THE LANGUAGE OF STRATIFIED SETS IS CONFLUENT AND STRONGLY NORMALISING

MURDOCH J. GABBAY

Abstract. We study the properties of the language of Stratified Sets (first-order logic with $\in$ and a stratification condition) as used in TST, TZT, and (with stratifiability instead of stratification) in Quine's NF. We find that the syntax forms a nominal algebra for substitution and that stratification and stratifiability imply confluence and strong normalisation under rewrites corresponding naturally to β-conversion.

Contents

1. Introduction 2
1.1. About Stratified Sets 2
1.2. What this paper does 3
1.3. Some remarks 3
2. Background on nominal techniques 5
2.1. Basic definitions 5
2.1.1. Atoms and permutations 5
2.1.2. Permutation actions on sets 6
2.1.3. Sets with a permutation action 6
2.1.4. Nominal sets 6
2.2. Examples 7
2.2.1. Atoms 7
2.2.2. Cartesian product 7
2.2.3. Full function space 7
2.2.4. Finite-supported function space 7
2.2.5. Full powerset 7
2.2.6. Finite powerset 8
2.2.7. Atoms-abstraction 8
2.3. The principle of equivariance 9
3. Internal syntax 10
3.1. Basic definition 10
3.2. Some notation 11

Key words and phrases: Stratified syntax, typed set theory, Quine's New Foundations, nominal rewriting, nominal algebra.

Thanks to the editor and to the anonymous referees.

LOGICAL METHODS
IN COMPUTER SCIENCE

DOI:10.2168/LMCS-???

© M. Gabbay
Creative Commons

4. The sigma-action 13
4.1. Basic definition and well-definedness 13
4.2. Nominal algebraic properties of the sigma-action 15
4.2.1. Alpha-equivalence of the sigma-action 16
4.2.2. Property (σ#) (garbage collection) 17
4.2.3. σ commutes with atoms-concretion 18
4.2.4. σ commutes with itself: the 'substitution lemma' 18
4.2.5. ($\sigma\mathbf{id}$): substitution for atoms and its corollaries 21
4.3. Sigma-algebras and SUB 22
4.4. The sugar $y{\in}x$ and its properties 22
5. The language of Typed Sets 24
5.1. Syntax of Stratified Sets 25
5.2. Interpretation for formulae and terms 26
5.3. Properties of the interpretation 27
5.4. Confluence 28
5.5. Strong normalisation 29
6. Conclusions and future work 33
References 34

1. Introduction

1.1. **About Stratified Sets.** Consider Russell's paradox, that if $s = \{a \mid a \notin a\}$ then $s \in s$ if and only if $s \notin s$. One way to avoid the term s is to restrict to the language of *Stratified Sets*. This is first-order logic with a binary relation $t{\in}s$ whose intuition is 't is an element of s' and:

- Variable symbols a (called *atoms* in this paper) are assigned levels, which are typically integers or natural numbers.
- We impose a *stratification* typing condition that we may only form $t{\in}s$ if the level of s is one plus the level of t. $level(s) = level(t)$+1.

See Definition 5.3 for full details. Then $s = \{a \mid a \notin a\}$ cannot be stratified, since whatever level we assign to a in $a \in a$ we cannot make $level(a) = level(a)$+1.

Stratified Sets are one of a family of syntaxes designed to exclude Russell's paradox:

- The language of ZF set theory restricts sets comprehension to bounded comprehension $\{a \in X \mid \phi\}$.
- Type Theories (such as Higher-Order Logic) impose more or less elaborate type systems. The canonical example of this is *simple types* $\tau ::= \iota \mid \tau \to \tau$.
- Stratified Sets stratifies terms as described.

One feature of Stratified Sets is that we can write a term representing the universal set:

$$\mathbf{univ} = \{a \mid \top\}$$

is easily stratified by giving a any level we like. Likewise we can write definitions such as 'the number 2' to be 'the set of all two-element sets':

$$\mathbf{2} = \{a \mid \exists b, c.(a = \{b, c\} \wedge b \neq c)\}$$

(Here we freely use syntactic sugar for readability; this can all be made fully formal.)

This feels liberating: we have the pleasure of full unbounded sets comprehension[1] and we have the pleasure of more sweeping types than are possible in the usual type theories such as Higher-Order Logic and its elaborations.[2]

1.2. **What this paper does.** The published literature using Stratified Sets does not view the basic syntax from the point of view of rewriting. On this topic, this paper makes three observations:

(1) The stratification condition implies that the syntax is confluent and strongly normalising under the natural rewrite
$$t \in \{a \mid \phi\} \quad \to \quad \phi[a{:=}t].$$
We can write:
Stratification $\Rightarrow$ *confluence and strong normalisation*.
Similarly for stratifiability. See Theorems 5.30 and 5.32.

(2) The syntax of normal forms becomes an algebra for substitution in a sense that will be made formal using nominal algebra.
See Theorem 4.19; in fact the proof of Theorem 5.30 uses this.

(3) Our proof is constructed using nominal techniques. The proofs in this paper should be fairly directly implementable in a nominal theorem-prover, such as Nominal Isabelle [Urb08].

In some senses, this paper is deliberately conventional, even simple: we write down a syntax and a rewrite relation and prove some nice properties. But the simplicity is deceptive:

- TST, TZT, and NF as usually presented do not include sets comprehension in their syntax, if that syntax is even made fully formal. So just noting that there are rewrite relations here that might be useful to look at, seems to be a new observation.
- The proofs are not trivial. It is easy to give a handwaving argument (such as that given in the first half of Remark 1.6 below) but it is surprisingly difficult to give a rigorous proof with all details.

More on this in Section 6.

We use nominal techniques (see the material in Section 2) to manage the α-binding in the syntax for universal quantification and sets comprehension. If the reader is unfamiliar with nominal techniques then they can just ignore this aspect: wherever we see reference to a nominal theorem, we can replace it with 'by α-conversion' or with 'it is a fact of syntax that'. The result should then be close to the kind of argument that might normally pass without comment or challenge.

1.3. **Some remarks.**

Remark 1.1. 'The language of Stratified Sets' is a description specific to this paper. In the literature, this syntax is unnamed and presented along with the theory we express using it:

(1) TST (which stands for Typed Set Theory) is typically taken to be first-order logic with $\in$ and variables stratified as $\mathbb{N} = \{0, 1, 2, \dots\}$, along with reasonable axioms for first-order logic and extensional sets equality.

(2) TZT is typically taken to be as the syntax and axioms of TST but with variables stratified as $\mathbb{Z} = \{0, 1, \text{-}1, 2, \text{-}2, \dots\}$.

[1] It still needs to be stratified, of course, but by Russell's paradox we must expect our party to be spoiled. Our choices are only: how, and where?

[2] Two attitudes are possible with types: embrace and enrich them, which leads us in the direction of (for instance) dependent types, or minimise type structure. Stratification minimises types all the way down to just being '$i \in \mathbb{Z}$'.

(3) Quine's New Foundations (NF) uses the language of first-order logic with $\in$, and reasonable axioms, and a **stratifiability condition** that variables *could* be stratified.

So in TST we would have to write (say) $a^1 \in b^2$ (choosing a level 1 variable symbol a and a level 2 variable symbol b), whereas in NF we could write just $a \in b$ and say "we *could* assign a level 1 and b level 2".

Remark 1.2. There is a slight ambiguity when we talk about stratification whether we insist that the syntax come delivered with an assignment of levels to all terms, or whether we insist on the weaker condition that an assignment *could* be made, but this assignment need not be a structural part of the formula or term. This distinguishes the languages of TST and TZT from that of NF: TST and TZT insist on stratification, and NF insists on stratifiability.

Our results will be agnostic in this choice (see for instance Theorems 5.30 and 5.32). So when we write *Stratified Sets* we could just as well write *Stratifiable Sets* and everything would still work with only minor bookkeeping changes.

Remark 1.3. The reader with a background in TST, TZT, and NF should note that this is not a paper about logical theories: it is a paper about their *syntax*. This is why we talk about 'Stratified Sets' in this paper, and not e.g. 'Typed Set Theories'. Our protagonist is a language, not a logic.

Remark 1.4 (Some words on terminology)**.** Some authors expand TST as 'Theory of Simple Types'. I think this terminology invites confusion with Simple Type Theory, so I prefer the alternative 'Typed Set Theory'. I would also like to write that TZT stands for 'Typed Zet Theory', but really TZT just stands for itself.

Remark 1.5 (References)**.** For the reader interested in the logical motivations for these syntaxes we provide references:

- A historical account of Russell's paradox is in [Gri04].
- For ZF set theory, see e.g. [Jec06].
- Excellent discussions of TST, TZT, and NF are in [For95] and [Hol98], and a clear summary with a brief but well-chosen bibliography is in [For97].

Remark 1.6 (Connection to the λ-calculus)**.** One way to see that something *like* this paper *should* work, fingers crossed, is by an analogy:

- The rewrite $t \in \{a \mid \phi\} \to \phi[a{:=}t]$ can be rewritten as $(\lambda a.\phi)t \to \phi[a{:=}t]$.
- Extensionality is $s = \{b \mid b \in s\}$, and we can rewrite this as $s = \lambda b.(sb)$.

These are of course familiar as β-reduction and η-expansion. The proofs need to be checked but the analogy above invites an analysis of the kind that we will now carry out.

And indeed this has been done, though not for stratified sets. In [KA10] (many thanks to an anonymous referee for bringing this to my attention) a development analogous to what is done in this paper for stratified syntax using nominal techniques, is carried out for the simply-typed lambda-calculus using de Bruijn indexes. Definitions and results bear a very nice comparison: for instance, Lemma 6 of [KA10] corresponds to Proposition 5.12.[3]

A technical device in [KA10], which goes back to a quite technical development in [WCPW03], is to work directly with a datatype of normal forms and substitution on them; this is just like the *internal syntax* and its *sigma-action* that we will see in this paper.

[3]There are also differences: In [KA10] the authors implement their proof in Agda, whereas implementation of the proofs in this paper for future work. On the other hand, proofs in this paper are given more-or-less in full, whereas in [KA10] the authors elide technical details. This paper proves confluence and strong normalisation, which is strictly more than [KA10] which considers only the existence of a reduction path to normal forms.

For me the motivation for setting things up in this way is partly practical and partly abstract: internal syntax with its sigma-action turns out to be a nominal sigma-algebra (Theorem 4.19) and this ties in with a literature on advanced nominal models of logic and computation [Gab16, GG17, GM08]. This paper was originally conceived as a prelude to building advanced nominal models of stratified and stratifiable syntaxes and type theories, though it has acquired independent interest.

It is therefore interesting to see analogous design choices appearing independently, motivated by apparently purely implementational concerns: what is good for the abstract mathematics also seems to be good for the proof-engineering.

2. Background on nominal techniques

Intuitively, a nominal set is "a set X whose elements $x \in \mathsf{X}$ may 'contain' finitely many names $a, b, c \in \mathbb{A}$". We may call names *atoms*. The notion of 'contain' used here is not the obvious notion of 'is a set element of': formally, we say that x has *finite support* (Definition 2.10).

Nominal sets are formally defined in Subsection 2.1. Examples are in Subsection 2.2. The reader might prefer to read this section only briefly at first, and then use it as a reference for the later sections where these underlying ideas get applied. More detailed expositions are also in [GP01, Gab11, DG12a, Pit13].

In the context of the broader literature, the message of this section is as follows:

- The reader with a category-theory background can read this section as exploring the category of nominal sets, or equivalently the Schanuel topos (more on this in [MM92, Section III.9], [Joh03, A.21, page 79], or [Gab11, Theorem 9.14]).
- The reader with a sets background can read this section as stating that we use Fraenkel-Mostowski set theory (**FM sets**).

  A discussion of this sets foundation, tailored to nominal techniques, can be found in [Gab11, Section 10]. FM sets add *urelemente* or *atoms* to the sets universe.
- The reader uninterested in foundations can note that previous work [GP01, Gab11, DG12a] has shown that just assuming names as primitive entities in Definition 2.1 yields a remarkable clutch of definitions and results, including Theorem 2.12, Corollary 2.13, and Theorem 2.26.

2.1. Basic definitions.

2.1.1. *Atoms and permutations.*

Definition 2.1. • For each $i{\in}\mathbb{Z}$ fix a disjoint countably infinite set $\mathbb{A}^i$ of **atoms**.[4]
- Write $\mathbb{A} = \bigcup_{i\in\mathbb{Z}} \mathbb{A}^i$.
- If $a{\in}\mathbb{A}$ (so a is an atom) write $level(a)$ for the unique number such that $a{\in}\mathbb{A}^{level(a)}$.
- We use a **permutative convention** that $a, b, c, \ldots$ range over *distinct* atoms.

  If we do not wish to use the permutative convention then we will refer to the atom using n (see for instance $(\sigma\texttt{eltatm})$ of Figure 2).

[4] These will serve as variable symbols in Definition 3.3.

2.1.2. *Permutation actions on sets.*

Definition 2.2. Suppose $\pi : \mathbb{A} \cong \mathbb{A}$ is a bijection on atoms.

(1) If $nontriv(\pi) = \{a \mid \pi(a) \neq a\}$ is finite then we call π **finite**.
(2) If $\pi(a) \in \mathbb{A}^i \Leftrightarrow a \in \mathbb{A}^i$ then call π **sort-respecting**.
(3) A **permutation** π is a finite sort-respecting bijection on atoms.
Henceforth π will range over permutations.

We will use the following notations in the rest of this paper:

Notation 2.3. (1) Write id for the **identity** permutation such that $\mathrm{id}(a) = a$ for all a.
(2) Write $\pi' \circ \pi$ for composition, so that $(\pi' \circ \pi)(a) = \pi'(\pi(a))$.
(3) If $i{\in}\mathbb{Z}$ and $a, b{\in}\mathbb{A}^i$ then write $(a\ b)$ for the **swapping** (terminology from [GP01]) mapping a to b, b to a, and all other c to themselves, and take $(a\ a) = \mathrm{id}$.
(4) Write π^{-1} for the inverse of π, so that $\pi^{-1} \circ \pi = \mathrm{id} = \pi \circ \pi^{-1}$.

2.1.3. *Sets with a permutation action.*

Notation 2.4. If $A \subseteq \mathbb{A}$ write

$$\mathit{fix}(A) = \{\pi \mid \forall a{\in}A.\pi(a) = a\}.$$

Definition 2.5. A **set with a permutation action** X is a pair $(|\mathsf{X}|, \cdot)$ of an **underlying set** $|\mathsf{X}|$ and a **permutation action** written $\pi{\cdot}x$ which is a group action on $|\mathsf{X}|$, so that $\mathrm{id}{\cdot}x = x$ and $\pi{\cdot}(\pi'{\cdot}x) = (\pi \circ \pi'){\cdot}x$ for all $x \in \mathsf{X}$ and permutations π and π'.

Definition 2.6. (1) Say that $A \subseteq \mathbb{A}$ **supports** $x \in \mathsf{X}$ when $\forall \pi.\pi \in \mathit{fix}(A) \Rightarrow \pi{\cdot}x = x$.
(2) If a finite $A \subseteq \mathbb{A}$ supporting x exists, call x **finitely supported** (by A) and say that x has **finite support**.

Notation 2.7. If X is a set with a permutation action then we may write

- $x \in \mathsf{X}$ as shorthand for $x \in |\mathsf{X}|$, and
- $X \subseteq \mathsf{X}$ as shorthand for $X \subseteq |\mathsf{X}|$.

2.1.4. *Nominal sets.*

Definition 2.8. Call a set with a permutation action X a **nominal set** when every $x \in \mathsf{X}$ has finite support. X, Y, Z will range over nominal sets.

Definition 2.9. Call a function $f \in \mathsf{X}{\Rightarrow}\mathsf{Y}$ **equivariant** when $\pi{\cdot}(f(x)) = f(\pi{\cdot}x)$ for all permutations π and $x \in \mathsf{X}$. In this case write $f : \mathsf{X}{\Rightarrow}\mathsf{Y}$.

The category of nominal sets and equivariant functions between them is usually called the category of *nominal sets*.

Definition 2.10. Suppose X is a nominal set and $x \in \mathsf{X}$. Define the **support** of x by

$$\mathit{supp}(x) = \bigcap\{A{\subseteq}\mathbb{A} \mid A \text{ is finite and supports } x\}.$$

Notation 2.11. • Write $a\#x$ as shorthand for $a \notin \mathit{supp}(x)$ and read this as a is **fresh for** x.
- If $T{\subseteq}\mathbb{A}$ write $T\#x$ as shorthand for $\forall a{\in}T.a\#x$.
- Given atoms $a_1, \ldots, a_n$ and elements $x_1, \ldots, x_m$ write $a_1, \ldots, a_n\#x_1, \ldots, x_m$ as shorthand for $\forall 1{\leq}j{\leq}m.\{a_1, \ldots, a_n\}\#x_j$. That is: $a_i\#x_j$ for every i and j.

Theorem 2.12. *Suppose* X *is a nominal set and* $x \in \mathsf{X}$*. Then* $supp(x)$ *is the unique least finite set of atoms that supports* x*.*

Proof. See [Gab11, Theorem 2.21(1)]. □

Corollary 2.13. (1) *If* $\pi(a) = a$ *for all* $a \in supp(x)$ *then* $\pi{\cdot}x = x$*. Equivalently:*
(a) *If* $\pi \in fix(supp(x))$ *then* $\pi{\cdot}x = x$*.*
(b) *If* $\forall a{\in}\mathbb{A}.(\pi(a){\neq}a \Rightarrow a\#x)$ *then* $\pi{\cdot}x = x$ *(see Notation 2.11).*
(2) *If* $\pi(a) = \pi'(a)$ *for every* $a{\in}supp(x)$ *then* $\pi{\cdot}x = \pi'{\cdot}x$*.*
(3) $a\#x$ *if and only if* $\exists b.(b\#x \wedge (b\ a){\cdot}x = x)$*.*

Proof. By routine calculations from the definitions and from Theorem 2.12 (see also [Gab11, Theorem 2.21(2)]). □

2.2. **Examples.** Suppose X and Y are nominal sets. We consider some examples, some of which will be useful later.

