a]r$ could have a similar condition $a\#\sigma$ but it is 'invisible' because $\mathit{supp}(\sigma) = \varnothing$ by assumption (Definition 5.4).

Proof. By unfolding definitions. □

Remark 5.12. Lemma 5.11 is suprising. After all, $a\#[a]\bar{a}$ (by Theorem 4.11) and also $a\#[\bar{a}::=a]$ (by Lemma 5.10). So how do $[a]\bar{a}$ and $[\bar{a}::=a]$ 'know' about a in the substitution if a is fresh for them?

They do not; but they remember its *index* within $\bar{a}$, that is, the position where it occurs in $\bar{a}$. This index is what makes $([a]\bar{a})[\bar{a}::=a]$ equal to $[a]a$.

At this level, 'nominal' ideas begin to converge with de Bruijn indexes (de Bruijn 1972).

6. Implementing semantic nominal terms

Semantic nominal terms from Definition 5.2 are non-finite because the $\bar{a}$ in $\bar{a}$ and $[\bar{a}]r$ is an infinite structure.

Nevertheless, semantic nominal terms are implementable. They admit an easy finite representation. This turns out to closely resemble nominal terms.[9]

In Subsection 6.1 we build *permissive nominal terms*. These build on ideas first introduced in (Dowek et al. 2010) and of course on nominal terms (Urban et al. 2004), though the nominal terms here have abstraction of both level 1 *and* level 2 atoms (atoms and unknowns, in the terminology of 'vanilla' nominal terms from (Urban et al. 2004)).[10]

Then, in Subsection 6.2 we inject permissive nominal terms into semantic nominal terms and show that the kernel of the map is exactly the notion of α-equivalence.

Thus we establish that in principle we can program on a finite representation (permissive nominal terms) and know that this *is* a representation of structures with good mathematical properties which we have developed earlier in the paper.

6.1. *The basic definition*

Remark 6.1. — *Finitely representing atoms (Definition 2.1).* Atoms may be represented as integers; we can take $\mathbb{A}^{<}$ to be negative integers and $\mathbb{A}^{>}$ to be non-negative integers.

[9] Now may be a good time to reiterate our motivations for building a non-trivial theory behind nominal terms. Nominal terms come equipped with an α-equivalence relation. It is there for a reason, and semantic nominal terms express that reason in a new and very simple manner: nominal terms' α-equivalence is the way it is, because it reflects equality of semantic nominal terms and more generally because it reflects equality of atoms-abstractions over two-level nominal sets.
Semantic nominal terms also extend 'nominal' inductive reasoning principles to nominal terms, show how to abstract over unknowns as well as atoms, give a mathematically non-obvious explanation of unknowns in terms of orderings on lists of atoms, and they link nominal sets/nominal abstract syntax, with two-level nominal sets/nominal terms.

[10] The pedant who notes that permissive nominal terms are not actually the same as nominal terms is right, and we direct them to a correspondence defined in (Dowek et al. 2010) which is bijective in a sense we make formal in that paper. To the level of detail that interests us now, they are close enough to be the same thing.

— *Finitely representing permission sets (Definition 2.3).* Given two sets U and V define $U \Delta V$ the **exclusive or** of U and V by:

$$U \Delta V = \{x \mid x \in U \wedge x \notin V\} \cup \{x \mid x \notin U \wedge x \in V\}$$

Then a permission set S may be represented by $S \Delta \mathbb{A}^<$; this is a finite set and uniquely identifies S, since $(S \Delta \mathbb{A}^<) \Delta \mathbb{A}^< = S$.

Definition 6.2. For each permission set fix a disjoint countably infinite set of **unknowns** of that permission set. X, Y, Z will range over distinct unknowns.

X, Y, Z may be represented as a pair (i, j) where i represents S and j represents the 'name' X of the unknown.

Write $p(X)$ for the permission set of X.

Definition 6.3. Recall from Definition 5.1 the term-formers f, g, h. These can easily be represented, e.g. by numbers. Define **(two-level) permissive nominal terms** by:

$$r, s, t ::= a \mid \pi \cdot X \mid \mathsf{f}(r, \ldots, r) \mid [a]r \mid [\pi \cdot X]r$$

This may be finitely represented in some standard way, like any inductive datatype.

We call $\pi \cdot X$ a **moderated unknown**.

