Substitution for Fraenkel-Mostowski foundations

Murdoch J. Gabbay[1] and **Michael J. Gabbay**[2]

Abstract. A fundamental and unanalysed logical concept is *substitution*. This seemingly innocuous operation — substituting a variable for a term or valuating a variable to an element of a domain — is hard to characterise other than by concrete constructions. It is widely viewed as a technicality to be dispensed with on the way to studying other things. Discussions of computer science foundations, and of the philosophy of logic, have largely ignored it.

We show that Fraenkel-Mostowski set theory gives a model of variables and substitution as constructions on sets. Thus models of variables and substitution are exhibited as constructions in a foundational universe, just like models of arithmetic (the ordinals) and other mathematical entities. The door is open for classes of denotations in which variables, substitution, and evaluations are constructed directly in sets and studied independently of syntax, in ways which would previously have not been possible.

1 Introduction

Computer science evolved out of the study of logic and the foundations of mathematics of the late 19th and early 20th centuries. Part of the motivation for that study was to devise a framework to explain mathematical knowledge, as can clearly by seen in the work of Gottlob Frege [10, 11]. Frege's development of predicate calculus was, amongst other things, intended to explain the content of statements about 'an arbitrary number' without positing some special entity that is an arbitrary-number. In terms of the modern predicate calculus, the solution was that a statement about an arbitrary number has the form of an open or universally quantified sentence. So: the content of $A(x)$ or $\forall x.A(x)$ is given by the contents of $A(x)[x/t]$ for all t that denote elements of a certain domain.

This approach leaves unexplained the phenomenon of the substitution of x for a term. Substitution is not trivial: substitutions may occur within other substitutions and when a substitution is carried out, variables must be renamed to ensure that no unwanted bindings result. Therefore the explanation of knowledge of generic mathematical statements in terms of substitutions just changes the issue to the explanation of knowledge about substitution. We might ask again what exactly we know when we know that $A(x)[x/t]$ for any t.

In fact, attempts to formalise the theory of substitution show that the 'explanation' in terms of substitutions just makes matters worse. The complexity of the knowledge needing an explanation has increased, if the content of $A(x)$ has been given in terms of the more complex $A(x)[x/t]$. "What is the content of the arbitrary t in $A(x)[x/t]$?" we may ask — an answer in terms of the even more complex $A(x)[x/y][y/t]$ is of no help.

We find no relief in replacing talk of substitution with talk of valuations: this merely translates the problem into a different language; the content of $\forall x.A(x)$ is given in terms of valuations $A(x)\langle x{\mapsto}d\rangle$ for all d — but what is a valuation, and what do we know when we know the generic statement that, say $A(x)\langle x{\mapsto}d\rangle = \top$ for all valuations on x?

This problem is not confined to the philosophical question of the content of an open or universally quantified statement. All formal languages used to express functions and computations, and reasoning about functions and computations, refer to substitution of variables for terms or to the resolution of a variable to a value. An account of the content of substitution and valuation would therefore shed light on functions and computation.

But are we misguided, or asking for too much, when we ask for an explanation of substitution? After all, the syntax of a formal language is impossible to formulate without using schematic variables and substitution. Research into computer science cannot get off the ground without, at least, some recourse to formal syntax. This suggests that we must be satisfied to take substitution as an unanalysable primitive — we must either take substitution as a purely formal syntactic manipulation, or accept that the only possible explanation of substitution on the syntax of one formal language must be in terms of substitution in another 'meta' formal language (the so-called Higher-Order Abstract Syntax approach [19]). In either case, we must give up on trying to provide a foundational theory to account for substitution independently of formal syntax.

In this paper we shall show that, on the contrary, there is an independent foundation that can interpret the action of substitution and valuation. This foundation is called Fraenkel-Mostowski set theory.[3]

Fraenkel-Mostowski set theory [6, 22] (**FM** set theory) was originally developed to prove the independence of the axiom of Choice from the other axioms of Zermelo-Fraenkel set theory. It was rediscovered and used by the first author and Pitts to model abstract syntax with binding [16]. An advantage of modelling syntax in a model of FM set theory is that datatypes of syntax quotiented by α-equivalence can be modelled inductively (rather than as *quotients* by α-equivalence of syntax-without-binding). This is because FM set

[1] http://www.gabbay.org.uk

[2] michael.gabbay(at)kcl.ac.uk, Michael Gabbay gratefully acknowledges the support of the British Academy under grant PDF/2006/509.

[3] We know of no set-theoretic foundational account of substitution in the literature, besides this paper. However, there have been many attempts to axiomatise the properties that such an account should have.
Fine [9] has axiomatised 'arbitrary objects', especially investigation of *dependency between arbitrary objects*; the intuition is that both x and $2 * x$ are arbitrary objects, but they are correlated. It remains to be seen whether a model of FM set theory can be considered as a model of Fine's axioms.
Aczel's 'generalised set theory' [3] and 'universes with parameters' [4] model variable symbols (Aczel calls them *parameters*) as atoms in a ZFA-like set theory. The resemblance ends there; Aczel imposes all the structure he needs as explicit axioms on names, and the substitution action is not capture-avoiding, which is one of the most difficult technical aspects of the work in this paper. The application is also quite different; Aczel investigates coinductive structures and non-wellfounded set theory [2] as a semantics for behaviour.

theory delivers a model of variable symbols and α-abstraction [16] — these feature in this paper as atoms (Subsection 2) and atoms-abstraction (Subsection 2.4).