2.2.1. *Atoms.* $\mathbb{A}$ is a nominal set with the *natural permutation action* $\pi{\cdot}a = \pi(a)$.

2.2.2. *Cartesian product.* $\mathsf{X} \times \mathsf{Y}$ is a nominal set with underlying set $\{(x, y) \mid x \in \mathsf{X}, y \in \mathsf{Y}\}$ and the *pointwise* action $\pi{\cdot}(x, y) = (\pi{\cdot}x, \pi{\cdot}y)$.

It is routine to check that $supp((x, y)) = supp(x){\cup}supp(y)$.

2.2.3. *Full function space.* $\mathsf{X}{\to}\mathsf{Y}$ is a set with a permutation action with underlying set all functions from $|\mathsf{X}|$ to $|\mathsf{Y}|$, and the **conjugation** permutation action

$$(\pi{\cdot}f)(x) = \pi{\cdot}(f(\pi^{-1}{\cdot}x)).$$

2.2.4. *Finite-supported function space.* $\mathsf{X}{\Rightarrow}\mathsf{Y}$ is a nominal set with underlying set the functions from $|\mathsf{X}|$ to $|\mathsf{Y}|$ with finite support under the conjugation action, and the conjugation permutation action.

2.2.5. *Full powerset.*

Definition 2.14. Suppose Z is a set with a permutation action. Give subsets $Z \subseteq \mathsf{Z}$ the **pointwise** permutation action

$$\pi{\cdot}Z = \{\pi{\cdot}z \mid z \in Z\}.$$

Then $pset(\mathsf{Z})$ (the full powerset of Z) is a set with a permutation action with

- underlying set $\{Z \mid Z \subseteq \mathsf{Z}\}$ (the set of all subsets of $|\mathsf{Z}|$), and
- the pointwise action $\pi{\cdot}Z = \{\pi{\cdot}z \mid z \in Z\}$.

A particularly useful instance of the pointwise action is for sets of atoms. As discussed in Subsection 2.2.1 above, if $a \in \mathbb{A}$ then $\pi{\cdot}a = \pi(a)$. Thus if $A \subseteq \mathbb{A}$ then

$$\pi{\cdot}A \quad \text{means} \quad \{\pi(a) \mid a \in A\}.$$

Lemma 2.15. *Even if* Z *is a nominal set,* $pset(\mathsf{Z})$ *need not be a nominal set.*

Proof. Take $\mathsf{Z} = \mathbb{A}$ which we enumerate as $\{a_0, a_1, a_2, \dots\}$ and we take $Z \in pset(\mathsf{Z})$ to be equal to $comb$ defined by

$$comb = \{a_0, a_2, a_4, \dots\}.$$

This does not have finite support (see also [Gab11, Remark 2.18]). □

2.2.6. *Finite powerset.* For this subsection, fix a nominal set X.

Definition 2.16. Write $FinPow(\mathsf{X})$ for the nominal set with

- underlying set the set of all finite subsets of X,
- with the pointwise action from Definition 2.14.

Notation 2.17. We might write $X \subseteq_{fin} \mathsf{X}$ for $X \in FinPow(\mathsf{X})$.

Lemma 2.18. *If $X \subseteq_{fin} \mathsf{X}$ then:*

(1) $\bigcup\{supp(x) \mid x \in X\}$ *is finite.*
(2) $\bigcup\{supp(x) \mid x \in X\} = supp(X)$.
(3) $x \in X$ *implies* $supp(x) \subseteq supp(X)$.

Rewriting this using Notation 2.11: if X is finite and $x \in X$ then $a\#X$ implies $a\#x$.[5]

Proof. The first part is immediate since by assumption there is some finite $A \subseteq \mathbb{A}$ that bounds $supp(x)$ for all $x \in X$. The second part follows by an easy calculation using part 3 of Corollary 2.13; full details are in [Gab11, Theorem 2.29], of which Lemma 2.18 is a special case. Part 3 follows from the first and second parts. □

2.2.7. *Atoms-abstraction.* Atoms-abstraction was the first real application of nominal techniques; it was used to build inductive datatypes of syntax-with-binding. Nominal atoms-abstraction captures the essence of α-binding. In this paper we use it to model the binding in universal quantification and sets comprehension (see Definition 3.3). The maths here goes back to [Gab01, GP01]; we give references to proofs in a more recent presentation [Gab11].

Assume a nominal set X and an $i \in \mathbb{Z}$.

Definition 2.19. Let the **atoms-abstraction** set $[\mathbb{A}^i]\mathsf{X}$ have

- Underlying set $\{[a]x \mid a \in \mathbb{A}^i,\ x \in \mathsf{X}\}$ where $[a]x = \{(\pi(a), \pi{\cdot}x) \mid \pi \in fix(supp(x) \backslash \{a\})\}$.
- Permutation action $\pi{\cdot}[a]x = [\pi{\cdot}a]\pi{\cdot}x$.

Lemma 2.20. *If $x \in \mathsf{X}$ and $a \in \mathbb{A}^i$ then $supp([a]x) = supp(x) \backslash \{a\}$. In particular $a\#[a]x$ (Notation 2.11).*

Proof. See [Gab11, Theorem 3.11]. □

Lemma 2.21. *Suppose $x \in \mathsf{X}$ and $a, b \in \mathbb{A}^i$. Then if $b\#x$ then $[a]x = [b](b\ a){\cdot}x$.*

Proof. See [Gab11, Lemma 3.12]. □

Definition 2.22. Suppose $z \in [\mathbb{A}^i]\mathsf{X}$ and $b \in \mathbb{A}^i$. Write $z@b$ for the unique $x \in \mathsf{X}$ such that $z = [b]x$, if this exists.

Lemma 2.23. *Suppose $b \in \mathbb{A}^i$ and $z \in [\mathbb{A}^i]\mathsf{X}$. Then $b\#z$ implies $z@b \in \mathsf{X}$ is well-defined.*

Proof. See [Gab11, Theorem 3.19]. □

Lemma 2.24. *Suppose $a \in \mathbb{A}^i$ and $x \in \mathsf{X}$ and $z \in [\mathbb{A}^i]\mathsf{X}$. Then:*

(1) $([a]x)@a = x$ *and if* $b\#x$ *then* $([a]x)@b = (b\ a){\cdot}x$.
(2) *If* $a\#z$ *then* $[a](z@a) = z$.

Proof. See [Gab11, Theorem 3.19]. □

[5] This is not necessarily true if X is infinite. For instance if we take $\mathsf{X} = \mathbb{A} = X$ then the reader can verify that $a\#X$ for every a, but $a\#a$ does not hold for any $a \in \mathbb{A}$. This is a feature of nominal techniques, not a bug; but for the case of finite sets, things are simpler.

2.3. **The principle of equivariance.**

Remark 2.25. We now come to the *principle of equivariance* (Theorem 2.26; see also [Gab11, Subsection 4.2] and [GP01, Lemma 4.7]). It enables a particularly efficient management of renaming and α-conversion in syntax and semantics and captures why it is so useful to use *names* to model them instead of, for instance, numbers.

In a nutshell we can say

Atoms are distinguishable, but interchangable.

and we make this formal as follows:

Theorem 2.26. *Suppose $\overline{x}$ is a list $x_1, \ldots, x_n$. Suppose π is a (not necessarily finite) permutation and write $\pi{\cdot}\overline{x}$ for $\pi{\cdot}x_1, \ldots, \pi{\cdot}x_n$. Suppose $\Phi(\overline{x})$ is a first-order logic predicate in the language of ZFA*[6] *with free variables $\overline{x}$. Suppose $\Upsilon(\overline{x})$ is a function specified using a first-order predicate in the language of ZFA with free variables $\overline{x}$.*

Then we have the following principles:

(1) **Equivariance of predicates.** $\Phi(\overline{x}) \Leftrightarrow \Phi(\pi{\cdot}\overline{x})$.[7]
(2) **Equivariance of functions.** $\pi{\cdot}\Upsilon(\overline{x}) = \Upsilon(\pi{\cdot}\overline{x})$.
(3) **Conservation of support.** *If $\overline{x}$ denotes elements with finite support then $supp(\Upsilon(\overline{x})) \subseteq supp(x_1) \cup \cdots \cup supp(x_n)$.*

Proof. See Theorem 4.4, Corollary 4.6, and Theorem 4.7 from [Gab11]. □

Remark 2.27. Theorem 2.26 states that atoms can be permuted in our theorems and lemmas provided we do so consistently in all parameters. So for instance if we have proved $\phi(a, b, c)$, then

- taking $\pi = (a\, c)$ we also know $\phi(c, b, a)$ and
- taking $\pi = (a\, a')(b\, b')(c\, c')$ we also know $\phi(a', b', c')$, but
- we do not necessarily know that we can deduce $\phi(a, b, a)$ (depending on ϕ this may still hold, of course, but not by equivariance since no permutation takes (a, b, c) to (a, b, a)).

Equivariance makes explicit a sense in which atoms have a dual nature: individually, atoms behave like pointers to themselves,[8] but collectively they have the flavour of variables ranging over the set of all atoms via the action of permutations.[9] See also the permutative convention from Definition 2.1.

We will use Theorem 2.26 frequently in this paper, either to move permutations around (parts 1 and 2) or to get 'free' bounds on the support of elements (part 3). 'Free' here means 'from the form of the definition, without having to verify it by calculations'. Theorem 2.26 is 'free' in the spirit of Wadler's marvellously titled *Theorems for free!* [Wad89].[10]

Discussions expanding on this remark are in [Gab17] (full paper) and [Gab18] (abstract).

---

[6] First-order logic with equality $=$, sets membership $\in$, and a constant or collection of constants for sets of atoms.

[7] It is important to realise here that $\overline{x}$ must contain *all* the variables mentioned in the predicate. It is not the case that $a = a$ if and only if $a = b$ — but it is the case that $a = b$ if and only if $b = a$ (both are false).

[8] In the implementation of FM set theory in my PhD thesis [GP01] this was literally true: I found it convenient to use *Quine atoms*, meaning that $a = \{a\}$.

[9] This too can be made precise. See Subsection 2.6 and Lemma 4.17 of [DG12b].

[10] Finally, we can be somewhat more precise about the effort these free equivariance deductions can save: With equivariance, the cost of deducing $\phi(\pi{\cdot}x, \pi{\cdot}y, \pi{\cdot}z)$, given a deduction of $\phi(x, y, z)$, is 1. Without equivariance, the cost of deducing $\phi(\pi{\cdot}x, \pi{\cdot}y, \pi{\cdot}z)$, given a deduction of $\phi(x, y, z)$, is roughly n where n is the cost of deducing $\phi(x, y, z)$. This is convenient in a rigorous but unmechanised proof such as the one in this paper; in an implementation it can quadratically reduce effort by saving roughly effort n for each ϕ. This is the difference between α-equivalence and renaming lemmas being a minor consideration, and them inflating to dominate the development. My feeling is that once renaming lemmas consume more than 80% of the developmental effort, development stalls.

$$\frac{a \in \mathbb{A}^i}{\mathtt{atm}(a) \in \mathsf{Set}^i(\kappa)} \qquad \frac{\mathcal{X} \subseteq_{\mathit{fin}} \mathsf{Pred}(\kappa)}{\mathtt{and}(\mathcal{X}) \in \mathsf{Pred}(\kappa{+}1)} \qquad \frac{X{\in}\mathsf{Pred}(\kappa)}{\mathtt{neg}(X) \in \mathsf{Pred}(\kappa{+}1)}$$

$$\frac{X{\in}\mathsf{Pred}(\kappa)\ \ a{\in}\mathbb{A}^i}{\mathtt{all}([a]X){\in}\mathsf{Pred}(\kappa{+}1)} \qquad \frac{a{\in}\mathbb{A}^{i+1}\ \ x{\in}\mathsf{Set}^i(\kappa)}{\mathtt{elt}(x,a) \in \mathsf{Pred}(\kappa{+}1)} \qquad \frac{X{\in}\mathsf{Pred}(\kappa)\ \ a{\in}\mathbb{A}^{i\text{-}1}}{\mathtt{st}([a]X){\in}\mathsf{Set}^i(\kappa{+}1)}$$

Figure 1: Syntax of internal predicates and terms

Proposition 2.28. (1) *$supp(\pi{\cdot}x) = \pi{\cdot}supp(x)$ (which means $\{\pi(a) \mid a \in supp(x)\}$).*
(2) *$a\#\pi{\cdot}x$ (Notation 2.11) if and only if $\pi^{-1}(a)\#x$, and $a\#x$ if and only if $\pi(a)\#\pi{\cdot}x$.*

Proof. Immediate consequence of part 2 of Theorem 2.26 (for the 'not-free' proof by concrete calculations see [Gab11, Theorem 2.19]). □

3. Internal syntax

3.1. Basic definition.

Remark 3.1. We are now ready to to define our syntax (Figure 1) and study its basic properties (with more advanced properties considered in Section 4).

Figure 1 defines a *nominal* datatype, in which atoms-abstraction is used to manage binding, as introduced in [GP01]. This gives us Lemma 3.6.

(1) Parts 1 and 3 of Lemma 3.6 say "We can alpha-convert".
(2) In part 2 of Lemma 3.6, *supp* corresponds exactly to the notion that would normally be written "Free variables of", and $a\#X$ corresponds to "a is not free in X".

So why not just write that? Nominal techniques are a general basket of ideas with implications that go well beyond modelling syntax, but the specific benefit of using nominal techniques to model syntax is that we get alpha-conversion for free from the ambient nominal theory (see [GP01] and Section 2). We do not have to define α-conversion and free variables of by induction, and then prove their properties (which is actually a more subtle undertaking than is often realised; cf. Remark 4.14).

The reader does not expect to see notions of ordered pairs, trees, numbers, functions, and function application developed from first principles every time we want to write abstract syntax and write a function on a syntax tree. It is assumed that these things have been worked out. Nominal techniques do that for binding (and more).

Notation 3.2. Write $\mathbb{Z}$ for the **integers**, so $\mathbb{Z} = \{0, 1, \text{-}1, 2, \text{-}2, \dots\}$ and $\mathbb{N}$ for the **natural numbers**, which we start at 0, so $\mathbb{N} = \{0, 1, 2, \dots\}$.

Definition 3.3. (1) Define datatypes
- Pred of **internal predicates** and
- Set^i for $i{\in}\mathbb{Z}$ of **internal (level i) sets**

inductively by the rules in Figure 1, where κ ranges over finite ordinals.
(2) Define
$$\mathsf{Pred} = \bigcup_\kappa \mathsf{Pred}(\kappa) \quad \text{and} \quad \mathsf{Set}^i = \bigcup_\kappa \mathsf{Set}^i(\kappa).$$
(3) Write $age(X)$ for the least κ such that $X{\in}\mathsf{Pred}(\kappa)$.
(4) Write $age(x)$ for the least κ such that $x{\in}\mathsf{Set}^i(\kappa)$.

Notation 3.4. • If $a{\in}\mathbb{A}$ we may call $\mathtt{atm}(a)$ an **internal atom**.
• If $X{\in}\mathsf{Pred}$ we may call $\mathtt{st}([a]X)$ an **internal comprehension**.
• We may call $\mathtt{atm}(a)$ or $\mathtt{st}([a]X)$ an **internal set**.[11]

Remark 3.5. We read through and comment on Definition 3.3:

(1) κ measures the *age* or *stage* of an element; at what point in the induction it is introduced into the datatype. This is an inductive measure.
(2) If we elide κ and levels and simplify, we can rewrite Definition 3.3 semi-formally as follows:

$$\begin{array}{rl} x \in \mathsf{Set} & ::= \mathtt{atm}(a) \mid \mathtt{st}([a]X) \\ X \in \mathsf{Pred} & ::= \mathtt{and}(\mathcal{X}) \mid \mathtt{neg}(X) \mid \mathtt{all}([a]X) \mid \mathtt{elt}(x,a) \end{array}$$

(3) `neg` represents negation. `and` represents logical conjunction.
(4) `and` takes a finite set rather than a pair of terms. This is a nonessential eccentricity that cuts down on cases later on. Truth is represented as $\mathtt{and}(\varnothing)$. See Example 3.9.
(5) `all` represents universal quantification; read $\mathtt{all}([a]X)$ as 'for all a, X' or in symbols: '$\forall a.\phi$'. In $\mathtt{all}([a]X)$, $[a]X$ is the nominal atoms-abstraction from Definition 2.19. It implements the binding of the universal quantifier by the standard nominal method.
(6) $\in$ represents a sets membership; read $\mathtt{elt}(x,a)$ as 'x is an element of a'. Note here that a is an atom; it does not literally have any elements. $\mathtt{elt}(x,a)$ represents the predicate 'we believe that x is an element of the variable a', or in symbols: '$x \in a$'.
(7) $\mathtt{st}([a]X)$ represents sets comprehension; read $\mathtt{st}([a]X)$ as 'the set of a such that X' or in symbols: '$\{a \mid X\}$'. Again, as standard in nominal techniques, nominal atoms-abstraction is used to represent the binding.
    If $a{\in}\mathbb{A}^i$ then $\mathtt{st}([a]X){\in}\mathsf{Set}^{i+1}$.
(8) $\mathtt{atm}(a)$ is a copy of $a{\in}\mathbb{A}$ wrapped in some formal syntax `atm`.