Definition 6.4. Define $fa(r)$ the **free atoms** of r and $\pi \cdot r$ the **(atoms) permutation action** on r by:

$$\begin{array}{lll} fa(a) = \{a\} & fa(\pi \cdot X) = \pi \cdot p(X) & fa(\mathsf{f}(r_1, \ldots, r_n)) = \bigcup fa(r_i) \\ fa([a]r) = fa(r) \backslash \{a\} & fa([\pi \cdot X]r) = fa(r) & \end{array}$$

$$\begin{array}{lll} \pi \cdot a = \pi(a) & \pi \cdot (\pi' \cdot X) = (\pi \circ \pi') \cdot X & \pi \cdot \mathsf{f}(r_1, \ldots, r_n) = \mathsf{f}(\pi \cdot r_1, \ldots \pi \cdot r_n) \\ \pi \cdot [a]r = [\pi(a)]\pi \cdot r & \pi \cdot [\pi' \cdot X]r = [(\pi \circ \pi') \cdot X]\pi \cdot r & \end{array}$$

Definition 6.5. A **permutation of unknowns** is a map from unknowns to moderated unknowns such that distinct unknowns map to (moderations of) distinct unknowns and

$$\begin{array}{l} nontriv(\Pi) = \{X \mid \Pi(X) \neq id \cdot X\} \text{ is finite, and} \\ \pi \cdot p(X) = p(X) \text{ if } \Pi(X) = \pi \cdot X, \text{ and} \\ \pi \cdot p(Y) = p(X) \text{ if } \Pi(X) = \pi \cdot Y. \end{array}$$

Π will range over permutations of unknowns.

Definition 6.6. Define $fV(r)$ the **free unknowns** of r and $\Pi \cdot r$ the **(unknowns) permutation action** on r by:

$$\begin{array}{lll} fV(a) = \varnothing & fV(\pi \cdot X) = \{X\} & fV(\mathsf{f}(r_1, \ldots, r_n)) = \bigcup fV(r_i) \\ fV([a]r) = fV(r) & fV([\pi \cdot X]r) = fV(r) \backslash \{X\} & \end{array}$$

$$\begin{array}{lll} \Pi \cdot a = a & \Pi \cdot (\pi \cdot X) = \pi \cdot \Pi(X) & \Pi \cdot \mathsf{f}(r_1, \ldots, r_n) = \mathsf{f}(\Pi \cdot r_1, \ldots \Pi \cdot r_n) \\ \Pi \cdot [a]r = [a]\Pi \cdot r & \Pi \cdot [\pi \cdot X]r = [\pi \cdot \Pi(X)]\Pi \cdot r & \end{array}$$

Definition 6.7. Given X, Y, and π such that $\pi \cdot p(Y) = p(X)$, write $(\pi \cdot Y\ X)$ for the **swapping** permutation mapping X to $\pi \cdot Y$, Y to $\pi^{-1} \cdot X$, and all other Z to $id \cdot Z$. Also

if $\pi{\cdot}p(X) = p(X)$ then write $(\pi{\cdot}X\ X)$ for the **swapping** permutation mapping X to $\pi{\cdot}X$ and all other Z to $id{\cdot}Z$.

Definition 6.8. Define **α-equivalence** $r =_\alpha s$ inductively by:

$$\frac{}{a =_\alpha a}\ (=_\alpha \mathbf{a}) \qquad \frac{(\pi(a) = \pi'(a) \text{ all } a \in p(X))}{\pi{\cdot}X =_\alpha \pi'{\cdot}X}\ (=_\alpha \mathbf{X})$$

$$\frac{r_i =_\alpha s_i}{\mathsf{f}(r_1,\ldots,r_n) =_\alpha \mathsf{f}(s_1,\ldots,s_n)}\ (=_\alpha \mathsf{f})$$

$$\frac{r =_\alpha s}{[a]r =_\alpha [a]s}\ (=_\alpha [\mathbf{aa}]) \qquad \frac{(b\ a){\cdot}r =_\alpha s \quad (b \notin fa(r))}{[a]r =_\alpha [b]s}\ (=_\alpha [\mathbf{ab}])$$

$$\frac{((\pi^{-1}{\circ}\pi'){\cdot}X\ X){\cdot}r =_\alpha s}{[\pi{\cdot}X]r =_\alpha [\pi'{\cdot}X]s}\ (=_\alpha [\mathbf{XX}]) \qquad \frac{((\pi^{-1}{\circ}\pi'){\cdot}Y\ X){\cdot}r =_\alpha s \quad (Y \notin fV(r))}{[\pi{\cdot}X]r =_\alpha [\pi'{\cdot}Y]s}\ (=_\alpha [\mathbf{XY}])$$