Unlike HOAS [19] there is no problem with 'exotic terms'; also more functions, such as α-inequality, may be expressed; finally, and there is no need in FM for levels of carefully-constrained meta-language. Unlike de Bruijn indexes [8] the reasoning and programming principles of syntax-with-binding in FM are natural and correspond very closely to informal practice. Seven years of research, culminating in an implementation of these ideas in Isabelle [23] have demonstrated the practical potential of this technique.

What makes this all take off is that the model of variable symbols and α-abstraction provided by FM set theory is applicable to all sets, including those modelling functions, predicates, domains, games, and so on. They can be applied to denotations other than sets modelling syntax. Since the introduction of these ideas [16] there now exist programming languages [21, 7], logics [20, 14], models of storage [5], and semantics of references using game theory [1] — research continues and the work all uses the model of variable symbols and α-abstraction which emerges from FM set theory.

However we usually are interested in variable symbols and α-abstraction because we want a *capture-avoiding substitution action*. In this paper we demonstrate that the variable symbols in models of FM set theory admit a substitution action defined as an operation between arbitrary sets. We also show that this substitution action avoids capture with α-abstraction. In short, any model of FM set theory is *also* a model of something that looks like 'substitution' in formal syntax, but which is valid for all sets.

We envisage denotations using FM set theory in which variables and open terms are explained directly as sets — without needing valuations — and substitution in syntax is explained directly as substitution on sets.

2 Fraenkel-Mostowski set theory

2.1 Axioms, permutations, equivariance

The language of FM set theory is first-order logic with binary predicates $=$ (set equality) and $\in$ (set membership) — like the language of ZF set theory — *and* one constant symbol $\mathbb{A}$ for 'the set of atoms'.

Definition 1. *The axioms of FM set theory are given in Figure 1.*

In Figure 1 we use standard definitional extensions of the language of sets. $\mathcal{P}_{fin}(\mathbb{A})$ is the finite powerset of $\mathbb{A}$ (the set of finite subsets of $\mathbb{A}$). 'S supports x' is described in Definition 4. The standard cumulative hierarchy model of these axioms is described in Remark 8.

We will use some notational conventions in the rest of this paper:

- An **atom** is a set member of $\mathbb{A}$ (the set of atoms).
- *The permutative convention:* $a, b, c, \ldots$ range over *distinct* atoms unless stated otherwise.
- A, B, C, S, T range over sets of atoms. For example $A \subseteq \mathbb{A}$.
- X, Y, Z, U, V range over elements that are not atoms and may be empty. For example X might equal $\emptyset$ or $\{a, \emptyset\}$, but X cannot equal a.
- x, y, z, u, v range over arbitrary elements.

Remark 2. An atom $a \in \mathbb{A}$ is 'empty' ($\forall x.x \notin a$) but not equal to $\emptyset$. $(\mathbf{Extensionality})$ is weakened so that an empty element is equal to $\emptyset$, or is an atom.

Note that $(\mathbf{AtmInf})$ insists that there are infinitely many atoms.

2.2 Atoms, equivariance and support

Write $(a\ b)$ for the **swapping** function from atoms to atoms:

$$(a\ b)(a) = b \qquad (a\ b)(b) = a \qquad (a\ b)(c) = c.$$

By our permutative convention, a, b, and c are distinct.

Let π range over functions generated by composing finitely many swappings, call these functions **permutations**. Write $\circ$ for functional composition and π^{-1} for the inverse of π, which is also a permutation. The action of permutations extends to all sets by ϵ-induction [18]:

$$\pi X = \{\pi x \mid x \in X\}.$$

Let $\phi(x_1, \ldots, x_n)$ range over predicates in the language of FM set theory that mention variables in $x_1, \ldots, x_n$. An n-rary function $F(x_1, \ldots, x_n)$ can be expressed by an $n{+}1$-ary predicate $\phi_F(x_1, \ldots, x_n, z)$ such that for each $x_1, \ldots, x_n$ there is a unique z making ϕ_F true. Then **equivariance** is the following two properties:

Theorem 3. *$\phi(x_1, \ldots, x_n) \Leftrightarrow \phi(\pi x_1, \ldots, \pi x_n)$, and $\pi(F(x_1, \ldots, x_n)) = F(\pi x_1, \ldots, \pi x_n)$ always hold.*

Proof. The first part is by an easy induction on the syntax of ϕ. We consider just one case: $x \in y$ implies $\pi x \in \pi y$ follows directly from the fact that $\pi y = \{\pi y' \mid y' \in y\}$. The reverse implication uses π^{-1}.