Lemma 3.6. *Suppose* $X{\in}\mathsf{Pred}$ *and* $i{\in}\mathbb{Z}$ *and* $a, a'{\in}\mathbb{A}^i$ *and* $a'\#X$. *Then:*

(1) $\mathtt{st}([a]X) = \mathtt{st}([a'](a'\ a){\cdot}X)$ *and* $\mathtt{all}([a]X) = \mathtt{all}([a'](a'\ a){\cdot}X)$.
(2) $a\#\mathtt{st}([a]X)$ *and* $a\#\mathtt{all}([a]X)$, *and* $\mathit{supp}(\mathtt{st}([a]X)), \mathit{supp}(\mathtt{all}([a]X)) \subseteq \mathit{supp}(X)\backslash\{a\}$.
(3) *For every finite* $S \subseteq_{\mathit{fin}} \mathbb{A}^i$ *there exist* $b \in \mathbb{A}^i\backslash S$ *and* $Y{\in}\mathsf{Pred}$ *such that* $\mathtt{st}([a]X) = \mathtt{st}([b]Y)$ *and* $\mathtt{all}([a]X) = \mathtt{all}([b]Y)$.

Proof. Immediate from Lemma 2.21, and Lemma 2.20 with Theorem 2.26. □

Lemma 3.7. *Suppose* $i{\in}\mathbb{Z}$ *and* $a, b \in \mathbb{A}^i$ *and* $X \in \mathsf{Pred}$ *and* $x \in \mathsf{Set}$. *Then* $\mathit{age}(X) = \mathit{age}((a\ b){\cdot}X)$ *and* $\mathit{age}(x) = \mathit{age}((a\ b){\cdot}x)$.

Proof. Direct from Theorem 2.26(2). □

3.2. **Some notation.**

Notation 3.8. Suppose $X, Y{\in}\mathsf{Pred}$ and $\mathcal{X} \subseteq_{\mathit{fin}} \mathsf{Pred}$. Define syntactic sugar $\mathtt{or}(\mathcal{X})$, $\mathtt{imp}(X,Y)$ and $\mathtt{iff}(X,Y)$ by

$$\begin{array}{rl} \mathtt{or}(\mathcal{X}) & = \mathtt{neg}((\mathtt{and}(\{\mathtt{neg}(X) \mid X \in \mathcal{X}\}))) \\ \mathtt{imp}(X,Y) & = \mathtt{or}(\{\mathtt{neg}(X), Y\}) \\ \mathtt{iff}(X,Y) & = \mathtt{and}(\{\mathtt{imp}(X,Y), \mathtt{imp}(Y,X)\}). \end{array}$$

[11] So every internal comprehension or internal atom is an internal set. Another choice of terminology would be to call $\mathtt{atm}(a)$ an internal atom, $\mathtt{st}([a]X)$ an internal set, and $\mathtt{atm}(a)$ or $\mathtt{st}([a]X)$ *internal elements*.
However, note that $\mathtt{st}([a]X)$ is not a set and neither is $\mathtt{atm}(a)$; they are both syntax and we can call them what we like.

Example 3.9. Define $\mathtt{F} \in \mathsf{Pred}$ and $\mathtt{T} \in \mathsf{Pred}$ by

$$\mathtt{F} = \mathtt{or}(\varnothing) \quad \text{and} \quad \mathtt{T} = \mathtt{and}(\varnothing).$$

Intuitively, $\mathtt{F}$ represents the empty disjunction, and $\mathtt{T}$ represents the empty conjunction.

Notation 3.10. Suppose that:

- $i{\in}\mathbb{Z}$ and
- $x = \mathtt{st}(x'){\in}\mathsf{Set}^i$ is an internal comprehension where $x' \in [\mathbb{A}^{i\text{-}1}]\mathsf{Pred}$ and $a'{\in}\mathbb{A}^{i\text{-}1}$ and
- $a'\#x$ (equivalently[12] $a'\#x'$).

Then write

$$x@a' \quad \text{for} \quad x'@a'.$$

Lemma 3.11 checks that Notation 3.10 makes sense:

Lemma 3.11. *Suppose $i{\in}\mathbb{Z}$ and $a'{\in}\mathbb{A}^{i\text{-}1}$. Suppose $x{\in}\mathsf{Set}^i$ is an internal comprehension and $a'\#x$ and $X' \in \mathsf{Pred}$. Then:*

(1) *$x@a'$ is well-defined and $x@a' \in \mathsf{Pred}$.*
(2) *$x = \mathtt{st}([a'](x@a'))$.*
(3) *If $\mathit{age}(x) = \kappa{+}1$ then $\mathit{age}(x@a') = \kappa$ (meaning that in an inductive argument using age, taking $x@a'$ strictly decreases the inductive measure).*
(4) *$(\mathtt{st}([a']X'))@a' = X'$.*

Proof. (1) By construction and Lemma 2.23.
(2) By construction and Lemma 2.24(2).
(3) By construction and Lemma 3.7.
(4) By construction and Lemma 2.24(1). □

Recall $\mathtt{F}{=}\mathtt{or}(\varnothing)$ from Example 3.9.

Definition 3.12. Suppose $i{\in}\mathbb{Z}$ and $a{\in}\mathbb{A}^{i\text{-}1}$. Define $\mathtt{empt}^i$ and $\mathtt{set}^i$ by

$$\mathtt{empt}^i = \mathtt{st}([a]\mathtt{F}) = \mathtt{st}([a]\mathtt{or}(\varnothing))$$
$$\mathtt{set}^i = \mathtt{st}([a]\mathtt{T}) = \mathtt{st}([a]\mathtt{and}(\varnothing)).$$

We conclude with an easy lemma:

Lemma 3.13. *Suppose $i{\in}\mathbb{Z}$ and $a{\in}\mathbb{A}^{i\text{-}1}$. Then:*

(1) *$\mathtt{empt}^i@a = \mathtt{F}$ and $\mathtt{st}([a]\mathtt{F}) = \mathtt{empt}^i$, and similarly $\mathtt{set}^i@a = \mathtt{T}$ and $\mathtt{st}([a]\mathtt{T}) = \mathtt{set}^i$.*
(2) *$a\#\mathtt{F}$ and $a\#\mathtt{T}$.*
(3) *Definition 3.12 does not depend on the choice of $a{\in}\mathbb{A}^{i\text{-}1}$.*

Proof. (1) From Definition 3.12 and Lemma 2.24(1).
(2) From part 1 of this result, since $a\#\mathtt{F}$ and $a\#\mathtt{T}$ by Theorem 2.26.
(3) From part 2 of this result, using Corollary 2.13. □

[12] By concrete calculations or by Theorem 2.26.

$$
\begin{array}{lll}
(\sigma\mathtt{and}) & & (\mathtt{and}(\mathcal{X}))[a{\mapsto}x] = \mathtt{and}(\{X[a{\mapsto}x] \mid X{\in}\mathcal{X}\}) \\
(\sigma\mathtt{neg}) & & (\mathtt{neg}(X))[a{\mapsto}x] = \mathtt{neg}((X[a{\mapsto}x])) \\
(\sigma\mathtt{all}) & b\#x \Rightarrow & (\mathtt{all}([b]X))[a{\mapsto}x] = \mathtt{all}([b](X[a{\mapsto}x])) \\
(\sigma\mathtt{eltatm}) & & (\mathtt{elt}(y,a))[a{\mapsto}\mathtt{atm}(n)] = \mathtt{elt}(y[a{\mapsto}\mathtt{atm}(n)],n) \\
(\sigma\mathtt{elta}) & & (\mathtt{elt}(y,a))[a{\mapsto}\mathtt{st}([a']X')] = X[a'{\mapsto}y[a{\mapsto}\mathtt{st}([a']X')]] \\
(\sigma\mathtt{eltb}) & & (\mathtt{elt}(y,b))[a{\mapsto}x] = \mathtt{elt}(y[a{\mapsto}x],b) \\
(\sigma\mathbf{a}) & & \mathtt{atm}(a)[a{\mapsto}x] = x \\
(\sigma\mathbf{b}) & & \mathtt{atm}(b)[a{\mapsto}x] = \mathtt{atm}(b) \\
(\sigma\mathtt{st}) & c\#x \Rightarrow & \mathtt{st}([c]X)[a{\mapsto}x] = \mathtt{st}([c](X[a{\mapsto}x]))
\end{array}
$$

Figure 2: The sigma-action (Definition 4.1)

4. The sigma-action

4.1. **Basic definition and well-definedness.** Intuitively, Definition 4.1 defines a substitution action. It is slightly elaborate, especially because of $(\sigma\mathtt{elta})$ of Figure 2, so it gets a fancy name ('σ-action') and we need to make formal and verify that it behaves as a substitution action should; see Remark 4.7.

Definition 4.1. Define a **σ-action** (sigma-action) to be a family of functions

$$\sigma_i : \mathsf{Pred} \times \mathbb{A}^i \times \mathsf{Set}^i \to \mathsf{Pred} \quad \text{and} \quad \sigma_{ij} : \mathsf{Set}^j \times \mathbb{A}^i \times \mathsf{Set}^i \to \mathsf{Set}^j$$

where $i,j{\in}\mathbb{Z}$, inductively by the rules in Figure 2. For readability we write

$$\sigma_i(Z,a,x) \text{ as } Z[a{\mapsto}x] \quad \text{and} \quad \sigma_i(z,a,x) \text{ as } z[a{\mapsto}x].$$

Furthermore in Figure 2:

- In rule $(\sigma\mathtt{and})$, $\mathcal{X}{\subseteq_{\mathit{fin}}}\mathsf{Pred}$.
- In rule $(\sigma\mathtt{neg})$, $X{\in}\mathsf{Pred}$.
- In rule $(\sigma\mathtt{all})$, $X{\in}\mathsf{Pred}$ and $b{\in}\mathbb{A}^j$ for some $j{\in}\mathbb{Z}$.
- In rule $(\sigma\mathtt{elta})$, $a'{\in}\mathbb{A}^{i\text{-}1}$.
- In rule $(\sigma\mathtt{eltatm})$, n ranges over *all* atoms in $\mathbb{A}^i$ (not just those distinct from a).
- In rule $(\sigma\mathtt{eltb})$, $b{\in}\mathbb{A}^j$ for some $j{\in}\mathbb{Z}$.
- In rule $(\sigma\mathtt{st})$, $X{\in}\mathsf{Pred}$ and $c{\in}\mathbb{A}^k$ for some $k{\in}\mathbb{Z}$.

Remark 4.2. Figure 2 slips in no fewer than three abuses of the mathematics:

(1) We do not know that $X \in \mathsf{Pred}$ implies $X[a{\mapsto}x] \in \mathsf{Pred}$, so we should not write $\mathtt{and}(\{X[a{\mapsto}x] \mid \ldots\})$ on the right-hand side of $(\sigma\mathtt{and})$, or indeed $X[a{\mapsto}x]$ on the right-hand side of $(\sigma\mathtt{neg})$, and so on.

In fact, all right-hand sides of Figure 2 are suspect except those of $(\sigma\mathbf{a})$ and $(\sigma\mathbf{b})$.

(2) We do not know whether the choice of fresh $a'{\in}\mathbb{A}^{i\text{-}1}$ in $(\sigma\mathtt{elta})$ matters, so we do not know that $(\sigma\mathtt{elta})$ is well-defined.

(3) The definition looks inductive at first glance, however in the case of $(\sigma\mathtt{elta})$ there is no guarantee that X (on the right-hand side) is smaller than $\mathtt{elt}(y,a)$ (on the left-hand side). The level of a' is strictly lower than the level of a, however levels are taken from $\mathbb{Z}$ which is totally ordered but not well-ordered by $\leq$.

In fact:

- $X \in \mathsf{Pred}$ does indeed imply $X[a{\mapsto}x] \in \mathsf{Pred}$.
- The choice of fresh a' in $(\sigma\mathtt{elta})$ is immaterial.

- The levels of atoms involved are bounded below (see Definition 4.4) so we only ever work on a well-founded fragment of $\mathbb{Z}$.

For proofs see Proposition 4.6 and Lemma 4.8.

Would it be more rigorous to interleave the proofs of these lemmas with the definition, so that at each stage we are confident that what we are writing actually makes sense? Certainly we could; the reader inclined to worry about this need only read Definition 4.1 alongside Proposition 4.6 and Lemma 4.8 as a simultaneous inductive argument.

Remark 4.3 (Why 'minimum level')**.** Levels are in $\mathbb{Z}$ and are totally ordered by $\leq$ but not well-founded (since integers can 'go downwards forever').

However, any (finite) internal predicate or internal set can mention only finitely many levels, so we can calculate the *minimum level* of a predicate or set, which is lower bound on the levels of atoms appearing in that predicate or set. We will use this lower bound to reason inductively on levels in Propositions 4.6 and 4.13.

Definition 4.4. Define $minlevel(Z)$ and $minlevel(z)$ the **minimum level** of Z or z, inductively on $Z{\in}\mathsf{Pred}$ and $z{\in}\mathsf{Set}^i$ for $i{\in}\mathbb{Z}$ as follows:

$$\begin{array}{ll} minlevel(\mathtt{atm}(a)) & = level(a) \\ minlevel(\mathtt{and}(\mathcal{X})) & = min(\{0\} \cup \{minlevel(X) \mid X{\in}\mathcal{X}\}) \\ minlevel(\mathtt{neg}(X)) & = minlevel(X) \\ minlevel(\mathtt{all}([a]X)) & = min(\{level(a), minlevel(X)\}) \\ minlevel(\mathtt{elt}(x,a)) & = min(\{minlevel(x), level(a)\}) \\ minlevel(\mathtt{st}([a]X)) & = min(\{level(a), minlevel(X)\}) \end{array}$$

Above, $min(\mathcal{I})$ is the least element of $\mathcal{I} \subseteq_{fin} \mathbb{Z}$. We add 0 in the clause for `and` as a 'default value' to exclude calculating a minimum for the empty set; any other fixed integer element would do as well or, if we do not want to make this choice, we can index $minlevel$ over a fixed but arbitrary choice. The proofs to follow will not care.

It will be convenient to apply $minlevel$ to a mixed list of internal predicates, atoms, and internal sets:

Notation 4.5. • Define $minlevel(a) = level(a)$.
- If $l = (l_1, l_2, \ldots, l_n)$ is a list of elements from $\mathsf{Pred} \cup \bigcup_{i\in\mathbb{Z}} \mathsf{Set}^i \cup \mathbb{A}$ then we write $minlevel(l)$ for the least element of $\{minlevel(l_1), \ldots, minlevel(l_n)\}$.

Proposition 4.6. *Suppose $i{\in}\mathbb{Z}$ and $a{\in}\mathbb{A}^i$ and $x{\in}\mathsf{Set}^i$.*

(1) *If $Z{\in}\mathsf{Pred}$ then*
- *$Z[a{\mapsto}x]$ is well-defined,*
- *$minlevel(Z[a{\mapsto}x]){\geq}minlevel(Z,a,x)$, and*
- *$Z[a{\mapsto}x]{\in}\mathsf{Pred}$.*

(2) *If $k{\in}\mathbb{Z}$ and $z{\in}\mathsf{Set}^k$ then*
- *$z[a{\mapsto}x]$ is well-defined,*
- *$minlevel(z[a{\mapsto}x]){\geq}minlevel(z,a,x)$, and*
- *$z[a{\mapsto}x]{\in}\mathsf{Set}^k$.*

Proof. Fix some $k{\in}\mathbb{Z}$. We prove the Proposition for all Z, a, x and z, a, x with $minlevel(Z,a,x){\geq}k$ and $minlevel(z,a,x){\geq}k$, by induction on

$$(level(a), age(Z)) \quad \text{and} \quad (level(a), age(z))$$

lexicographically ordered. Since k was arbitrary, this suffices to prove it for all Z, a, x and z, a, x.

We consider the possibilities for $Z{\in}\mathsf{Pred}$:

- *The case of* $\mathtt{and}(\mathcal{X})$ *for* $\mathcal{X} \subseteq_{fin} \mathsf{Pred}$.
  By Figure 2 ($\sigma\mathtt{and}$) $Z[a{\mapsto}x]{=}\mathtt{and}(\{X'[a{\mapsto}x]|X'{\in}\mathcal{X}\})$. We use the inductive hypothesis on each $X'[a{\mapsto}x]$ and some easy arithmetic calculations.
- *The case of* $\mathtt{neg}(X')$ *for* $X'{\in}\mathsf{Pred}$.
  By Figure 2 ($\sigma\mathtt{neg}$) $Z[a{\mapsto}x] = \mathtt{neg}((X'[a{\mapsto}x]))$. We use the inductive hypothesis on $X'[a{\mapsto}x]$.
- *The case of* $\mathtt{all}([b]X')$ *for* $X'{\in}\mathsf{Pred}$ *and* $b{\in}\mathbb{A}^j$ *for some* $j{\in}\mathbb{Z}$.
  Using Lemma 3.6(1) we may assume without loss of generality that $b\#x$. By Figure 2 ($\sigma\mathtt{all}$) $(\mathtt{all}([b]X'))[a{\mapsto}x] = \mathtt{all}([b'](X[a{\mapsto}x]))$. We use the inductive hypothesis on $X'[a{\mapsto}x]$.
- *The case of* $\mathtt{elt}(z,a)$ *for* $z{\in}\mathsf{Set}^{i\text{-}1}$. There are two sub-cases:
  - *Suppose* $x{=}\mathtt{atm}(n)$ *for some* $n{\in}\mathbb{A}^i$.
    By Figure 2 ($\sigma\mathtt{eltatm}$) $(\mathtt{elt}(z,a))[a{\mapsto}x] = \mathtt{elt}(z[a{\mapsto}\mathtt{atm}(n)],n)$. We use the inductive hypothesis on $z[a{\mapsto}\mathtt{atm}(n)]$.
  - *Suppose* $x{=}\mathtt{st}([a']X')$ *where* $X' = x@a'$ *for some fresh* $a'{\in}\mathbb{A}^{i\text{-}1}$ *(so* $a'\#x, z$*).*
    By Figure 2 ($\sigma\mathbf{elta}$) $(\mathtt{elt}(z,a))[a{\mapsto}x] = X'[a'{\mapsto}z[a{\mapsto}x]]$. We have the inductive hypothesis on $z[a{\mapsto}x]$. We also have the inductive hypothesis (since $k{\leq}level(a'){=}i\text{-}1 \lneq i{=}level(a)$)[13] on $X'[a'{\mapsto}z[a{\mapsto}x]]$, and this suffices.
- *The case of* $\mathtt{elt}(z,c)$ *where* $c{\in}\mathbb{A}^k$ *and* $z{\in}\mathsf{Set}^{k\text{-}1}$ *and* $k{\in}\mathbb{Z}$.
  By Figure 2 ($\sigma\mathbf{eltb}$) and the inductive hypothesis.