Theorem 6.9. — *$\pi{\cdot}X =_\alpha \pi'{\cdot}X$ if and only if $\pi(a) = \pi'(a)$ for all $a \in p(X)$.*
— *$[a]r =_\alpha [a]s$ if and only if $r =_\alpha s$.*
— *$[a]r =_\alpha [b]s$ if and only if $b \notin fa(r)$ and $(b\ a){\cdot}r =_\alpha s$.*
— *$\mathsf{f}(r_1,\ldots,r_n) =_\alpha \mathsf{f}(s_1,\ldots,s_n)$ if and only if $r_i =_\alpha s_i$ for $1 \leqslant i \leqslant n$.*
— *$[\pi{\cdot}X]r =_\alpha [\pi'{\cdot}X]s$ if and only if $((\pi^{-1}{\circ}\pi'){\cdot}X\ X){\cdot}r =_\alpha s$.*
— *$[\pi{\cdot}X]r =_\alpha [\pi'{\cdot}Y]s$ if and only if $Y \notin fV(r)$ and $((\pi^{-1}{\circ}\pi'){\cdot}Y\ X){\cdot}r =_\alpha s$.*

Proof. By a routine argument on derivations. We consider one case: Suppose $[\pi{\cdot}X]r =_\alpha [\pi'{\cdot}Y]s$ is derivable with some derivation $\mathcal{D}$. We examine the rules in Definition 6.8 and see that $\mathcal{D}$ must conclude with $(=_\alpha [\mathbf{XY}])$. Therefore $Y \notin fV(r)$ and $((\pi^{-1}{\circ}\pi'){\cdot}Y\ X){\cdot}r =_\alpha s$. □

6.2. *The isomorphism between implementation and theory*

There are more semantic nominal terms than permissive nominal terms because there are uncountably many level 2 atoms and only countably many unknowns. We choose a fixed but arbitrary map γ to 'pick out' a countable sub-selection of the level 2 atoms to represent unknowns. Once we have done that it is not trivial, but routine, to bi-ject permissive-nominal terms quotiented by α-equivalence with (a subset of) semantic nominal terms; this is Theorem 6.14.

Definition 6.10. Fix a *choice of representatives* map γ from unknowns X to level 2 atoms $\bar{a}$ such that:

— *atoms*$(\gamma(X)) = p(X)$ for all X.
— The map $X \mapsto orb(\gamma(X))$ is injective.

So γ maps each unknown X to some fixed but arbitrary $\bar{a}$.

It does not matter how γ is obtained, or which γ we chose: all we need is that one exists.

Definition 6.11. Define a map $[\![\text{-}]\!]$ from permissive nominal terms to semantic nominal

terms by:

$$\begin{array}{lll} [\![a]\!] = a & [\![\pi{\cdot}X]\!] = \pi{\cdot}\gamma(X) & [\![\mathsf{f}(r_1,\ldots,r_n)]\!] = \mathsf{f}([\![r_1]\!],\ldots,[\![r_n]\!]) \\ [\![[a]r]\!] = [a][\![r]\!] & [\![[\pi{\cdot}X]r]\!] = [\pi{\cdot}\gamma(X)][\![r]\!] & \end{array}$$

Also define a map $[\![\text{-}]\!]$ on permutations of unknowns by:

- $[\![\Pi]\!](\pi{\cdot}\gamma(X)) = (\pi \circ \pi'){\cdot}\gamma(X)$ if $\Pi(X) = \pi'{\cdot}X$.
- $[\![\Pi]\!](\pi{\cdot}\gamma(X)) = (\pi \circ \pi'){\cdot}\gamma(Y)$ if $\Pi(X) = \pi'{\cdot}Y$.
- $[\![\Pi]\!](\bar{a}) = \bar{a}$ if there exists no X such that $orb(\bar{a}) = orb(\gamma(X))$ (so in a suitable sense $\bar{a}$ is not in the image of γ).

Lemma 6.12. *$[\![\Pi]\!]$ is a level 2 permutation, so $[\![\text{-}]\!]$ defines a map from permutations of unkowns to permutations of level 2 atoms.*

Proof. We look at Definition 2.11 and see that we need to check that $atoms([\![\Pi]\!](\gamma(X))) = atoms(\gamma(X))$, $\overline{nontriv}([\![\Pi]\!])$ is finite, and $[\![\Pi]\!]{\cdot}(\pi{\cdot}\bar{a}) = \pi{\cdot}[\![\Pi]\!](\bar{a})$. These can all be checked by unfolding definitions. □

Lemma 6.13. *1. $fa(r) = supp([\![r]\!])$ and $fV(r) = \overline{supp}([\![r]\!])$.*
2. $[\![\pi{\cdot}r]\!] = \pi{\cdot}[\![r]\!]$.
3. $X \in nontriv(\Pi)$ if and only if $orb(\gamma(X)) \in \overline{nontriv}([\![\Pi]\!])$.
4. $[\![\Pi{\cdot}r]\!] = [\![\Pi]\!]{\cdot}[\![r]\!]$.

Proof. We use Theorem 4.11 for the first part and routine calculations for the rest. □

With the results we have proved so far, it is easy to verify that permissive nominal terms (Definition 6.3) quotiented by α-equivalence (Definition 6.8) are isomorphic with a subset of semantic nominal terms (Definition 5.2).

Theorem 6.14. *$[\![r]\!] = [\![s]\!]$ if and only if $r =_\alpha s$.*

Proof. By further routine calculations. We use Lemma 6.13, Propositions 4.9 and 4.10, and Theorem 6.9. □

7. Conclusions

We have explored a semantic theory of *two-level nominal sets*, inspired by nominal sets (sets with a finitely-supported permutation action) and by nominal terms (first-order syntax with variable and meta-variable symbols).

We have seen that we can model a level 2 variable as an infinite sequence of the level 1 variables on which it depends. More precisely, it is the *ordering* on the permission set which an infinite sequence gives rise to, that matters. It is not clear that this should all work, but it does.

So in a succession of results, starting with Proposition 2.9 and culminating in the construction of semantic nominal terms in Section 5, we make the journey from nominal sets semantics to a concrete syntax, which we show how to implement in Section 6.

The fragment of Section 6 without level 2 atoms-abstraction—which following (Gabbay and Mathijssen 2008b) one could call the *one-and-a-halfth order* fragment—has been programmed by Mulligan as part of his thesis (Mulligan 2009).

The reader familiar with nominal sets and nominal abstract syntax ('nominal' syntax-with-binding) from (Gabbay and Pitts 2001) can think of semantic nominal terms as a nominal abstract syntax version of nominal terms, extended with abstraction by unknowns as well as atoms, and can think of two-level nominal sets as reflecting into nominal sets semantics the idea of unknowns coming from nominal terms. The unknown with a suspended permutation $\pi \cdot X$ in nominal terms corresponds to choosing a representative $\bar{a}$ and writing an element of its permutation equivalence class as $\pi \cdot \bar{a}$.

We would prefer the reader to maintain a sense of irony about this paper. It presents a model of nominal unknowns as infinite sequences: that should not be read as a claim that nominal unknowns *are* infinite sequences of atoms. We claim only that they can be modelled as such, and we make a (so far only semi-substantiated) suggestion that this model has potential to advance our understanding, e.g. by inspiring new semantics, algorithms, proofs and definitions.

Related work

The idea that 'small' should correspond to 'well-orderable' was explored in (Gabbay 2002). Abstraction by infinite sequences of atoms was considered in (Gabbay 2007).[11] A connection between well-orders and nominal terms unknowns was proposed in (Gabbay and Mulligan 2009a). This paper extends on that work by considering level 2 permutations and the notion of two-level nominal set.

Two companion pieces to this work are currently under development—not quite sequels, but motivated by the same general current of thinking:

- In one paper, we will concentrate on syntax and operational semantics (Gabbay 2010a). We will explore what nominal unification and nominal rewriting (Urban et al. 2004; Fernández and Gabbay 2007) look like if we model unknowns as ω-lists of atoms. We will check whether this model offers proofs of new properties, and shorter proofs of known properties. We will also check the computational properties of working with the syntax of Definition 5.2.
- In another paper, we will focus on models and logical theories (Gabbay 2010b). We will consider a version of nominal algebra (Gabbay and Mathijssen 2009) in which syntax and semantics admit infinitely-supported elements (but we retain valuations). We will study soundness and completeness, and how to move between finitely-supported and infinitely-supported models.

Nominal algebra has a valuation semantics; unknowns are given a denotation via a *valuation* (Gabbay and Mathijssen 2009, Definition 4.14). We speculate that it may be possible

[11] This notion of abstraction was not identical to the level 2 abstraction here. In (Gabbay 2007) $supp([\bar{a}]x) = supp(x) \backslash atoms(x)$. See part 2 of Lemma 41 in (Gabbay 2007) and contrast this with part 3 of Theorem 4.11 in this paper.

to give a *two-level* nominal sets semantics to nominal algebra, thus eliminating valuations entirely. Thus, just as atoms map to themselves, so would unknowns. Valuations would be eliminated. We could recover a denotation for instantiation of unknowns, by assuming a level 2 substitution action as discussed below in Future Work.

This author is not aware of any other sets-based denotation for meta-variables, and certainly not nominal ones.

The denotation implicit in (Gacek et al. 2009) is based on capture-avoiding substitution and *raising*—a limited form of function application. Similarly, the denotations in (Sun 1999; Dowek et al. 2002) are based on controlled forms of functional abstraction.

As has been observed by Levy and Villaret and by Dowek and the author (Levy and Villaret 2008; Dowek et al. 2009, 2010) a connection can be made between raising and the capturing substitution of nominal terms. However, the translation from nominal syntax to raised syntax is quadratic.