We consider the possibilities for $z{\in}\mathsf{Set}^k$:

- *The case that* z *is an internal atom.*
  We use ($\sigma\mathbf{a}$) or ($\sigma\mathbf{b}$) of Figure 2.
- *The case that* z *is an internal comprehension.*
  Choose fresh $c{\in}\mathbb{A}^{k\text{-}1}$ (so $c\#x, z$), so that by Lemma 3.11(2) $z = \mathtt{st}([c]z@c)$. We use the first part of this result and Figure 2 ($\sigma\mathtt{st}$). □

4.2. **Nominal algebraic properties of the sigma-action.**

Remark 4.7. Several useful properties of the σ-action from Definition 4.1 are naturally expressed as nominal algebra judgements — equalities subject to freshness conditions [GM09]. Some are listed for the reader's convenience in Figure 3, which goes back to nominal axiomatic studies of substitution from [GM06, GM08].

In this paper we are dealing with a concrete model, so the judgements in Figure 3 are not assumed and are not axioms. Instead they must be proved; they are propositions and lemmas:

- ($\sigma\alpha$) is Lemma 4.8.
- ($\sigma\#$) is Lemma 4.10.
- ($\sigma\sigma$) is Proposition 4.13.
- ($\sigma\mathbf{swp}$) and ($\sigma\mathbf{asc}$) are Corollaries 4.15 and 4.16.
- ($\sigma\mathbf{id}$) is Lemma 4.17.
- ($\sigma\mathbf{ren}$) is Lemma 4.18.
- ($\sigma@$) is Lemma 4.12.

These are familiar properties of substitution on syntax: for instance

- ($\sigma\alpha$) looks like an α-equivalence property — and indeed it is — and

[13] k was chosen no greater than the minimum level of Z, a, and x. Now $x = \mathtt{st}([a']X')$, and it follows from Definition 4.4 that $k \leq level(a')$.

$$
\begin{array}{lrl}
(\sigma\alpha) & b'\#Z \Rightarrow & Z[b{\mapsto}y] = ((b'\ b){\cdot}Z)[b'{\mapsto}y] \\
(\sigma\#) & b\#Z \Rightarrow & Z[b{\mapsto}y] = Z \\
(\sigma\sigma) & a\#y \Rightarrow & Z[a{\mapsto}x][b{\mapsto}y] = Z[b{\mapsto}y][a{\mapsto}x[b{\mapsto}y]] \\
(\sigma\mathbf{swp}) & a\#y,\ b\#x \Rightarrow & Z[a{\mapsto}x][b{\mapsto}y] = Z[b{\mapsto}y][a{\mapsto}x] \\
(\sigma\mathbf{asc}) & a\#y,\ b\#Z \Rightarrow & Z[a{\mapsto}x[b{\mapsto}y]] = Z[a{\mapsto}x][b{\mapsto}y] \\
(\sigma\mathbf{id}) & & Z[a{\mapsto}\mathtt{atm}(a)] = Z \\
(\sigma\mathbf{ren}) & a'\#Z \Rightarrow & Z[a{\mapsto}\mathtt{atm}(a')] = (a'\ a){\cdot}Z \\
(\sigma@) & c\#x \Rightarrow & (z@c)[a{\mapsto}x] = z[a{\mapsto}x]@c
\end{array}
$$

Figure 3: Further nominal algebra properties of the σ-action

- $(\sigma\#)$ (Lemma 4.10) is sometimes called *garbage collection* and corresponds to the property "if a is not free in t then $t[a{\mapsto}s] = t$", and
- $(\sigma\sigma)$ (Proposition 4.13) is often called the *substitution lemma*. See the discussion in Remark 4.14.

But, the proofs of these properties that we see in this paper are not replays of the familiar syntactic properties.

This is because the σ-action on Pred is not a simple 'tree-grafting' operation — not even a capture-avoiding one — because of $(\sigma\mathtt{elta})$ in Figure 2. The proofs work, but we cannot take this for granted, and they require checking.

4.2.1. *Alpha-equivalence of the sigma-action.*

Lemma 4.8 $((\sigma\alpha))$**.** *Suppose* $i{\in}\mathbb{Z}$ *and* $a, a'{\in}\mathbb{A}^i$ *and* $x{\in}\mathsf{Set}^i$*. Suppose* $Z{\in}\mathsf{Pred}$ *and* $a'\#Z$ *and* $z{\in}\mathsf{Set}^k$ *and* $a'\#z$*. Then:*

(1) $Z[a{\mapsto}x] = ((a'\ a){\cdot}Z)[a'{\mapsto}x]$ *and* $z[a{\mapsto}x] = ((a'\ a){\cdot}z)[a'{\mapsto}x]$.
(2) $supp(Z[a{\mapsto}x]) \subseteq (supp(Z){\setminus}\{a\}) \cup supp(x)$ *and* $supp(z[a{\mapsto}x]) \subseteq (supp(z){\setminus}\{a\}) \cup supp(x)$.
(3) *If* $a\#x$ *then* $a\#Z[a{\mapsto}x]$ *and* $a\#z[a{\mapsto}x]$.

Proof. By induction on

$$age(Z) \quad \text{and} \quad age(z).$$

We consider the possibilities for $Z{\in}\mathsf{Pred}$:

- *The case of* $\mathtt{and}(\mathcal{X})$ *for* $\mathcal{X} \subseteq_{fin} \mathsf{Pred}$. By Lemma 2.18 $a\#X'$ for every $X'{\in}\mathcal{X}$, so by the inductive hypothesis $X'[a{\mapsto}x] = ((a'\ a){\cdot}X')[a'{\mapsto}x]$. We use Figure 2 $(\sigma\mathtt{and})$ and Theorem 2.26.
- *The case of* $\mathtt{neg}(X)$ *for* $X{\in}\mathsf{Pred}$. By Figure 2 $(\sigma\mathtt{neg})$ and the inductive hypothesis for X.
- *The case of* $\mathtt{all}([b]X)$ *for* $X{\in}\mathsf{Pred}$ *and* $b{\in}\mathbb{A}^j$ *for some* $j{\in}\mathbb{Z}$. Using Lemma 3.6(1) we may assume without loss of generality that $b\#x$. We use Figure 2 $(\sigma\mathtt{all})$ and the inductive hypothesis.
- *The case of* $\mathtt{elt}(y, a)$ *for some* $y{\in}\mathsf{Set}^{i\text{-}1}$. There are two sub-cases:
  - *Suppose* $x{=}\mathtt{atm}(n)$ *for some* $n{\in}\mathbb{A}^i$. We reason as follows:

$$
\begin{array}{lll}
(\mathtt{elt}(y,a))[a{\mapsto}\mathtt{atm}(n)] = \mathtt{elt}(y[a{\mapsto}\mathtt{atm}(n)], n) & & \text{Figure 2}(\sigma\mathtt{eltatm}) \\
= \mathtt{elt}(((a'\ a){\cdot}y)[a'{\mapsto}\mathtt{atm}(n)], n) & & \text{Ind hyp for } y \\
= (\mathtt{elt}(((a'\ a){\cdot}y), a'))[a'{\mapsto}\mathtt{atm}(n)] & & \text{Figure 2}(\sigma\mathtt{eltatm}) \\
= ((a'\ a){\cdot}(\mathtt{elt}(y,a)))[a'{\mapsto}\mathtt{atm}(n)] & & \text{Theorem 2.26}
\end{array}
$$

– *Suppose* $x{=}\mathtt{st}([b']X')$ *where* $X' = x@b'$ *for some fresh* $b'{\in}\mathbb{A}^{i\text{-}1}$ *(so* $b'\#x, y, z$*).*
We reason as follows:

$$\begin{array}{rll}(\mathtt{elt}(y,a))[a{\mapsto}x] & = X'[b'{\mapsto}y[a{\mapsto}x]] & \text{Figure 2}(\sigma\mathbf{elta})\\ & = X'[b'{\mapsto}((a'\ a){\cdot}y)[a'{\mapsto}x]] & \text{Ind hyp for } y\\ & = (\mathtt{elt}((a'\ a){\cdot}y, a'))[a'{\mapsto}x] & \text{Figure 2}(\sigma\mathbf{elta})\end{array}$$

- *The case* $\mathtt{elt}(y,b)$ *for* $j{\in}\mathbb{Z}$ *and* $b{\in}\mathbb{A}^j$ *and* $y{\in}\mathsf{Set}^{j\text{-}1}$. We reason as follows:

$$\begin{array}{rll}(\mathtt{elt}(y,b))[a{\mapsto}x] & = \mathtt{elt}(y[a{\mapsto}x], b) & \text{Figure 2}(\sigma\mathbf{eltb})\\ & = \mathtt{elt}(((a'\ a){\cdot}y)[a'{\mapsto}x], b) & \text{Ind hyp for } y\\ & = (\mathtt{elt}(((a'\ a){\cdot}y), b))[a'{\mapsto}x] & \text{Figure 2}(\sigma\mathbf{eltb})\\ & = ((a'\ a){\cdot}(\mathtt{elt}(y,b)))[a'{\mapsto}x] & \text{Theorem 2.26}\end{array}$$

We consider the possibilities for $z{\in}\mathsf{Set}^k$:

- *The case that* z *is an internal atom.* We use $(\sigma\mathbf{a})$ or $(\sigma\mathbf{b})$ of Figure 2.
- *The case that* z *is an internal comprehension.* We use Lemma 3.11(2&3) for a fresh $c{\in}\mathbb{A}^{k\text{-}1}$ (so $c\#z$), $(\sigma\mathtt{st})$, and the inductive hypothesis.

  For part 2, we note that by Theorem 2.26 and Proposition 2.28

$$\begin{array}{c}supp(Z[a{\mapsto}x]) \subseteq supp(Z) \cup \{a\} \cup supp(x) \quad \text{and}\\ supp(((a'\ a){\cdot}Z)[a'{\mapsto}x]) \subseteq (a'\ a){\cdot}supp(Z) \cup \{a'\} \cup supp(x).\end{array}$$

We take a sets intersection. The case of z is similar.

Part 3 follows, recalling from Notation 2.11 that $a\#x$ means $a{\notin}supp(x)$. □

Remark 4.9. *Note for experts:* We could set Definition 4.1 up differently: we could have σ_i and σ_{ij} input abstractions

$$\sigma_i : ([\mathbb{A}^i]\mathsf{Pred}) \times \mathsf{Set}^i \to \mathsf{Pred} \quad \text{and} \quad \sigma_{ij} : ([\mathbb{A}^i]\mathsf{Set}^j) \times \mathsf{Set}^i \to \mathsf{Set}^j.$$

Then Lemma 4.8 would become immediate from Lemmas 2.20 and 2.21 and Theorem 2.26. This is nice but note that we incur a well-definedness proof-obligation that the choice of name for the abstracted atom does not matter. There is probably still a net gain but it is not quite as great as it might first seem. For this reason, we use the more elementary set-up in Definition 4.1.

4.2.2. *Property* $(\sigma\#)$ *(garbage collection).*

Lemma 4.10 $((\sigma\#))$**.** *Suppose* $i{\in}\mathbb{Z}$ *and* $a{\in}\mathbb{A}^i$ *and* $x{\in}\mathsf{Set}^i$ *and* $Z{\in}\mathsf{Pred}$ *and* $z{\in}\mathsf{Set}^k$ *for* $k{\in}\mathbb{Z}$*. Then*

$$\begin{array}{l}a\#Z \Rightarrow Z[a{\mapsto}x] = Z\\ a\#z \Rightarrow \ \ z[a{\mapsto}x] = z.\end{array}$$

Proof. By induction on

$$age(Z) \quad \text{and} \quad age(z).$$

We consider the possibilities for $Z{\in}\mathsf{Pred}$:

- *The case of* $\mathtt{and}(\mathcal{X})$ *for* $\mathcal{X} \subseteq_{fin} \mathsf{Pred}$.
  By Figure 2 $(\sigma\mathtt{and})$ $(\mathtt{and}(\mathcal{X}))[a{\mapsto}x] = \mathtt{and}(\{X[a{\mapsto}x] \mid X{\in}\mathcal{X}\})$. By Lemma 2.18(3) $a\#X$ for every $X{\in}\mathcal{X}$. We use the inductive hypothesis on each X.
- *The case of* $\mathtt{neg}(X)$ *for* $X{\in}\mathsf{Pred}$.
  By Figure 2 $(\sigma\mathtt{neg})$ $\mathtt{neg}(X)[a{\mapsto}x] = \mathtt{neg}(X[a{\mapsto}x])$. We use the inductive hypothesis on X.

- *The case of* $\mathtt{all}([b]X)$ *for* $X{\in}\mathsf{Pred}$ *and* $b{\in}\mathbb{A}^j$ *for some* $j{\in}\mathbb{Z}$.
  Using Lemma 3.6(1) we may assume without loss of generality that $b\#x$. By Figure 2 $(\sigma\mathtt{all})$ $(\mathtt{all}([b]X))[a{\mapsto}x] = \mathtt{all}([b](X[a{\mapsto}x]))$. We use the inductive hypothesis on X.
- *The case of* $\mathtt{elt}(y,a)$ *for* $i{\in}\mathbb{Z}$ *and* $y{\in}\mathsf{Set}^{i\text{-}1}$.
  This is impossible because we assumed $a\#Z$.
- *The case of* $\mathtt{elt}(y,b)$ *for* $j{\in}\mathbb{Z}$ *and* $b{\in}\mathbb{A}^j$ *and* $y{\in}\mathsf{Set}^{j\text{-}1}$.
  By Figure 2 $(\sigma\mathtt{eltb})$ $(\mathtt{elt}(y,b))[a{\mapsto}x] = \mathtt{elt}(y[a{\mapsto}x],b)$. We use the inductive hypothesis on y.

We consider the possibilities for $z{\in}\mathsf{Set}^k$:

- If z is an internal atom then we reason using $(\sigma\mathbf{a})$ or $(\sigma\mathbf{b})$ of Figure 2.
- If z is an internal comprehension then we use Lemma 3.11(2&3) for a fresh $c{\in}\mathbb{A}^{k\text{-}1}$ (so $c\#z$), $(\sigma\mathtt{st})$, and the inductive hypothesis. □

Recall $\mathtt{F}{=}\mathtt{or}(\varnothing)$ and $\mathtt{T}{=}\mathtt{and}(\varnothing)$ from Example 3.9. Corollary 4.11 is an easy consequence of Lemma 4.10 and will be useful later:

Corollary 4.11. *Suppose* $i{\in}\mathbb{Z}$ *and* $a{\in}\mathbb{A}^i$ *and* $x{\in}\mathsf{Set}^i$. *Then*

$$\mathtt{F}[a{\mapsto}x] = \mathtt{F} \quad \textit{and} \quad \mathtt{T}[a{\mapsto}x] = \mathtt{T}.$$

Proof. By Theorem 2.26 $\mathit{supp}(\mathtt{F}){=}\varnothing$ so that $a\#x$. We use Lemma 4.10. Similarly for $\mathtt{T}$. □

4.2.3. *σ commutes with atoms-concretion.* Lemma 4.12 will be useful later, starting with Proposition 4.13:

Lemma 4.12 $((\sigma@))$**.** *Suppose* $i{\in}\mathbb{Z}$ *and* $a{\in}\mathbb{A}^i$ *and* $x{\in}\mathsf{Set}^i$. *Suppose* $k{\in}\mathbb{Z}$ *and* $z \in \mathsf{Set}^k$ *is an internal comprehension and* $c{\in}\mathbb{A}^{k\text{-}1}$ *and* $c\#z, x$. *Then*

$$(z@c)[a{\mapsto}x] = z[a{\mapsto}x]@c.$$

($z@c$ *is from Notation 3.10.)*

Proof. By Lemma 3.11(2) we may write $z = \mathtt{st}([c]Z)$ where $Z = z@c$. We reason as follows:

$$\begin{array}{lll}
(z@c)[a{\mapsto}x] = Z[a{\mapsto}x] & & Z = z@c \\
= \mathtt{st}([c](Z[a{\mapsto}x]))@c & & \text{Lemma 3.11(2)} \\
= (\mathtt{st}([c]Z)[a{\mapsto}x])@c & & \text{Figure 2}(\sigma\mathtt{st}), c\#x \\
= (z[a{\mapsto}x])@c & & z = \mathtt{st}([c]Z)
\end{array}$$

□

4.2.4. *σ commutes with itself: the 'substitution lemma'.*

Proposition 4.13. *Suppose* $Z{\in}\mathsf{Pred}$ *and* $k{\in}\mathbb{Z}$ *and* $z{\in}\mathsf{Set}^k$. *Suppose* $i{\in}\mathbb{Z}$ *and* $a{\in}\mathbb{A}^i$ *and* $x{\in}\mathsf{Set}^i$ *and suppose* $j{\in}\mathbb{Z}$ *and* $b{\in}\mathbb{A}^j$ *and* $y{\in}\mathsf{Set}^j$ *and* $a\#y$. *Then*

$$\begin{array}{l}
Z[a{\mapsto}x][b{\mapsto}y] = Z[b{\mapsto}y][a{\mapsto}x[b{\mapsto}y]] \\
z[a{\mapsto}x][b{\mapsto}y] = z[b{\mapsto}y][a{\mapsto}x[b{\mapsto}y]].
\end{array}$$

Proof. For brevity we may write

$$\sigma \text{ for } [a{\mapsto}x][b{\mapsto}y] \quad \text{and} \quad \sigma' \text{ for } [b{\mapsto}y][a{\mapsto}x[b{\mapsto}y]].$$

Fix some $k{\in}\mathbb{Z}$. We prove the Lemma for all Z, a, x, b, y and z, a, x, b, y with $minlevel(Z, a, x, b, y){\geq}k$ and $minlevel(z, a, x, b, y){\geq}k$ (Definition 4.4), reasoning by induction on

$$(level(a){+}level(b),\, age(Z)) \quad \text{and} \quad (level(a){+}level(b),\, age(z))$$

lexicographically ordered. Since k was arbitrary, this suffices to prove it for all Z, a, x, b, y and z, a, x, b, y.