Not all models of meta-variables are functional. A semantic basis for meta-variables is implicit in the categorical constructions in (Fiore and Hur 2008). A notion of 'hole' is basic to (Cardelli et al. 2003) though this work operates in the context of a specific concrete model. Of course, variables ranging over nominal sets are also the semantics for nominal terms unknowns, which model meta-variables. Note that Fiore had also previously created a presheaf semantics for names and binding (Fiore et al. 1999), and as it turns out the presheaves used are virtually identical to nominal sets and in fact both were introduced in the self-same conference (Gabbay and Pitts 1999).

Semantic nominal terms have informed the design of *permissive-nominal logic* (Dowek and Gabbay 2010). The syntax of permissive-nominal logic is roughly equivalent to the implementation from Section 6. There is one subtlety; in (Dowek and Gabbay 2010) we took the permutation action to be $\pi \cdot \forall X.\phi = \forall X.\pi \cdot \phi$ (and not $\pi \cdot \forall X.\phi = \forall \pi \cdot X.\pi \cdot \phi$). For the special case of permissive-nominal logic this makes no difference because substitutions are equivariant, and it follows that $\forall X.\phi$ is equivalent to $\forall \pi \cdot X.\phi$.

The permissive-nominal terms syntax of this paper clearly generalises the permissive-nominal terms syntax of (Dowek et al. 2010). Note that (Dowek et al. 2010) also gives an elementary mapping from nominal terms to permissive-nominal terms. That is why we do not need to worry about the distinction between nominal terms and their permissive variant, in this paper.

Nominal logic from (Pitts 2003) is a first-order axiomatisation of nominal sets from (Gabbay and Pitts 2001). For the purposes of this discussion, nominal sets are equal to Fraenkel-Mostowski set theory minus the sets hierarchy. By design, this paper extends nominal sets with a level 2 permutation action—it would be interesting to write down a corresponding axiomatisation.

Contextual Modal Type Theory (Nanevski et al. 2008) has types for open code and a two-level notion of context with the two levels corresponding to 'values' an 'code'. Two-level nominal sets, which support a denotation of meta-variables, might be useful in giving new kinds of denotational semantics to this, and to logics and programming languages in a similar spirit. Indeed, it is an interesting open question to what extent nominal denotations in general can be brought to bear on this problem.

Future work

Add level 2 β*-reduction to the syntax.* We have substitution from Definition 5.9 and so it is easy to build a notion of level 2 β-reduction; it suffices to fix a binary *application* term-former and to define a congruence $\rightarrow$ on terms such that $([\bar{a}]r)s \rightarrow r[\bar{a}::=s]$ provided that $supp(s) \subseteq atoms(\bar{a})$. The properties of this rewrite system, such as confluence, types, and normalisation properties, remain to be investigated. Semantic nominal terms might help inform the design of the multi-level λ-calculi considered e.g. in (Gabbay and Lengrand 2008; Gabbay and Mulligan 2009b), which feature notions of β-reduct at multiple levels with the idea that this could model object- and meta-level computation.

Design choice in level 1 support of level 2 abstraction. We have chosen that $[\bar{a}]x$ abstracts the order of $\bar{a}$ in x, but not the atoms in $\bar{a}$: see part 3 of Theorem 4.11. This is one design choice, but it is not the only one.

It is easy to design a stronger definition (in the sense that more things become equal) such that it abstracts *also* the atoms in $\bar{a}$.

Our reasons for defining things as they are in this paper are twofold:

- The stronger definition can be obtained from the weaker definition in this paper via a further quotient or equivalence-class, but not vice-versa.
- If we interpret $[\bar{a}][a]\bar{a}$ as $\lambda X.[a]X$ (bearing in mind the mention of multi-level λ-calculi above) then we would like $(\lambda X.[a]X)a$ to be equal to/to reduce to $[a]a$. If λX abstracts $pmss(X)$ in $[a]X$, then this would fail and we would obtain $[b]a$ instead.

Add level 2 substitution to the denotation. One of the most interesting potential applications of two-level nominal sets is to impose a level 2 substitution action (substitution for level 2 atoms). This could be done either axiomatically, following (Gabbay and Mathijssen 2006), or by concrete sets constructions, following (Gabbay 2009). Being familiar with the constructions in (Gabbay 2009), this author expects the case of level 2 substitution to be *easier* than the level 1 substitution. If we can do this then we will have even more explicitly an 'abstract theory of meta-variables', in the sense that level 2 atoms will not only exist as they do in two-level nominal sets, but also be substitutable-for. Potentially, the applications of this are great, since many systems for meta-programming, contexts, objects, modules, and incomplete objects, might be modelled using it. This is future work.

Indeed, it might be possible to model the γ and δ variables from work like (Wirth 2004) using the two levels of variable of this paper. We have in mind that atoms correspond to δ (intuitively: universal) variables, and unknowns correspond to γ (intuitively: existential) variables, with $s\bar{u}pp(\bar{a})$ expressing the dependence/independence conditions generated by nested quantifiers. This too is future work.

Interpret level 2 abstraction in the λ*-calculus.* It remains to extend the correspondence between nominal terms and higher-order patterns of (Levy and Villaret 2008; Dowek et al. 2009, 2010) to the case of syntax with level 2 abstraction. This may or may not turn out to be possible.

Adjoints to level 2 abstraction. Given a two-level nominal set X write:

- $\mathsf{X} \otimes \mathbb{A}$ for the two-level nominal set with underlying set $\{(x, a) \mid x \in |\mathsf{X}|,\ a \in \mathbb{A},\ a \notin supp(x)\}$ and the natural permutation actions and
- $\mathsf{X} \otimes \bar{\mathbb{A}}$ for the two-level nominal set with underlying set $\{(x, \bar{a}) \mid x \in |\mathsf{X}|,\ \bar{a} \in \bar{\mathbb{A}},\ \bar{a} \notin s\bar{u}pp(x)\}$ and the natural permutation actions.

It is a fact that $\text{-} \otimes \mathbb{A}$ is left adjoint to $[\mathbb{A}]\text{-}$ (see (Gabbay and Pitts 2001) or (Gabbay 2011)). This does *not* hold of level 2 atoms: $\text{-} \otimes \bar{\mathbb{A}}$ is not left adjoint to $[\bar{\mathbb{A}}]\text{-}$. An intuition for why this is so is as follows: given an abstraction $x \in |[\mathbb{A}]\mathsf{X}|$, concretion $x@a$ is defined for *any* $a\#x$. However, given an abstraction $x \in |[\bar{\mathbb{A}}]\mathsf{X}|$ concretion $x@\bar{a}$ is defined only for $\bar{a}\#x$ such that $atoms(\bar{a}) = abs(x)$.

In a category of two-level FM sets (where we do not insist that the underlying set be equivariant), it may be possible to recover adjunctions via the two-level FM set $\bar{\mathbb{A}}_S = \{\bar{a} \in \bar{\mathbb{A}} \mid atoms(\bar{a}) = S\}$.

What is arguably the really important property of atoms-abstraction—the arrows out of atoms-abstraction developed in Subsection 4.6—*does* hold in the level 2 case of $[\bar{\mathbb{A}}]\mathsf{X}$.

Add level 1 *substitution to the denotation of unknowns.* An obvious generalisation of the notion of level 2 atom is to take it to be an ω-tuple such that all but finitely many elements of the tuple are distinct atoms. This is interesting because we suspect it could provide a nice model of substitution for atoms (if we substitute a for, say, 2 in $\bar{a}$, then we should get $\bar{a}$ in which a is replaced by 2). The same effect could be obtained by adding an explicit substitution or explicit parallel substitution, and this would be equivalent, but this author suspects that for the purposes of mathematical proof integrating the substitution into the unknown itself will be easier.

This paper is not the place to discuss the complexities of understanding substitution, but we can note that substitution underlies unification, quantification, λ-abstraction, explicit substitution calculi, calculi of incomplete objects, and more. The author has also considered axiomatisation and (abstract) models of substitution in (Gabbay and Mathijssen 2008a). Suffice it to say that a new and good concrete model of substitution would be a useful thing to have.

Non-syntactic applications of two-level nominal sets. In this paper we have applied two-level nominal sets to model (nominal terms) syntax.

But sets are very general. There is no reason we should stop there. Just as the sets model given by nominal sets has been used to model non-syntactic structures (like domains, functions, games, and so on) so we can imagine that two-level nominal sets might also be applied to non-syntactic structures. These remain to be investigated.

Atoms and unknowns generalise to variables with dependencies. In this paper, level 2 variables are infinite lists. We could easily take unknowns to be 'lists, plus extra information' (e.g. if we wanted more than one sort of unknown). The property we need to preserve is Proposition 2.9. Taking infinite lists is a convenient way to get this.

But we might take this much further. We could dispense entirely with the distinction

between atoms and unknowns and consider *variables* x and a *variable dependency relation* $x \leqslant y$ meaning intuitively 'y depends on x', of which the assertion '$a \in atoms(\bar{a})$' would be a special case.

This paper, then, studies a special case where x either depends on nothing (and is an atom) or depends on certain classes of atoms (permission-sets). Connections here with Fine's theory of *arbitrary objects* (Fine 1985), though Fine was more abstract than we would be, e.g. we would still have 'nominal' permutation-based notions of support and freshness.

If this can be done, then the 'lists' model of this paper is relevant because it suggests how to design a permutation action for variables with dependencies. Permutations of variables should satisfy those laws that they would satisfy, if variables were associated with lists of their variable dependencies, in order. Making these intuitions formal is future work.