We consider the possibilities for $Z{\in}\mathsf{Pred}$:

- *The case of* $\mathtt{and}(\mathcal{X})$ *for* $\mathcal{X} \subseteq_{fin} \mathsf{Pred}$. We use rule $(\sigma\mathtt{and})$ of Figure 2 and the inductive hypothesis.
- *The case of* $\mathtt{neg}(X)$ *for* $X{\in}\mathsf{Pred}$. We use $(\sigma\mathtt{neg})$ of Figure 2 and the inductive hypothesis.
- *The case of* $\mathtt{all}([a']X)$ *for* $X{\in}\mathsf{Pred}$ *and* $a'{\in}\mathbb{A}^{i'}$ *for some* $i'{\in}\mathbb{Z}$. We use Lemma 3.6(1) to assume without loss of generality that $a'\#x, y$, and then we use $(\sigma\mathtt{all})$ of Figure 2 and the inductive hypothesis.
- *The case of* $\mathtt{elt}(z, b)$ *for* $z{\in}\mathsf{Set}^{j\text{-}1}$ *where* $j{\in}\mathbb{Z}$. There are two sub-cases:
  - *Suppose* $y{=}\mathtt{atm}(n)$ *for some* $n{\in}\mathbb{A}^i$ *other than* a (we assumed $a\#y$ so $n{=}a$ is impossible). We reason as follows:

$$\begin{array}{lll}
(\mathtt{elt}(z,b))\ [a{\mapsto}x][b{\mapsto}\mathtt{atm}(n)] & & \\
\quad = (\mathtt{elt}(z[a{\mapsto}x], b))[b{\mapsto}\mathtt{atm}(n)] & & \text{Figure 2}(\sigma\mathtt{eltb}) \\
\quad = \mathtt{elt}(z\sigma, n) & & \text{Figure 2}(\sigma\mathtt{eltatm}) \\
\quad = \mathtt{elt}(z\sigma', n) & & \text{IH } age(z){<}age(\mathtt{elt}(z,b)),\ a\#y \\
\quad = (\mathtt{elt}(z[b{\mapsto}\mathtt{atm}(n)], n))[a{\mapsto}x[b{\mapsto}\mathtt{atm}(n)]] & & \text{Figure 2}(\sigma\mathtt{eltb}) \\
\quad = (\mathtt{elt}(z, b))[b{\mapsto}\mathtt{atm}(n)][a{\mapsto}x[b{\mapsto}\mathtt{atm}(n)]] & & \text{Figure 2}(\sigma\mathtt{eltatm})
\end{array}$$

  - *Suppose* $y{=}\mathtt{st}([b']Y')$ *where* $Y' = y@b'$ *for some fresh* $b'{\in}\mathbb{A}^{j\text{-}1}$ *(so* $b'\#z, x, y$ *and* $k{\leq}level(b')$*).* Note by Theorem 2.26 that $a\#Y'$ and $b'\#x[b{\mapsto}y]$. We reason as follows:

$$\begin{array}{lll}
(\mathtt{elt}(z,b))\ [a{\mapsto}x][b{\mapsto}y] & & \\
\quad = (\mathtt{elt}(z[a{\mapsto}x], b))[b{\mapsto}y] & & \text{Figure 2}(\sigma\mathtt{eltb}) \\
\quad = Y'[b'{\mapsto}z\sigma] & & \text{Figure 2}(\sigma\mathtt{elta}) \\
\quad = Y'[b'{\mapsto}z\sigma'] & & \text{IH } age(z){<}age(\mathtt{elt}(z,b)),\ a\#y \\
\quad = Y'[a{\mapsto}x[b{\mapsto}y]][b'{\mapsto}z\sigma'] & & \text{Lemma 4.10},\ a\#Y' \\
\quad = Y'[b'{\mapsto}z[b{\mapsto}y]]\ [a{\mapsto}x[b{\mapsto}y]] & & \text{IH } level(b'){<}level(b),\ b'\#x[b{\mapsto}y] \\
\quad = (\mathtt{elt}(z, b))[b{\mapsto}y][a{\mapsto}x[b{\mapsto}y]] & & \text{Figure 2}(\sigma\mathtt{elta})
\end{array}$$

- *The case of* $\mathtt{elt}(z, a)$ *for* $z{\in}\mathsf{Set}^{i\text{-}1}$ *where* $i{\in}\mathbb{Z}$. There are two sub-cases:
  - *Suppose* $x{=}\mathtt{atm}(n)$ *for some* $n{\in}\mathbb{A}^i$. If $n{\neq}b$ then we reason as follows:

$$\begin{array}{lll}
(\mathtt{elt}(z,a))\ [a{\mapsto}\mathtt{atm}(n)][b{\mapsto}y] & & \\
\quad = (\mathtt{elt}(z[a{\mapsto}\mathtt{atm}(n)], n))[b{\mapsto}y] & & \text{Figure 2}(\sigma\mathtt{eltatm}) \\
\quad = \mathtt{elt}(z\sigma, n) & & \text{Figure 2}(\sigma\mathtt{eltb}) \\
\quad = \mathtt{elt}(z\sigma', n) & & \text{IH } age(z){<}age(\mathtt{elt}(z,a)),\ a\#y \\
\quad = \mathtt{elt}(z[b{\mapsto}y][a{\mapsto}\mathtt{atm}(n)], n) & & (\sigma\mathbf{b})\ n{\neq}b \\
\quad = (\mathtt{elt}(z[b{\mapsto}y], a))[a{\mapsto}\mathtt{atm}(n)] & & \text{Figure 2}(\sigma\mathtt{eltatm}) \\
\quad = (\mathtt{elt}(z[b{\mapsto}y], a))[a{\mapsto}\mathtt{atm}(n)[b{\mapsto}y]] & & (\sigma\mathbf{b})\ n{\neq}b \\
\quad = (\mathtt{elt}(z, a))[b{\mapsto}y][a{\mapsto}\mathtt{atm}(n)[b{\mapsto}y]] & & \text{Figure 2}(\sigma\mathtt{eltb})
\end{array}$$

If $n{=}b$ so that $x{=}\mathtt{atm}(b)$, and $y{=}\mathtt{atm}(m)$ for some $m{\in}\mathbb{A}^j$ other than a, then we reason as follows:

$$\begin{array}{lll}
(\mathtt{elt}(z,a))\ [a{\mapsto}\mathtt{atm}(b)][b{\mapsto}\mathtt{atm}(m)] & & \\
\quad = (\mathtt{elt}(z[a{\mapsto}\mathtt{atm}(b)],b))[b{\mapsto}\mathtt{atm}(m)] & & \text{Figure 2}(\sigma\mathtt{eltatm}) \\
\quad = \mathtt{elt}(z[a{\mapsto}\mathtt{atm}(b)][b{\mapsto}\mathtt{atm}(m)],m) & & \text{Figure 2}(\sigma\mathtt{eltatm}) \\
\quad = \mathtt{elt}(z[b{\mapsto}\mathtt{atm}(m)][a{\mapsto}\mathtt{atm}(b)[b{\mapsto}\mathtt{atm}(m)]],m) & & \text{IH } age(z){<}age(\mathtt{elt}(z,a)),\ a\#m \\
\quad = (\mathtt{elt}(z[b{\mapsto}\mathtt{atm}(m)],m))[a{\mapsto}\mathtt{atm}(b)[b{\mapsto}\mathtt{atm}(m)]] & & \text{Figure 2}(\sigma\mathtt{eltb}) \\
\quad = (\mathtt{elt}(z,b))[b{\mapsto}\mathtt{atm}(m)][a{\mapsto}\mathtt{atm}(b)[b{\mapsto}\mathtt{atm}(m)]] & & \text{Figure 2}(\sigma\mathtt{eltatm})
\end{array}$$

If $n{=}b$ so that $x{=}\mathtt{atm}(b)$, and $y{=}\mathtt{st}([b']Y')$ where $Y' = y@b'$ for some fresh $b'{\in}\mathbb{A}^{j\text{-}1}$ (so $b'\#z,n,y$ and $k{\leq}level(b')$) then we reason as follows (note by Theorem 2.26 that $a\#Y'$ and $b'\#x[b{\mapsto}y]$):

$$\begin{array}{lll}
(\mathtt{elt}(z,a))\ [a{\mapsto}\mathtt{atm}(b)][b{\mapsto}y] & & \\
\quad = (\mathtt{elt}(z[a{\mapsto}\mathtt{atm}(b)],b))[b{\mapsto}y] & & \text{Figure 2}(\sigma\mathtt{eltatm}) \\
\quad = Y'[b'{\mapsto}z\sigma] & & \text{Figure 2}(\sigma\mathtt{elta}) \\
\quad = Y'[b'{\mapsto}z\sigma'] & & \text{IH } age(z){<}age(\mathtt{elt}(z,a)),\ a\#y \\
\quad = Y'[a{\mapsto}\mathtt{atm}(b)[b{\mapsto}y]][b'{\mapsto}z\sigma'] & & \text{Lemma 4.10},\ a\#Y' \\
\quad = Y'[b'{\mapsto}z[b{\mapsto}y]][a{\mapsto}\mathtt{atm}(b)[b{\mapsto}y]] & & \text{IH } level(b'){<}level(b),\ b'\#x[b{\mapsto}y] \\
\quad = (\mathtt{elt}(z,a))[b{\mapsto}y][a{\mapsto}\mathtt{atm}(b)[b{\mapsto}y]] & & \text{Figure 2}(\sigma\mathtt{eltb})
\end{array}$$

– *Suppose* $x{=}\mathtt{st}([a']X')$ *where* $X' = x@a'$ *for some fresh* $a'{\in}\mathbb{A}^{i\text{-}1}$ *(so* $a'\#z,x,y$ *and* $k{\leq}level(a')$*).* We reason as follows:

$$\begin{array}{lll}
(\mathtt{elt}(z,a))[a{\mapsto}x][b{\mapsto}y] = X'[a'{\mapsto}z[a{\mapsto}x]]\ [b{\mapsto}y] & & \text{Figure 2}(\sigma\mathtt{elta}) \\
\quad = X'[b{\mapsto}y][a'{\mapsto}z\sigma] & & \text{IH } k{\leq}level(a'){=}level(a)\text{-}1,\ a'\#y \\
\quad = X'[b{\mapsto}y][a'{\mapsto}z\sigma'] & & \text{IH } age(z){<}age(\mathtt{elt}(z,a)),\ a\#y \\
\quad = (x[b{\mapsto}y]@a')[a'{\mapsto}z\sigma'] & & \text{Lemma 4.12},\ a'\#y \\
\quad = (\mathtt{elt}(z[b{\mapsto}y],a))[a{\mapsto}x[b{\mapsto}y]] & & \text{Figure 2}(\sigma\mathtt{elta}) \\
\quad = (\mathtt{elt}(z,a))[b{\mapsto}y][a{\mapsto}x[b{\mapsto}y]] & & \text{Figure 2}(\sigma\mathtt{elta}),\ a\#y
\end{array}$$

- *The case of* $\mathtt{elt}(z,c)$ *for* $k{\in}\mathbb{Z}$ *and* $c{\in}\mathbb{A}^k$ *and* $z{\in}\mathsf{Set}^{k\text{-}1}$. We reason as follows:

$$\begin{array}{lll}
(\mathtt{elt}(z,c))[a{\mapsto}x][b{\mapsto}y] = \mathtt{elt}(z\sigma,c) & & \text{Figure 2 } (\sigma\mathtt{eltb}), \text{ twice} \\
\quad = \mathtt{elt}(z\sigma',c) & & \text{IH } age(z){<}age(\mathtt{elt}(z,a)),\ a\#y \\
\quad = (\mathtt{elt}(z,c))[b{\mapsto}y][a{\mapsto}x[b{\mapsto}y]] & & \text{Figure 2 } (\sigma\mathtt{eltb}), \text{ twice}
\end{array}$$

We consider the possibilities for $z{\in}\mathsf{Set}^k$:

- If z is an internal atom then we reason using $(\sigma\mathbf{a})$ and $(\sigma\mathbf{b})$ of Figure 2.
- If z is an internal comprehension then we use Lemma 3.11(2&3) for a fresh $c{\in}\mathbb{A}^{k\text{-}1}$ (so $c\#z$), $(\sigma\mathtt{st})$, and the inductive hypothesis. □

Remark 4.14. Were Proposition 4.13 about the syntax of first-order logic or the λ-calculus, then it could be called *the substitution lemma*, and the proof would be a routine induction on syntax.

In fact, even in the case of first-order logic or the λ-calculus, the proof is not routine. Issues with binders (Figure 2 $(\sigma\mathtt{elta})$, and one explicit in $(\sigma\mathtt{st})$) were the original motivation for my thesis [Gab01] and for nominal techniques in general.

For a standard non-rigorous non-nominal proof of the substitution lemma see [Bar84]; for a detailed discussion of the lemma in the context of Nominal Isabelle, see [Bar14] which includes many further references.

But the proof of Proposition 4.13 is not just a replay of the proofs; neither in the 'classic' sense of [Bar84] nor in the 'nominal' sense of [Gab01, Bar14]. This is because of the interaction of $\mathtt{elt}$

with the σ-action, mostly because of $(\sigma\mathtt{elta})$ (to a lesser extent also because of the nominal binder $(\sigma\mathtt{st})$).

Corollary 4.15 ($(\sigma\mathbf{swp})$)**.** *Suppose* $Z{\in}\mathsf{Pred}$ *and* $k{\in}\mathbb{Z}$ *and* $z{\in}\mathsf{Set}^k$*. Suppose* $i{\in}\mathbb{Z}$ *and* $a{\in}\mathbb{A}^i$ *and* $x{\in}\mathsf{Set}^i$ *and suppose* $j{\in}\mathbb{Z}$ *and* $b{\in}\mathbb{A}^j$ *and* $y{\in}\mathsf{Set}^j$*. Suppose* $a\#y$ *and* $b\#x$*. Then*

$$Z[a{\mapsto}x][b{\mapsto}y] = Z[b{\mapsto}y][a{\mapsto}x]$$
$$z[a{\mapsto}x][b{\mapsto}y] = z[b{\mapsto}y][a{\mapsto}x].$$

Proof. From Proposition 4.13 and Lemma 4.10. □

Corollary 4.16 ($(\sigma\mathbf{asc})$)**.** *Suppose* $Z{\in}\mathsf{Pred}$ *and* $k{\in}\mathbb{Z}$ *and* $z{\in}\mathsf{Set}^k$*. Suppose* $i{\in}\mathbb{Z}$ *and* $a{\in}\mathbb{A}^i$ *and* $x{\in}\mathsf{Set}^i$ *and suppose* $j{\in}\mathbb{Z}$ *and* $b{\in}\mathbb{A}^j$ *and* $y{\in}\mathsf{Set}^j$*. Suppose* $a\#y$ *and* $b\#Z, z$*.*[14] *Then*

$$Z[a{\mapsto}x[b{\mapsto}y]] = Z[a{\mapsto}x][b{\mapsto}y]$$
$$z[a{\mapsto}x[b{\mapsto}y]] = z[a{\mapsto}x][b{\mapsto}y].$$

Proof. From Proposition 4.13 and Lemma 4.10. □

4.2.5. $(\sigma\mathbf{id})$*: substitution for atoms and its corollaries.* We called $\mathtt{atm}(a)$ in Definition 3.3 an *internal atom*. Atoms in nominal techniques interpet variables, so if we call $\mathtt{atm}(a)$ an internal atom this should suggest that $\mathtt{atm}(a)$ should behave like a variable (or a variable symbol). Rules $(\sigma\mathbf{a})$ and $(\sigma\mathbf{b})$ from Figure 2 are consistent with that, and Lemma 4.17 makes formal more of this intuition:

Lemma 4.17 ($(\sigma\mathbf{id})$)**.** *Suppose* $i{\in}\mathbb{Z}$ *and* $a{\in}\mathbb{A}^i$*. Then:*
(1) *If* $Z{\in}\mathsf{Pred}$ *then* $Z[a{\mapsto}\mathtt{atm}(a)] = Z$*.*
(2) *If* $k{\in}\mathbb{Z}$ *and* $z{\in}\mathsf{Set}^k$ *then* $z[a{\mapsto}\mathtt{atm}(a)] = z$*.*

Proof. We reason by induction on

$$age(Z) \quad \text{and} \quad age(z).$$

We consider the possibilities for $Z{\in}\mathsf{Pred}$:

- If $Z = \mathtt{and}(\mathcal{Z})$ for $\mathcal{Z} \subseteq_{fin} \mathsf{Pred}$ or $Z = \mathtt{neg}(Z')$ for $Z'{\in}\mathsf{Pred}$ then we use rules $(\sigma\mathtt{and})$ and $(\sigma\mathtt{neg})$ of Figure 2 and the inductive hypothesis.
- If $Z = \mathtt{all}([a']Z')$ for $Z'{\in}\mathsf{Pred}$ and $a'{\in}\mathbb{A}^{i'}$ for some $i'{\in}\mathbb{Z}$ then we use $(\sigma\mathtt{all})$ of Figure 2 and the inductive hypothesis.
- If $Z = (\mathtt{elt}(z, b))$ for $j{\in}\mathbb{Z}$ and $b{\in}\mathbb{A}^j$ and $z{\in}\mathsf{Set}^{j\text{-}1}$ then we use rule $(\sigma\mathtt{eltb})$ of Figure 2 and the inductive hypothesis.
- If $Z = (\mathtt{elt}(z, a))$ for $z{\in}\mathsf{Set}^{i\text{-}1}$ then we use $(\sigma\mathtt{eltatm})$ of Figure 2 and the inductive hypothesis for z.