Acknowledgements.

Thanks to two anonymous referees. Supported by grant RYC-2006-002131 at the Polytechnic University of Madrid.

References

Mirna Bognar. *Contexts in Lambda Calculus*. PhD thesis, Vrije Universiteit Amsterdam, 2002.

Luca Cardelli, Philippa Gardner, and Giorgio Ghelli. Manipulating trees with hidden labels. In *FoSSaCS 2003*, pages 216–232, 2003.

James Cheney. Completeness and Herbrand theorems for nominal logic. *Journal of Symbolic Logic*, 71:299–320, 2006.

James Cheney and Christian Urban. Nominal logic programming. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 30(5):1–47, 2008.

Nicolaas G. de Bruijn. Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation, with application to the Church-Rosser theorem. *Indagationes Mathematicae*, 5(34):381–392, 1972.

Gilles Dowek and Murdoch J. Gabbay. Permissive Nominal Logic. In *Principles and Practice of Declarative Programming, 12th International ACM SIGPLAN Symposium (PPDP 2010)*, 2010.

Gilles Dowek, Thérèse Hardin, and Claude Kirchner. Binding logic: Proofs and models. In *Proceedings of the 9th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR 2002)*, pages 130–144. Springer, 2002. ISBN 3-540-00010-0.

Gilles Dowek, Murdoch J. Gabbay, and Dominic P. Mulligan. Permissive Nominal Terms and their Unification. In *Proceedings of the 24th Italian Conference on Computational Logic (CILC'09)*, 2009.

Gilles Dowek, Murdoch J. Gabbay, and Dominic P. Mulligan. Permissive Nominal Terms and their Unification: an infinite, co-infinite approach to nominal techniques (journal version). *Logic Journal of the IGPL*, 18(6):769–822, 2010.

Maribel Fernández and Murdoch J. Gabbay. Nominal rewriting (journal version). *Information and Computation*, 205(6):917–965, June 2007.

Kit Fine. *Reasoning with Arbitrary Objects*. Blackwell, 1985.

Marcelo Fiore and Chung-Kil Hur. Term equational systems and logics. *Electronic Notes in Theoretical Computer Science*, 218:171–192, 2008.

Marcelo P. Fiore, Gordon D. Plotkin, and Daniele Turi. Abstract syntax and variable binding. In *Proceedings of the 14th IEEE Symposium on Logic in Computer Science (LICS 1999)*, pages 193–202. IEEE Computer Society Press, 1999.

Murdoch J. Gabbay. *A Theory of Inductive Definitions with alpha-Equivalence*. PhD thesis, University of Cambridge, UK, March 2001.

Murdoch J. Gabbay. FM-HOL, a higher-order theory of names. In F. Kamareddine, editor, *35 Years of Automath*, April 2002.

Murdoch J. Gabbay. A General Mathematics of Names. *Information and Computation*, 205 (7):982–1011, July 2007.

Murdoch J. Gabbay. A study of substitution, using nominal techniques and Fraenkel-Mostowski sets. *Theoretical Computer Science*, 410(12-13):1159–1189, March 2009.

Murdoch J. Gabbay. Meta-variables as infinite lists: computational properties in nominal unification and rewriting. 2010a. Submitted for publication.

Murdoch J. Gabbay. Permissive-nominal algebra with infinite name-restriction, using semantic-nominal terms. 2010b. Submitted for publication.

Murdoch J. Gabbay. Foundations of nominal techniques: logic and semantics of variables in abstract syntax. *Bulletin of Symbolic Logic*, 2011. In press.

Murdoch J. Gabbay and Stéphane Lengrand. The lambda-context calculus. *Electronic Notes in Theoretical Computer Science*, 196:19–35, January 2008.

Murdoch J. Gabbay and Aad Mathijssen. Capture-avoiding Substitution as a Nominal Algebra. In *ICTAC 2006: Theoretical Aspects of Computing*, volume 4281 of *Lecture Notes in Computer Science*, pages 198–212, November 2006.

Murdoch J. Gabbay and Aad Mathijssen. Capture-Avoiding Substitution as a Nominal Algebra. *Formal Aspects of Computing*, 20(4-5):451–479, June 2008a.

Murdoch J. Gabbay and Aad Mathijssen. One-and-a-halfth-order Logic. *Journal of Logic and Computation*, 18(4):521–562, August 2008b.

Murdoch J. Gabbay and Aad Mathijssen. Nominal universal algebra: equational logic with names and binding. *Journal of Logic and Computation*, 19(6):1455–1508, December 2009.

Murdoch J. Gabbay and Aad Mathijssen. A nominal axiomatisation of the lambda-calculus. *Journal of Logic and Computation*, 20(2):501–531, April 2010.