We consider the possibilities for $z{\in}\mathsf{Set}^k$:

- If z is an atom then we reason using $(\sigma\mathbf{a})$ or $(\sigma\mathbf{b})$ of Figure 2.
- If z is an internal comprehension then we use Lemma 3.11(2&3) for a fresh $c{\in}\mathbb{A}^{k\text{-}1}$ (so $c\#z$), $(\sigma\mathtt{st})$, and the inductive hypothesis. □

[14] We expect a stronger version of Corollary 4.16 to be possible in which we do not assume $a\#y$. However, the proof would require an induction resembling the proof of Proposition 4.13 — the proof assuming $a\#y$ can piggyback on the induction already given in Proposition 4.13. We will not need this stronger version, so we do not bother.

Given what we have so far, Lemma 4.18 is not hard to prove.

Lemma 4.18 ($(\sigma\mathbf{ren})$)**.** *Suppose* $i{\in}\mathbb{Z}$ *and* $a, a'{\in}\mathbb{A}^i$*. Then:*
- *If* $Z{\in}\mathsf{Pred}$ *and* $a'\#Z$ *then* $Z[a{\mapsto}\mathtt{atm}(a')] = (a'\ a){\cdot}Z$*.*
- *If* $k{\in}\mathbb{Z}$ *and* $z{\in}\mathsf{Set}^k$ *and* $a'\#z$ *then* $z[a{\mapsto}\mathtt{atm}(a')] = (a'\ a){\cdot}z$*.*

Proof. Suppose $Z{\in}\mathsf{Pred}$ and $a'\#Z$. We note by Lemma 4.8(1) (since $a'\#Z$) that $Z[a{\mapsto}\mathtt{atm}(a')] = ((a'\ a){\cdot}Z)[a'{\mapsto}\mathtt{atm}(a')]$ and use Lemma 4.17(1). The case of $z{\in}\mathsf{Set}^k$ is exactly similar. □

4.3. **Sigma-algebras and** SUB**.** We can now observe that our sigma-action is consistent with the nominal algebra literature in the following sense:

Theorem 4.19. *The syntaxes of internal predicates and internal terms, with the sigma-action from Definition 4.1, are* sigma-algebras *in the sense of* [Gab16, GG17]*, and models of* SUB *in the sense of* [GM08].

Concretely, this means that the sigma-action from Definition 4.1 and Figure 2 should
- *distribute through* `and`*,* `neg`*, and*
- *distribute in a capture-avoiding manner through* `all`*, and* `st`*, and*
- *should act on* `atm` *by direct substitution (see* $(\sigma\mathbf{a})$ *and* $(\sigma\mathbf{b})$ *in Figure 2), and*
- *should satisfy the equalities in Figure 3.*

Proof. Immediate from the definitions and lemmas thus far, which were designed to verify these properties. □

Remark 4.20. There is redundancy in Figure 3. For instance, a nominal algebra that satisfies $(\sigma\alpha)$ satisfies $(\sigma\mathbf{id})$ if and only if it satisfies $(\sigma\mathbf{ren})$. One half of this implication is implicit in the proof of Lemma 4.18, which derives $(\sigma\mathbf{ren})$ from (the lemmas corresponding to) $(\sigma\alpha)$ and $(\sigma\mathbf{id})$; going in the other direction is no harder.

Likewise $(\sigma\mathbf{swp})$ can be derived from $(\sigma\sigma)$ and $(\sigma\#)$. This does no harm: in this paper we are interested in exploring the good properties of Definition 4.1, rather than studying minimal sets of axioms for their own sake (for which see [GM08]).

Remark 4.21. We do not demand that the sigma-action should distribute through `elt` but this is because this is syntactically impossible: $\mathtt{elt}(y,a)[a{\mapsto}x]$ cannot be equal to $\mathtt{elt}(y[a{\mapsto}x],x)$ because $\mathtt{elt}(y[a{\mapsto}x],x)$ is not syntax according to Figure 1.

We shall see in Subsection 4.4, however, that this all works after all, in a suitable sense, and this will become an important observation when interpreting TST in internal syntax in Section 5.

Remark 4.22. Another way to approach the proofs in this paper would be to admit $\mathtt{elt}(y,x)$ and an explicit substitution term-former, and orient Figure 2 as rewrite rules. We would obtain a *nominal rewrite system* [FG07]. Essentially this would amount to converting Figure 2 (and the proofs that use it) to a 'small-step' presentation, from the current 'big-step' form.

4.4. **The sugar** $y{\in}x$ **and its properties.** Figure 1 only permits the syntax $\mathtt{elt}(y,a)$, not the syntax $\mathtt{elt}(y,x)$. We can obtain the power of $\mathtt{elt}(y,x)$ via a more sophisticated operation which we construct out of components already available:

Notation 4.23. • Suppose $i{\in}\mathbb{Z}$ and $x{\in}\mathsf{Set}^i$ is an internal comprehension[15] and $y{\in}\mathsf{Set}^{i\text{-}1}$. Then define $y{\in}x$ by

$$y{\in}x = (x@b)[b{\mapsto}y]$$

where we choose $b{\in}\mathbb{A}^{i\text{-}1}$ fresh (so $b\#x, y$).

• Suppose $i{\in}\mathbb{Z}$ and $a{\in}\mathbb{A}^i$ and $y{\in}\mathsf{Set}^{i\text{-}1}$. Then define $y{\in}\mathtt{atm}(a)$ and $y{\in}a$ by

$$y{\in}\mathtt{atm}(a) = \mathtt{elt}(y,a) = y{\in}a.$$

Remark 4.24. Two natural sanity properties for Notation 4.23 are that

(1) it should interact well with sets comprehension on the right-hand side, and
(2) it should interact well with the sigma-action substituting variables for terms.

This is Lemmas 4.25 and 4.26.

Lemma 4.25. *Suppose $X{\in}\mathsf{Pred}$ and $i{\in}\mathbb{Z}$ and $a{\in}\mathbb{A}^i$ and $x{\in}\mathsf{Set}^i$ and $a\#x$. Then (using Notation 4.23)*

$$x{\in}\mathtt{st}([a]X) = X[a{\mapsto}x].$$

Proof. Note by Lemma 2.20 that $a\#\mathtt{st}([a]X)$. By Notation 4.23 (since $a\#x, \mathtt{st}([a]X)$) $x{\in}\mathtt{st}([a]X)$ is equal to $((\mathtt{st}([a]X))@a)[a{\mapsto}x]$ and by Lemma 3.11(4) this is equal to $X[a{\mapsto}x]$. □

Lemma 4.26. *Suppose $i, j{\in}\mathbb{Z}$ and $x{\in}\mathsf{Set}^{i+1}$ and $y{\in}\mathsf{Set}^i$ and $a{\in}\mathbb{A}^j$ and $u{\in}\mathsf{Set}^j$. Then:*

(1) $(y{\in}x)[a{\mapsto}u] = y[a{\mapsto}u]{\in}x[a{\mapsto}u]$.
(2) $(y{\in}a)[a{\mapsto}u] = y[a{\mapsto}u]{\in}u$, *where* $j = i{+}1$.
(3) $(y{\in}b)[a{\mapsto}u] = y[a{\mapsto}u]{\in}b$, *where* $b{\in}\mathbb{A}^{i+1}$.

Proof. First, suppose we have proved part 1 of this result. Then part 2 follows using Figure 2 ($\sigma\mathbf{a}$) and part 3 follows using Figure 2 ($\sigma\mathbf{b}$).

To prove part 1 there are three cases:

• *Suppose $x{=}\mathtt{atm}(a')$ for some $a'{\in}\mathbb{A}^{i+1}$ not equal to a.*
We reason as follows:

$$\begin{array}{lll}(y{\in}\mathtt{atm}(a'))[a{\mapsto}u] = (\mathtt{elt}(y,a'))[a{\mapsto}u] & & \text{Notation 4.23}\\ = \mathtt{elt}(y[a{\mapsto}u], a') & & \text{Figure 2}(\sigma\mathbf{eltb})\\ = y[a{\mapsto}u]{\in}\mathtt{atm}(a') & & \text{Notation 4.23}\\ = y[a{\mapsto}u]{\in}(\mathtt{atm}(a')[a{\mapsto}u]) & & \text{Figure 2}(\sigma\mathbf{b})\end{array}$$

• *Suppose $x{=}\mathtt{atm}(a)$ (so that $j{=}i{+}1$).*
There are two sub-cases:
– *Suppose $u{=}\mathtt{atm}(n)$ for some $n{\in}\mathbb{A}^i$.* We reason as follows:

$$\begin{array}{lll}(y{\in}\mathtt{atm}(a))[a{\mapsto}\mathtt{atm}(n)] = (\mathtt{elt}(y,a))[a{\mapsto}\mathtt{atm}(n)] & & \text{Notation 4.23}\\ = \mathtt{elt}(y[a{\mapsto}\mathtt{atm}(n)], n) & & \text{Figure 2}(\sigma\mathbf{eltatm})\\ = y[a{\mapsto}\mathtt{atm}(n)]{\in}\mathtt{atm}(n) & & \text{Notation 4.23}\end{array}$$

– *Suppose $u{=}\mathtt{st}([a']U')$ where $U' = u@a'$ for some fresh $a'{\in}\mathbb{A}^{i\text{-}1}$ (so $a'\#u, y$).* We reason as follows:

$$\begin{array}{lll}(y{\in}\mathtt{atm}(a))[a{\mapsto}u] = (\mathtt{elt}(y,a))[a{\mapsto}u] & & \text{Notation 4.23}\\ = U'[a'{\mapsto}y[a{\mapsto}u]] & & \text{Figure 2}(\sigma\mathbf{elta})\\ = y[a{\mapsto}u]{\in}u & & \text{Notation 4.23}\end{array}$$

[15] Terminology from Notation 3.4. So x has the form $\mathtt{st}([b]x@b)$ for some $b{\in}\mathbb{A}^{i\text{-}1}$ with $b\#x$, and x does not have the form $\mathtt{atm}(a)$ for any $a{\in}\mathbb{A}^{i1}$.

- *Suppose x is an internal comprehension (not an internal atom).*
  Choose $b{\in}\mathbb{A}^i$ fresh (so $b\#x, y, u$). We reason as follows:

$$\begin{array}{rll}
(y{\in}x)[a{\mapsto}u] & = (x@b)[b{\mapsto}y][a{\mapsto}u] & \text{Notation 4.23} \\
& = (x@b)[a{\mapsto}u][b{\mapsto}y[a{\mapsto}u]] & \text{Proposition 4.13 } b\#u \\
& = (x[a{\mapsto}u]@b)[b{\mapsto}y[a{\mapsto}u]] & \text{Lemma 4.12 } b\#x,u \\
& = y[a{\mapsto}u]{\in}x[a{\mapsto}u] & \text{Notation 4.23}
\end{array}$$

□

Corollary 4.27. *Suppose $k{\in}\mathbb{Z}$ and $c{\in}\mathbb{A}^k$ and $x{\in}\mathsf{Set}^k$ and $y{\in}\mathsf{Set}^{k\text{-}1}$ and $c\#y$. Then*

$$\textit{if } z{=}\mathtt{st}([c](y{\in}c)){\in}\mathsf{Set}^{k+1} \textit{ then } x{\in}z = y{\in}x.$$

Proof. We reason as follows:

$$\begin{array}{rll}
x{\in}z & = x{\in}\mathtt{st}([c]y{\in}c) & \text{Assumption} \\
& = (y{\in}c)[c{\mapsto}x] & \text{Lemma 4.25} \\
& = y[c{\mapsto}x]{\in}x & \text{Lemma 4.26(1)} \\
& = y{\in}x & \text{Lemma 4.10 } c\#y
\end{array}$$

□

Remark 4.28. It is quite interesting to reflect on the inductive measures used in the proofs above. Collecting them a list, they are:

- $age(Z)$ and $age(z)$ in Lemmas 4.8, 4.10, and 4.17.
- $(level(a), age(Z))$ and $(level(a), age(z))$ in Proposition 4.6.
- $(level(a){+}level(b), age(Z))$ and $(level(a){+}level(b), age(z))$ in Proposition 4.13.

So we see that the inductive proofs fall into two categories:

(1) those inductive proofs that are by a direct induction on structure and are 'fairly simple', and
(2) those inductive proofs that depend on the hierarchy of levels and are 'slightly harder'.

Looking deeper at the slightly harder results, the inductive quantities seem to follow a slogan of

> *take the sum of the levels of relevant atoms, and the age of the relevant terms, lexicographically ordered.*

These inductive quantities are simple, though a certain amount of thinking was required to develop them in the first place. For future work, if e.g. a package for handling stratified syntax is implemented in a theorem-prover, then the slogan above might form the basis of a generic automated proof-method.

Though the proofs above are probably susceptible to automation by a sufficiently advanced tactic, they are not all the same.

5. The language of Typed Sets

We now have everything we need to develop the syntax of Typed Set Theory.

$$\phi, \psi ::= \bot \mid \neg\phi \mid \phi \wedge \phi \mid \forall a.\phi \mid s \in s$$
$$s, t, r ::= a \mid \{a \mid \phi\}$$

Figure 4: The syntax of Stratified Sets

5.1. **Syntax of Stratified Sets.**

Definition 5.1. Let **formulae** and **terms** be inductively defined as in Figure 4. In that figure, a ranges over atoms (Definition 2.1) and $\forall$ is taken to bind a and we quotient by α-equivalence. We write $\phi[a{:=}s]$ and $r[a{:=}s]$ for the usual capture-avoiding substitution on syntax.

Remark 5.2. Quotienting by α-equivalence means that a formula ϕ is actually an α-equivalence class of syntax-trees and similarly for a term r. This is a typical treatment but we could just as easily set things up differently, e.g. using nominal abstract syntax or de Bruijn indexes. Definition 5.1 as written is designed to be close to what one might find in a typical paper on TST+ or NF if the syntax were specified.[16]

Definition 5.3 is standard:

Definition 5.3. Suppose t is a term (Definition 5.1). Then extend $level(a)$ from Definition 2.1 from atoms to all terms by:

$$level(\{a \mid \phi\}) = level(a){+}1$$

Call a formula ϕ or term t **stratified** when:

if $s' \in s$ is a subterm of ϕ or t then $level(s) = level(s'){+}1$.

Example 5.4. Suppose $a \in \mathbb{A}^2$, $b \in \mathbb{A}^3$, and $c \in \mathbb{A}^4$. Then $a \in b$ and $b \in c$ are stratified, and $a \in c$, $b \in a$, and $a \in a$ are not stratified.

Definition 5.5. The language of **Stratified Sets** consists of stratified formulae and terms.

Remark 5.6. We only care about *stratified* formulae and terms henceforth — that is, we restrict attention to those formulae and terms that are stratified.

So for all terms and formulae considered from now on, the reader should assume they are stratified, where:

- $\in$ polymorphically takes two terms of type i and $i{+}1$ to a formula for each $i \in \mathbb{Z}$, and
- sets comprehension $\{a \mid \phi\}$ takes an atom of type $i \in \mathbb{Z}$ and a formula ϕ to a term of type $i{+}1$.

We further assume that levels are arranged to respect stratification where this is required, so for example when we write $[a{:=}s]$ it is understood that we assume $a \in \mathbb{A}^{level(s)}$.

Remark 5.7. We could add equality $s{=}t$ to our syntax in Figure 4, at some modest cost in extra cases in inductive arguments. The pertinent stratification condition would be that if $s{=}t$ is a subterm then $level(s) = level(t)$. Our results extend without issues to the syntax with equality.

[16] . . . which it typically is not. For instance [For95] does not formally define its syntax e.g. via an inductive definition in the style of Definition 5.1, and this is not unusual.

$$\begin{array}{rlrl}
\langle\bot\rangle &= \mathtt{F} & \langle t{\in}s\rangle &= \langle t\rangle{\in}\langle s\rangle \\
\langle\neg\phi\rangle &= \mathtt{neg}(\langle\phi\rangle) & \langle\{a|\phi\}\rangle &= \mathtt{st}([a]\langle\phi\rangle) \\
\langle\phi{\wedge}\psi\rangle &= \mathtt{and}(\{\langle\phi\rangle,\langle\psi\rangle\}) & \langle a\rangle &= \mathtt{atm}(a) \\
\langle\forall a.\phi\rangle &= \mathtt{all}([a]\langle\phi\rangle) & &
\end{array}$$

Figure 5: Interpretation of formulae and terms

5.2. **Interpretation for formulae and terms.**

Definition 5.8. Define an **interpretation** of stratified formulae ϕ and terms s as in Figure 5, mapping ϕ to $\langle\phi\rangle \in \mathsf{Pred}$ and s of level $i{\in}\mathbb{Z}$ to $\langle s\rangle \in \mathsf{Set}^i$.

Remark 5.9. For the reader's convenience we give pointers for the notation used in the right-hand sides of the equalities in Figure 5:

- $\mathtt{F}$ is from Example 3.9.
- $\mathtt{neg}$ is from Definition 3.3.
- $\langle t\rangle{\in}\langle s\rangle$ is from Notation 4.23.
- $\mathtt{st}([a]\langle\phi\rangle)$ is from Definitions 2.19 and 3.3.
- $\mathtt{atm}$ is from Definition 3.3.

Note that the translation in Figure 5 from the syntax of formulae ϕ and terms s from Figure 4 to the syntax of internal predicates and internal sets from Definition 3.3 is not entirely direct: $t{\in}s$ is primitive in formulae but only primitive in internal predicates if s is an atom.

Definition 5.10. Define the **size** of a stratified formula ϕ and stratified term t inductively as follows:

$$\begin{array}{rlrl}
\mathit{size}(a) &= 1 & \mathit{size}(\{a|\phi\}) &= \mathit{size}(\phi){+}1 \\
\mathit{size}(\bot) &= 1 & \mathit{size}(\phi{\wedge}\psi) &= \mathit{size}(\phi)+\mathit{size}(\psi)+1 \\
\mathit{size}(\neg\phi) &= \mathit{size}(\phi){+}1 & \mathit{size}(\forall a.\phi) &= \mathit{size}(\phi){+}1 \\
\mathit{size}(t{\in}s) &= \mathit{size}(t){+}\mathit{size}(s)+1 & &
\end{array}$$

Lemma 5.11. *Suppose ϕ is a stratified formula and s is a stratified term with $\mathit{level}(s) = i \in \mathbb{Z}$. Then*

$$\langle\phi\rangle \in \mathsf{Pred} \quad \textit{and} \quad \langle s\rangle \in \mathsf{Set}^i.$$

Proof. By induction on $\mathit{size}(\phi)$ and $\mathit{size}(s)$:[17]

- *The case of a.* By Figure 5 $\langle a\rangle{=}\mathtt{atm}(a)$. By Definition 3.3 $\mathtt{atm}(a){\in}\mathsf{Set}^i$.
- *The case of $\{b|\phi\}$ for $j{\geq}1$ and $b{\in}\mathbb{A}^j$.* By Figure 5 $\langle\{b|\phi\}\rangle{=}\mathtt{st}([b]\langle\phi\rangle)$. By Definition 5.3 $\mathit{level}\{b|\phi\} = j{+}1$. By inductive hypothesis $\langle\phi\rangle{\in}\mathsf{Pred}$ and by Definition 3.3 $\mathtt{st}([b]\langle\phi\rangle) \in \mathsf{Set}^{j+1}$.
- *The case of $\bot$.* By Figure 5 $\langle\bot\rangle{=}\mathtt{F} \in \mathsf{Pred}$.
- *The case of $\neg\phi$.* From Figure 5 and Definition 3.3 using the inductive hypothesis.
- *The case of $\phi{\wedge}\psi$.* From Figure 5 and Definition 3.3 using the inductive hypothesis.
- *The case of $\forall a.\phi$.* From Figure 5 and Definition 3.3 using the inductive hypothesis.
- *The case of $t{\in}s$.* We refer to Notation 4.23 and use Lemma 3.11 and Proposition 4.6. □

[17] A structural induction on nominal abstract syntax [GP01] would also work, and in a nominal mechanised proof might be preferable. Similarly for Proposition 5.12.

5.3. **Properties of the interpretation.**

Proposition 5.12. *Suppose ϕ is a stratified formula and t, and r are stratified terms and $b{\in}\mathbb{A}^{level(t)}$. Then:*

$$\langle\phi\rangle[b{\mapsto}\langle t\rangle] = \langle\phi[b{:=}t]\rangle$$
$$\langle r\rangle[b{\mapsto}\langle t\rangle] = \langle r[b{:=}t]\rangle$$

Note by Lemma 5.11 that $\langle t\rangle{\in}\mathsf{Set}^{level(t)}$ so that the σ-action $[b{\mapsto}\langle t\rangle]$ above is well-defined (Definition 4.1).

Proof. By induction on $size(\phi)$ and $size(r)$. We consider each case in turn; the interesting case is for $\in$, where we use Lemma 4.26:

- *The case of $\bot$.* We reason as follows:

$$\begin{array}{rll}
\langle\bot\rangle[b{\mapsto}\langle t\rangle] &= \mathtt{F}[b{\mapsto}\langle t\rangle] & \text{Figure 5}\\
&= \mathtt{F} & \text{Corollary 4.11}\\
\langle\bot[b{:=}t]\rangle &= \langle\bot\rangle & \text{Fact of syntax}\\
&= \mathtt{F} & \text{Figure 5}
\end{array}$$

- *The case of $\neg\phi$.* We reason as follows:

$$\begin{array}{rll}
\langle\neg\phi\rangle[b{\mapsto}\langle t\rangle] &= (\mathtt{neg}(\langle\phi\rangle))[b{\mapsto}\langle t\rangle] & \text{Figure 5}\\
&= \mathtt{neg}(\langle\phi\rangle[b{\mapsto}\langle t\rangle]) & \text{Figure 2}\\
&= \mathtt{neg}(\langle\phi[b{:=}t]\rangle) & \text{IH } size(\phi){<}size(\neg\phi)\\
&= \langle\neg(\phi[b{:=}t])\rangle & \text{Figure 2}\\
&= \langle(\neg\phi)[b{:=}t]\rangle & \text{Fact of syntax}
\end{array}$$

- *The case of $\phi{\wedge}\psi$.* We reason as follows:

$$\begin{array}{rll}
\langle\phi{\wedge}\psi\rangle[b{\mapsto}\langle t\rangle] &= (\mathtt{and}(\{\langle\phi\rangle,\langle\psi\rangle\}))[b{\mapsto}\langle t\rangle] & \text{Figure 5}\\
&= \mathtt{and}(\{(\langle\phi\rangle[b{\mapsto}\langle t\rangle]),(\langle\psi\rangle[b{\mapsto}\langle t\rangle])\}) & \text{Figure 2}\\
&= \mathtt{and}(\{\langle\phi[b{:=}t]\rangle,\langle\psi[b{:=}t]\rangle\}) & \text{IH } size(\phi),size(\psi){<}size(\phi{\wedge}\psi)\\
&= \langle\phi[b{:=}t]{\wedge}(\psi[b{:=}t])\rangle & \text{Figure 2}\\
&= \langle(\phi{\wedge}\psi)[b{:=}t]\rangle & \text{Fact of syntax}
\end{array}$$

- *The case of $\forall a.\phi$.* We reason as follows, where we α-rename if necessary to assume $a\#t$ (from which it follows by Theorem 2.26 that $a\#\langle t\rangle$):

$$\begin{array}{rll}
\langle\forall a.\phi\rangle[b{\mapsto}\langle t\rangle] &= \mathtt{all}([a]\langle\phi\rangle)[b{\mapsto}\langle t\rangle] & \text{Figure 5}\\
&= \mathtt{all}([a](\langle\phi\rangle[b{\mapsto}\langle t\rangle])) & \text{Figure 2 } a\#\langle t\rangle\\
&= \mathtt{all}([a]\langle\phi[b{:=}t]\rangle) & \text{IH } size(\phi){<}size(\forall a.\phi)\\
&= \langle\forall a.(\phi[b{:=}t])\rangle & \text{Figure 5}\\
&= \langle(\forall a.\phi)[b{:=}t]\rangle & \text{Fact of syntax, } a\#t
\end{array}$$

- *The case of b.* By Figure 5 $\langle b\rangle{=}\mathtt{atm}(b)$. By assumption $\langle t\rangle{\in}\mathsf{Set}^{level(b)}$ so by Figure 2 ($\sigma\mathbf{a}$)

$$\mathtt{atm}(b)[b{\mapsto}\langle t\rangle] = \langle t\rangle.$$

- *The case of a (any atom other than b).* By Figure 5 $\langle a\rangle{=}\mathtt{atm}(a)$. We use rule ($\sigma\mathbf{b}$) of Figure 2.

$$\frac{}{t{\in}\{a|\phi\} \to \phi[a{:=}t]} \qquad \frac{\phi \to \phi'}{\neg\phi \to \neg\phi'} \qquad \frac{\phi \to \phi'}{\phi{\wedge}\phi'' \to \phi'{\wedge}\phi''} \qquad \frac{\phi \to \phi'}{\phi''{\wedge}\phi \to \phi''{\wedge}\phi'}$$

$$\frac{\phi \to \phi'}{\{a|\phi\} \to \{a|\phi'\}} \qquad \frac{\phi \to \phi'}{\forall a.\phi \to \forall a.\phi'} \qquad \frac{s \to s'}{t{\in}s \to t{\in}s'} \qquad \frac{t \to t'}{t{\in}s \to t'{\in}s}$$

Figure 6: Rewrite system on formulae and terms

- *The case of* $\{a|\phi\}$. α-converting if necessary assume a is fresh (so $a\#t$, and by Theorem 2.26 also $a\#\langle t\rangle$). We reason as follows:

$$\begin{array}{rll}
\langle\{a|\phi\}\rangle[b{\mapsto}\langle t\rangle] & = (\mathtt{st}([a]\langle\phi\rangle))[b{\mapsto}\langle t\rangle] & \text{Figure 5}\\
& = \mathtt{st}([a]\langle\phi\rangle[b{\mapsto}\langle t\rangle]) & \text{Figure 2}(\sigma\mathtt{st}),\ a\#\langle t\rangle\\
& = \mathtt{st}([a]\langle\phi[b{:=}t]\rangle) & \text{IH } size(\phi){<}size(\{a|\phi\})\\
& = \langle\{a|\phi[b{:=}t]\}\rangle & \text{Figure 5},\ a\#t\\
& = \langle\{a|\phi\}[b{:=}t]\rangle & \text{Fact of syntax}
\end{array}$$

- *The case of* $t'{\in}s'$. We reason as follows:

$$\begin{array}{rll}
\langle t'{\in}s'\rangle[b{\mapsto}\langle t\rangle] & = (\langle t'\rangle{\in}\langle s'\rangle)[b{\mapsto}\langle t\rangle] & \text{Figure 5}\\
& = (\langle t'\rangle[b{\mapsto}\langle t\rangle]){\in}(\langle s'\rangle[b{\mapsto}\langle t\rangle]) & \text{Lemma 4.26}\\
& = (\langle t'[b{:=}t]\rangle){\in}(\langle s'[b{:=}t]\rangle) & \text{IH } size(t'), size(s'){<}size(t'{\in}s')\\
& = \langle(t'[b{:=}t]){\in}(s'[b{:=}t])\rangle & \text{Figure 5}
\end{array}$$

□

Lemma 5.13. *Suppose ϕ is a stratified formula and s is a stratified term. Suppose $a \in \mathbb{A}^{i+1}$ and $level(s) = i$. Then:*

(1) $\langle s{\in}\{a|\phi\}\rangle = \langle\phi[a{:=}s]\rangle$.
(2) $\langle s{\in}\{a|\phi\}\rangle = \langle\phi\rangle[a{\mapsto}\langle s\rangle]$.

Proof. We reason as follows:

$$\begin{array}{rll}
\langle s{\in}\{a|\phi\}\rangle & = \langle s\rangle{\in}\langle\{a|\phi\}\rangle & \text{Figure 5}\\
& = \langle s\rangle{\in}\mathtt{st}([a]\langle\phi\rangle) & \text{Figure 5}\\
& = (\mathtt{st}([a]\langle\phi\rangle)@a)[a{\mapsto}\langle s\rangle] & \text{Notation 4.23}\\
& = \langle\phi\rangle[a{\mapsto}\langle s\rangle] & \text{Lemma 3.11(4)}\\
& = \langle\phi[a{:=}s]\rangle & \text{Proposition 5.12}
\end{array}$$

□

5.4. **Confluence.**

Definition 5.14. (1) Let $\to$ be a rewrite relation on the language of Stratified Sets (Definition 5.5) defined by the rules in Figure 6.
(2) Write $\to^*$ for the transitive reflexive closure of $\to$ (so the least transitive reflexive relation containing $\to$).

Notation 5.15. The natural injection of internal predicates and internal terms into stratified formulae and terms is clear; to save notation we elide it, thus effectively treating the syntax of internal predicates and terms from Definition 3.3 as a direct subset of the syntax of stratified formulae and terms from Figure 4. The reader who dislikes this abuse of notation can fill in an explicit injection function ι as required, to map the former injectively into the latter. Either way, the meaning will be the same.

Notation 5.16. Call a formula of the form $t{\in}\{a|\phi\}$ a **reduct**.

Lemma 5.17. *$\langle\phi\rangle$ considered as a formula, is a $\rightarrow$-normal form, and similarly for $\langle s\rangle$.*

Proof. Reducts are impossible because the internal syntax from Figure 1 only allows us to form $y{\in}x$ (written $\mathtt{elt}(y,x)$ in that figure) when x is an atom and not a comprehension. □

We can now state a kind of converse to Lemma 5.17:

Theorem 5.18. (1) *$\phi \rightarrow^* \langle\phi\rangle$ and $s \rightarrow^* \langle s\rangle$.*
Here we use Notation 5.15 to treat $\langle\phi\rangle$ and $\langle s\rangle$ directly as stratified syntax.
(2) *If $\phi \rightarrow \phi'$ then $\langle\phi\rangle = \langle\phi'\rangle$. If $s \rightarrow s'$ then $\langle s\rangle = \langle s'\rangle$.*
(3) *$\phi \rightarrow^* \phi'$ then $\phi' \rightarrow^* \langle\phi\rangle$, and if $s \rightarrow^* s'$ then $s' \rightarrow^* \langle s'\rangle$. As a corollary, the rewrite relation $\rightarrow$ from Figure 6 is confluent.*

Proof. (1) By induction on syntax. The interesting case is for $t{\in}\{a|\phi\}$. Suppose $\phi \rightarrow^* \langle\phi\rangle$. Then

$$\begin{array}{rll} t{\in}\{a|\phi\} & \rightarrow^* \langle t\rangle{\in}\{a|\phi\} & \text{IH, Figure 6} \\ & \rightarrow \langle\phi\rangle[a{\mapsto}\langle t\rangle] & \text{Figure 6} \\ & = \langle t{\in}\{a|\phi\}\rangle & \text{Lemma 5.13} \end{array}$$

(2) By induction on the derivation of the rewrite (that is, on the term-context in which the rewrite takes place). We consider three cases:
- Suppose $t{\in}\{a|\phi\} \rightarrow \phi[a{:=}t]$. By Lemma 5.13 $\langle t{\in}\{a|\phi\}\rangle = \langle\phi[a{:=}t]\rangle$.
- Suppose $t{\in}\{a|\phi\} \rightarrow t{\in}\{a|\phi'\}$ because $\{a|\phi\} \rightarrow \{a|\phi'\}$ because $\phi \rightarrow \phi'$. By induction hypothesis $\langle\phi\rangle = \langle\phi'\rangle$. Then using Lemma 5.13 we have
$$\langle t{\in}\{a|\phi\}\rangle \overset{L5.13}{=} \langle\phi\rangle[a{\mapsto}\langle t\rangle] = \langle\phi'\rangle[a{\mapsto}\langle t\rangle] \overset{L5.13}{=} \langle t{\in}\{a|\phi\}\rangle.$$
- Suppose $t{\in}\{a|\phi\} \rightarrow t'{\in}\{a|\phi\}$ because $t \rightarrow t'$. By induction hypothesis $\langle t\rangle = \langle t'\rangle$. We have
$$\langle t{\in}\{a|\phi\}\rangle \overset{L5.13}{=} \langle\phi\rangle[a{\mapsto}\langle t\rangle] = \langle\phi\rangle[a{\mapsto}\langle t'\rangle] \overset{L5.13}{=} \langle t'{\in}\{a|\phi\}\rangle.$$

(3) We combine parts 1 and 2 of this result. □

5.5. **Strong normalisation.**

Notation 5.19. • Call a comprehension $\{a|\phi\}$ **ternary** when a occurs in ϕ *at least* three times.[18]
If a occurs in ϕ zero, one, or two times then we call $\{a|\phi\}$ **non-ternary**.
- Call a formula ϕ **ternary** if every comprehension in it is ternary.
- Call a term t **ternary** if every comprehension in it is ternary.

Definition 5.20. Suppose ϕ is a formula and t is a term. Write $na(\phi)$ and $na(t)$ for the predicate or term obtained by padding every non-ternary comprehension $\{a'|\phi'\}$ in ϕ or t to

$$\{a'|\phi'{\wedge}\exists c.(a'{\in}c{\wedge}a'{\in}c{\wedge}a'{\in}c)\}$$

where $c \in \mathbb{A}^{level(a')+1}$.[19]

[18] This is arguably an abuse of notation; 'ternary' might suggest *exactly* three times. But we need a name.

[19] The precise choice of c does not matter since we bind it with $\exists$. The level matters, so the result is stratified.

Lemma 5.21. *Suppose ϕ is a formula and t is a term. Then:*
(1) *$na(\phi)$ and $na(t)$ are ternary.*
(2) *If ϕ and s are ternary then $\phi[a{:=}s]$ is ternary. Similarly for $t[a{:=}s]$.*
(3) *If ϕ is ternary and $\phi \to \phi'$ then ϕ' is ternary. Similarly for $t \to t'$.*

Proof. (1) By construction.
(2) By an easy calculation.
(3) By an easy calculation from Figure 6 and part 2 of this result. □

Definition 5.22. Define the **complexity** of a stratified formula ϕ and stratified term t as follows:

$$\begin{array}{rcll} cplx(a) & = & 1 & \\ cplx(\{a|\phi\}) & = & 1 + cplx(\phi) & \\ cplx(\bot) & = & 3 & \\ cplx(\phi \wedge \psi) & = & cplx(\phi) + 1 + cplx(\psi) & \\ cplx(\neg\phi) & = & 1 + cplx(\phi) & \\ cplx(\forall a.\phi) & = & 1 + cplx(\phi) & \\ cplx(b \in \{a|\phi\}) & = & cplx(\phi) & \\ cplx(t \in s) & = & cplx(t) + 1 + cplx(s) & \begin{array}{l} t \text{ not an atom, or} \\ s \text{ not a comprehension} \end{array} \end{array}$$

Define the **number of atomic reducts** of ϕ and t as follows:

$$\begin{array}{rcll} atomic(a) & = & 0 & \\ atomic(\{a|\phi\}) & = & atomic(\phi) & \\ atomic(\bot) & = & 0 & \\ atomic(\phi \wedge \psi) & = & atomic(\phi) + atomic(\psi) & \\ atomic(\neg\phi) & = & atomic(\phi) & \\ atomic(\forall a.\phi) & = & atomic(\phi) & \\ atomic(t \in b) & = & atomic(t) & \\ atomic(t \in \{a|\phi\}) & = & atomic(\phi) + 1 & t \text{ an atom} \\ atomic(t \in s) & = & atomic(t) + atomic(s) & t \text{ not an atom} \end{array}$$

Remark 5.23. (1) If we view $\{a|\phi\}$ as $\lambda a.\phi$ and $t \in s$ as $s\,t$ then an *atomic reduct* is just a term of the form $(\lambda a.\phi)b$. Then $atomic$ counts the number of atomic reducts in formulae ϕ and terms t, and $cplx$ is a measure of size in which atomic reducts are skipped. Indeed, though we never use this directly, from Definition 5.10 we see that

$$size(\phi) = cplx(\phi) + 2 * atomic(\phi),$$

and similarly for t.
(2) Intuitively, $cplx(\bot) = 3$ means "$\bot$ has the same complexity as $b \in a$". Technically, $cplx(\bot) = 3$ makes Lemma 5.24(2) hold, and allows the arithmetic in Lemma 5.26 to go through.