Murdoch J. Gabbay and Dominic P. Mulligan. Semantic nominal terms. In *TAASN*, March 2009a.

Murdoch J. Gabbay and Dominic P. Mulligan. Two-level lambda-calculus. In *Proceedings of the 17th International Workshop on Functional and (Constraint) Logic Programming (WFLP 2008)*, volume 246, pages 107–129, August 2009b.

Murdoch J. Gabbay and Andrew M. Pitts. A New Approach to Abstract Syntax Involving Binders. In *Proceedings of the 14th Annual Symposium on Logic in Computer Science (LICS 1999)*, pages 214–224. IEEE Computer Society Press, July 1999.

Murdoch J. Gabbay and Andrew M. Pitts. A New Approach to Abstract Syntax with Variable Binding. *Formal Aspects of Computing*, 13(3–5):341–363, July 2001.

Andrew Gacek, Dale Miller, and Gopalan Nadathur. A two-level logic approach to reasoning about computations. Submitted, November 2009.

Dimitri Hendriks and Vincent van Oostrom. Adbmal. In *CADE*, pages 136–150, 2003.

Leon Henkin, J. Donald Monk, and Alfred Tarski. *Cylindric Algebras*. North Holland, 1971 and 1985. Parts I and II.

J. Roger Hindley and Jonathan P. Seldin. *Lambda-Calculus and Combinators, An Introduction*. Cambridge University Press, 2nd edition, 2008.

Gueorgui I. Jojgov. Holes with binding power. In *TYPES*, volume 2646 of *Lecture Notes in Computer Science*, pages 162–181. Springer, 2002.

Jordi Levy and Mateu Villaret. Nominal unification from a higher-order perspective. In *Rewriting Techniques and Applications, Proceedings of RTA 2008*, volume 5117 of *Lecture Notes in Computer Science*. Springer, 2008.

Giulio Manzonetto and Antonino Salibra. Applying universal algebra to lambda calculus. *Journal of Logic and computation*, 20(4):877–915, August 2010.

Conor McBride. *Dependently Typed Functional Programs and their Proofs*. PhD thesis, University of Edinburgh, 1999.

Eugenio Moggi, Walid Taha, Zine-El-Abidine Benaissa, and Tim Sheard. An idealized MetaML: Simpler, and more expressive. In *Proceedings of the 8th European Symposium on Programming Languages and Systems (ESOP'99)*, volume 1576 of *Lecture Notes in Computer Science*, pages 193–207. Springer, 1999. ISBN 3-540-65699-5.

Dominic P. Mulligan. Implementation of permissive nominal terms. Available at `http://www2.macs.hw.ac.uk/~dpm8/permissive/perm.htm`, 2009.

Aleksandar Nanevski, Frank Pfenning, and Brigitte Pientka. Contextual modal type theory. *ACM Transactions on Computational Logic (TOCL)*, 9(3), 2008.

Cesar Mu noz. A calculus of substitutions for incomplete-proof representation in type theory. Technical Report 3309, INRIA, France, November 1997.

Lawrence C. Paulson. Isabelle: the next 700 theorem provers. In P. Odifreddi, editor, *Logic and Computer Science*, pages 361–386. Academic Press, 1990.

Andrew M. Pitts. Nominal logic, a first order theory of names and binding. *Information and Computation*, 186(2):165–193, 2003.

Masahiko Sato, Takafumi Sakurai, Yukiyoshi Kameyama, and Atsushi Igarashi. Calculi of meta-variables. In *CSL*, volume 2803 of *Lecture Notes in Computer Science*, pages 484–497, 2003.

Yong Sun. An algebraic generalization of Frege structures - binding algebras. *Theoretical Computer Science*, 211:189–232, 1999.

Carolyn Talcott. A theory of binding structures and applications to rewriting. *Theoretical computer science*, (112):99–143, 1993.

Alwen Tiu. A logic for reasoning about generic judgments. *Electronic Notes in Theoretical Computer Science*, 174(5):3–18, 2007.

Nikos Tzevelekos. Full abstraction for nominal general references. In *Proceedings of the 22nd IEEE Symposium on Logic in Computer Science (LICS 2007)*, pages 399–410. IEEE Computer Society Press, 2007.

Christian Urban, Andrew M. Pitts, and Murdoch J. Gabbay. Nominal Unification. In *CSL*, volume 2803 of *Lecture Notes in Computer Science*, pages 513–527. Springer, December 2003.

Christian Urban, Andrew M. Pitts, and Murdoch J. Gabbay. Nominal Unification. *Theoretical Computer Science*, 323(1–3):473–497, September 2004. ISSN 0304-3975.

Claus-Peter Wirth. Descente infinie + Deduction. *Logic Journal of the IGPL*, 12(1):1–96, 2004.