Lemma 5.24. *Suppose s is a term and ϕ is a predicate. Then:*
(1) *$cplx(s) \geq 1$.*
(2) *$cplx(\phi) \geq 3$.*
(3) *If s is not an atom (so it is a comprehension) then $cplx(s) \geq 4$.*

Proof. By calculations and induction using Definition 5.22; the base cases are $\bot$ and $t \in s$ (in particular, $b \in a$). □

Lemma 5.25. *Suppose ϕ is a predicate and t is a term. Suppose a is an atom and s is a term and $level(a) = level(s)$ (so the substitution $[a{:=}s]$ is well-defined). Then:*

(1) *If s is an atom then*

$$cplx(\phi[a{:=}s]) = cplx(\phi) \quad \textit{and} \quad cplx(t[a{:=}s]) = cplx(t).$$

(2) *If s is not an atom (so is a comprehension) then*

$$\begin{array}{c} cplx(\phi[a{:=}s]) \geq cplx(\phi) + n * (cplx(s)\text{-}1) \quad \textit{and} \\ cplx(t[a{:=}s]) \geq cplx(t) + n * (cplx(s)\text{-}1) \end{array}$$

where n is the number of instances of a in ϕ or t.

Proof. (1) By a routine induction on ϕ and t using Definition 5.22.

(2) Essentially this is clear because we replace n instances of a each with complexity 1, with n instances of s each with complexity $cplx(s)$. The proof is by induction on syntax using Definition 5.22. Interesting cases of the induction are ϕ or t that do not mention a (so $n = 0$), $t'{\in}a$, $a{\in}b$, and $a{\in}\{a'|\phi'\}$. We consider each in turn:

- If ϕ or t do not mention a then $\phi[a{:=}s] = \phi$ and $t[a{:=}s] = t$ and the result follows.
- It is a fact that $cplx(t'{\in}a) = cplx(t') + 2$, and we calculate as follows:

$$\begin{array}{rl} cplx(t'[a{:=}s]{\in}s) & = cplx(t'[a{:=}s]) + 1 + cplx(s) \\ & \geq cplx(t') + (n\text{-}1) * (cplx(s)\text{-}1) + 1 + cplx(s) \\ & = cplx(t') + (n\text{-}1) * (cplx(s)\text{-}1) + 2 + (cplx(s)\text{-}1) \\ & = cplx(t') + 2 + n * (cplx(s)\text{-}1) \\ & = cplx(t'{\in}a) + n * (cplx(s)\text{-}1) \end{array}$$

- It is a fact that $cplx(a{\in}b) = 3$, and we calculate as follows:

$$\begin{array}{rl} cplx(s{\in}b) & = cplx(s) + 2 \\ & = 3 + 1 * (cplx(s)\text{-}1) \\ & = cplx(a{\in}b) + 1 * (cplx(s)\text{-}1) \end{array}$$

- It is a fact that $cplx(a{\in}\{a'|\phi'\}) = cplx(\phi')$, and we calculate as follows:

$$\begin{array}{rl} cplx(s{\in}\{a'|\phi'[a{:=}s]\}) & = cplx(s) + 2 + cplx(\phi'[a{:=}s]) \\ & \geq cplx(s) + 2 + cplx(\phi') + (n\text{-}1) * (cplx(s)\text{-}1) \\ & \gneq (cplx(s)\text{-}1) + cplx(\phi') + (n\text{-}1) * (cplx(s)\text{-}1) \\ & = cplx(\phi') + n * (cplx(s)\text{-}1) \end{array}$$

□

Lemma 5.26. *Suppose $\{a|\phi\}$ is ternary (so ϕ mentions a free at least three times) and s is a term and $level(s) = level(a)$. Then:*

(1) *If s is an atom then*

$$cplx(s{\in}\{a|\phi\}) = cplx(\phi[a{:=}s]) = cplx(\phi).$$

(2) *If s is not an atom then*

$$cplx(s{\in}\{a|\phi\}) \lneq cplx(\phi[a{:=}s]).$$

Proof. (1) From Lemma 5.25(1).

(2) Using Definition 5.22 and Lemma 5.25(2) we have the following facts:

- $cplx(s{\in}\{a|\phi\}) = cplx(\phi) + 3 + (cplx(s)\text{-}1)$.
- $cplx(\phi[a{:=}s]) \geq cplx(\phi) + n * (cplx(s)\text{-}1)$.

We can drop the $cplx(\phi)$ on both sides and do some arithmetic:

$$\begin{aligned} 3 + (cplx(s)\text{-}1) \lneq n * (cplx(s)\text{-}1) &\Leftrightarrow 3 \lneq (n\text{-}1) * (cplx(s)\text{-}1) \\ &\Leftrightarrow 3 \lneq (n\text{-}1) * 3 \qquad \text{Lemma 5.24(3)} \\ &\Leftrightarrow 2 \lneq n. \end{aligned}$$

We assumed ϕ is ternary, which means precisely that $n \geq 3$, so this is true. □

Corollary 5.27. *If ϕ is ternary and $\phi \to \phi'$ then precisely one of the following must hold:*

- $cplx(\phi) \lneq cplx(\phi')$ *(in words:* complexity increases*).*
- $cplx(\phi') = cplx(\phi)$ *and* $atomic(\phi) \gneq atomic(\phi')$ *(in words:* atomic reducts decrease*).*

Similarly for $t \to t'$.

Proof. Consider a reduct $s \in \{a \mid \phi'\}$.

- If s is not an atom then we use Lemma 5.26(2).
- If s is an atom then using Lemma 5.26(1) complexity is unchanged; however the number of atomic reducts decrements. □

Proposition 5.28. *The rewrite system from Figure 6 is terminating (no infinite chain of rewrites).*

Proof. We consider just the case of reducing formulae; reducing terms is no harder.

$na(\phi)$ is just an annotated copy of ϕ, so if ϕ has an infinite chain of rewrites then so must $na(\phi)$.[20] We see that it would suffice to prove that reductions from $na(\phi)$ are terminating.

So suppose $na(\phi) = \phi$, and ϕ is ternary (apply na if required).

Consider ϕ' and suppose $\phi \to^* \phi'$. From Theorem 5.18 $\phi' \to^* \langle\phi\rangle$, and by Corollary 5.27(1)

$$cplx(\phi) \leq cplx(\phi') \leq cplx(\langle\phi\rangle).$$

Thus the set $\{cplx(\phi') \mid \phi \to^* \phi'\}$ is bounded above by $cplx(\langle\phi\rangle)$. It follows using Corollary 5.27(1&2) and considering the measure

$$\big(cplx(\langle\phi\rangle) - cplx(\phi'), atomic(\phi')\big),$$

lexicographically ordered, that any chain of reductions from ϕ must terminate. □

Remark 5.29. The proof of Proposition 5.28 is not difficult.[21] This is in itself interesting:

We noticed in Remark 1.6 how stratified syntax can be viewed as a fragment of the simply-typed λ-calculus, where $t \in \{a \mid \phi\}$ corresponds to a β-reduct and extensionality $s = \{b \mid b \in s\}$ corresponds to an η-expansion. Yet, the direct proof of strong normalisation for the simply-typed λ-calculus is quite different and seems harder than the proof of Proposition 5.28 (a concise but clear presentation is in Chapter 6 of [GTL89]).

Theorem 5.30. *Formulae and terms of Stratified Sets, with the rewrites from Figure 6, are confluent and strongly normalising.*

Proof. Confluence and *weak normalisation* (every formula/term has some rewrite to a normal form) are from Theorem 5.18.

Strong normalisation follows from weak normalisation and termination (Proposition 5.28). □

[20] In a machine implementation we would probably want to refine Definition 5.20 to annotate with $a' \in c \wedge a' \in c \wedge a' \in c$ for c a fresh constant-symbol or variable symbol (one for each level), instead of $\exists c.(a' \in c \wedge a' \in c \wedge a' \in c)$. This would make it easier to automatically track the annotations.

[21] . . . but not trivial. Thanks to an anonymous referee for spotting my errors.

Remark 5.31. So from Theorems 5.30 and 4.19 we see that:

(1) the syntax of formulae and terms has normal forms, and furthermore
(2) normal forms with the natural substitution action given by substitute-then-renormalise, corresponds precisely to the theory of internal predicates and terms from Definition 3.3 and 4.1, and furthermore
(3) this theory of normal forms is an instance of the notion of nominal algebras for substitution, also called sigma-algebras, as used in the previous literature studying λ-calculus, first-order logic, and pure substitution [GG17, Gab16, GM08].

Recall from Remarks 1.1 and 1.2 that in stratifiable syntax, as used in Quine's NF, variables do not have predefined levels but we insist on a *stratifiability* condition that ϕ and s are only legal if we *could* assign levels to their variables to stratify them. We obtain as an easy corollary:

Theorem 5.32. *Formulae and terms of stratifiable syntax, with the rewrites from Figure 6, are confluent and strongly normalising.*

Proof. The result follows from Theorem 5.30 by taking a stratifiable ϕ, and stratifying it so that we now have ϕ' in the language of Typed Sets. Rewrites on ϕ' clearly correspond 1-1 with rewrites on ϕ, since Figure 6 makes no reference to the levels of variables. □

6. Conclusions and future work

Stratified Sets occupy a nice middle ground between ZF sets and simple types. They typically appear used as a foundational syntax. However, we have seen in this paper that Typed-Sets-the-syntax in and of itself forms a well-behaved rewrite system, and a well-behaved nominal algebra. This had not previously been noted, and this paper gives a reasonably full and detailed account of how rewriting and nominal algebra apply. This account is intended to be suitable for

- readers familiar with rewriting who are unfamiliar with stratified sets syntax[22]
- readers familiar with stratified sets syntax but unfamiliar with techniques from rewriting and nominal algebra.

We have also tried to smooth a path to implementing these proofs in a machine, hopefully in a nominal context. We have designed the proofs to be friendly to such an implementation as future work, yet without compromising readability for humans. Where we have cut corners (relative to a machine implementation), we tried to signpost this fact (see for instance Remark 4.2 and Notation 5.15).

Concerning other applications, it is often possible to use normal forms to build denotations. In some contexts, the normal form *is* the denotation of the terms that reduce to it. That will not work for Stratified Sets because we are usually interested in imposing additional axioms. But there are standard things that can be done about that, and this has been investigated in a nominal context in papers like [Gab16, GG17]. These papers build denotations for first-order logic and the λ-calculus using maximally consistent sets, and using nominal techniques to manage binding in denotations (extending how we used nominal techniques in this paper to manage binding in syntax). Having normal forms is useful here and the ideas in this paper can be used to give a denotational analysis of theories in the languages of Stratified and Stratifiable Sets. This is future work.

We can ask about a converse to Theorems 5.30 and 5.32. We have shown that a stratifiable formula rewrites to a normal form. Now if a formula (without levels) rewrites to normal form, is it

[22] Stratified sets syntax is not hard to define — but it requires experience to learn what kinds of predicates are and are not stratifiable. In use, stratifiability is a subtle and powerful condition.

stratifiable? We see that we cannot hope for a perfect converse by the following easy example: if we write $\varnothing = \{a \mid \bot\}$ then $\varnothing \in \{a \mid a \notin a\}$ is not stratifiable but it rewrites to $\varnothing \notin \varnothing$, which is stratifiable for instance as $\{a^0 \mid \bot\} \notin \{a^1 \mid \bot\}$, which we could also write just as $\varnothing^1 \notin \varnothing^2$. However there may be special cases in which stratification information can be recovered from normalisation, and this is future work.

References

[Bar84] Henk P. Barendregt. *The Lambda Calculus: its Syntax and Semantics (revised ed.)*. North-Holland, 1984.

[Bar14] Barendregt's substitution lemma, June 2014. http://isabelle.in.tum.de/nominal/example.html, retrieved 2014/June/8.

[DG12a] Gilles Dowek and Murdoch J. Gabbay. Permissive Nominal Logic (journal version). *Transactions on Computational Logic*, 13(3), 2012.

[DG12b] Gilles Dowek and Murdoch J. Gabbay. PNL to HOL: from the logic of nominal sets to the logic of higher-order functions. *Theoretical Computer Science*, 451:38–69, 2012.

[FG07] Maribel Fernández and Murdoch J. Gabbay. Nominal rewriting (journal version). *Information and Computation*, 205(6):917–965, June 2007.

[For95] Thomas E. Forster. *Set theory with a universal set: exploring an untyped universe*. Clarendon Press, 1995.

[For97] Thomas E. Forster. Quine's NF, 60 years on. *American Mathematical Monthly*, 104(9):838–845, 1997.

[Gab01] Murdoch J. Gabbay. *A Theory of Inductive Definitions with alpha-Equivalence*. PhD thesis, University of Cambridge, UK, March 2001.

[Gab11] Murdoch J. Gabbay. Foundations of nominal techniques: logic and semantics of variables in abstract syntax. *Bulletin of Symbolic Logic*, 17(2):161–229, 2011.

[Gab16] Murdoch J. Gabbay. Semantics out of context: nominal absolute denotations for first-order logic and computation. *Journal of the ACM*, 63(3):1–66, June 2016.

[Gab17] Murdoch J. Gabbay. Equivariant ZFA and the foundations of nominal techniques. Submitted. See arXiv preprint arxiv.org/abs/1801.09443, 2017.

[Gab18] Murdoch J. Gabbay. Equivariant ZFA with choice: a position paper. In *Proceedings of the 25th Automated Reasoning Workshop (ARW 2018)*, March 2018. See arXiv preprint arxiv.org/abs/1803.08727.

[GG17] Murdoch J. Gabbay and Michael J. Gabbay. Representation and duality of the untyped lambda-calculus in nominal lattice and topological semantics, with a proof of topological completeness. *Annals of Pure and Applied Logic*, 168:501–621, March 2017.

[GM06] Murdoch J. Gabbay and Aad Mathijssen. Capture-avoiding Substitution as a Nominal Algebra. In *Proceedings of the 3rd International Colloquium on Theoretical Aspects of Computing (ICTAC 2006)*, volume 4281 of *Lecture Notes in Computer Science*, pages 198–212, Berlin, November 2006. Springer.

[GM08] Murdoch J. Gabbay and Aad Mathijssen. Capture-Avoiding Substitution as a Nominal Algebra. *Formal Aspects of Computing*, 20(4-5):451–479, June 2008.

[GM09] 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.

[GP01] 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.

[Gri04] Nicholas Griffin. The Prehistory of Russell's Paradox. In Godehard Link, editor, *One Hundred Years of Russell's Paradox*, number 6 in Series in Logic and Its Applications. De Gruyter, 2004.

[GTL89] Jean-Yves Girard, Paul Taylor, and Yves Lafont. *Proofs and types*. Cambridge University Press, 1989.

[Hol98] Randall Holmes. *Elementary set theory with a universal set*, volume 10. Centre National de recherches de Logique, 1998.

[Jec06] Thomas Jech. *Set theory*. Springer, 2006. Third edition.

[Joh03] Peter T. Johnstone. *Sketches of an Elephant: A Topos Theory Compendium*, volume 43 and 44 of *Oxford Logic Guides*. OUP, 2003.

[KA10] Chantal Keller and Thorsten Altenkirch. Hereditary substitutions for simple types, formalized. In *Proceedings of the third ACM SIGPLAN workshop on Mathematically structured functional programming*, pages 3–10. ACM, 2010.

[MM92] Saunders Mac Lane and Ieke Moerdijk. *Sheaves in Geometry and Logic: A First Introduction to Topos Theory*. Universitext. Springer, 1992.

[Pit13] Andrew M. Pitts. *Nominal Sets: Names and Symmetry in Computer Science*. Cambridge University Press, May 2013.

[Urb08] Christian Urban. Nominal reasoning techniques in Isabelle/HOL. *Journal of Automatic Reasoning*, 40(4):327–356, 2008.

[Wad89] Philip Wadler. Theorems for free! In *Proceedings of the 4th International Conference on Functional Programming Languages and Computer Architecture*, pages 347–359. ACM, 1989.

[WCPW03] Kevin Watkins, Iliano Cervesato, Frank Pfenning, and David Walker. A concurrent logical framework: The propositional fragment. In *International Workshop on Types for Proofs and Programs*, pages 355–377. Springer, 2003.

This work is licensed under the Creative Commons Attribution-NoDerivs License. To view a copy of this license, visit http://creativecommons.org/licenses/by-nd/2.0/ or send a letter to Creative Commons, 171 Second St, Suite 300, San Francisco, CA 94105, USA, or Eisenacher Strasse 2, 10777 Berlin, Germany