The second part follows using the standard encoding of an n-ary function as an $n{+}1$-ary predicate. □

Equivariance (Theorem 3) holds because atoms have no internal set structure. It is a useful source of one-line proofs [14, 12]; we shall exploit that in this paper. Equivariance is also a sense in which atoms are 'abstract': if we pick some sets, containing some specific atoms, and prove a property of them, then that property is as true of the sets with the atoms permuted; the identity of atoms only matters up to permutations.

2.3 Support

Definition 4. *If $S \subseteq \mathbb{A}$ write $\mathsf{fix}(S) = \{\pi \mid \forall a \in A.\pi(a) = a\}$.*
*Say that $S \subseteq \mathbb{A}$ **supports** x when $\forall \pi \in \mathsf{fix}(A).\pi x = x$.*
*Define $supp(x)$ the **support** of x by:*

$$supp(x) = \bigcap\{S \mid S \text{ is finite, } S \text{ supports } x\}.$$

$supp(x)$ always exists in FM set theory because $(\mathbf{Fresh})$ insists that a finite S supporting x exists. Write $a\#x$ when $a \notin supp(x)$. Read this 'a is **fresh for** x'. We may write $a\#t_1, t_2$ for '$a\#t_1$ and $a\#t_2$', and so on.

Remark 5. For example:

- $supp(\emptyset) = \emptyset$. $\quad \pi\emptyset = \emptyset$ for all $\pi \in \mathsf{fix}(\emptyset)$.
- $supp(\mathbb{A}) = \emptyset$.
  $\pi\{a, b, c, \ldots\} = \{\pi(a), \pi(b), \pi(c), \ldots\} = \{a, b, c, d, \ldots\}$ for all $\pi \in \mathsf{fix}(\emptyset)$.
- $supp(a) = \{a\}$. $\quad \pi(a) = a$ for all $\pi \in \mathsf{fix}(\{a\})$.
- $supp(\{a\}) = \{a\}$. $\quad \pi(\{a\}) = \{a\}$ for all $\pi \in \mathsf{fix}(\{a\})$.
- $supp(\mathbb{A} \setminus \{a\}) = \{a\}$.
  $\pi\{b, c, d, \ldots\} = \{\pi(b), \pi(c), \pi(d), \ldots\} = \{b, c, d, \ldots\}$ for all $\pi \in \mathsf{fix}(\{a\})$.
- $supp(\{a, b\}) = \{a, b\}$. $\quad \pi\{a, b\}{=}\{a, b\}$ for all $\pi{\in}\mathsf{fix}(\{a, b\})$.

$$(\textbf{Sets})\quad \forall x.(\exists y.y \in x) \Rightarrow x \notin \mathbb{A} \qquad (\textbf{Extensionality})\quad \forall x.x \notin \mathbb{A} \Rightarrow x = \{z \mid z \in x\}$$
$$(\textbf{Comprehension})\quad \forall x.\exists y.y \notin \mathbb{A} \wedge y = \{z \in x \mid \phi(z)\} \quad (y \text{ not free in } \phi) \qquad (\epsilon\textbf{-Induction})\quad \big(\forall x.(\forall y \in x.\phi(y)) \Rightarrow \phi(x)\big) \Rightarrow \forall x.\phi(x)$$
$$(\textbf{Replacement})\quad \forall x.\exists z.z \notin \mathbb{A} \wedge z = \{F(y) \mid y \in x\} \qquad (\textbf{Pairset})\quad \forall x, y.\exists z.z = \{x, y\}$$
$$(\textbf{Union})\quad \forall x.\exists z.z \notin \mathbb{A} \wedge z = \{y \mid \exists y'.(y \in y' \wedge y' \in x)\} \qquad (\textbf{Powerset})\quad \forall x.\exists z.z = \{y \mid y \subseteq x\}$$
$$(\textbf{Infinity})\quad \exists x.\emptyset \in x \wedge \forall y.y \in x \Rightarrow y \cup \{y\} \in x \qquad (\textbf{AtmInf})\quad \mathbb{A} \notin \mathcal{P}_{fin}(\mathbb{A}) \qquad (\textbf{Fresh})\quad \forall x.\exists S \in \mathcal{P}_{fin}(\mathbb{A}).S \text{ supports } x$$

Figure 1. Axioms of FM set theory

- $supp(\mathbb{A} \setminus \{a, b\}) = \{a, b\}$.
  $\pi\{c, d, e, \ldots\} = \{\pi(c), \pi(d), \pi(e), \ldots\} = \{c, d, e, \ldots\}$ for all $\pi \in \mathsf{fix}(\{a, b\})$.
- $supp(\{a, \{a\}, \{c\}, \{d\}, \ldots\}) = \{a, b\}$.

$$\begin{aligned}\pi(\{a,\{a\},\{c\},\{d\},\ldots\}) &= \{\pi(a),\{\pi(a)\},\{\pi(c)\},\{\pi(d)\},\ldots\}\\ &= \{a,\{a\},\{c\},\{d\},\ldots\}\end{aligned}$$

  provided that $\pi \in \mathsf{fix}(\{a, b\})$.

Remark 6. Ideas from syntax match ideas from FM sets as follows: *variable symbols* matches *atoms* and *free variables* matches *support*. Of course, it is possible to take the complement of a set, but not possible to take the complement of a syntax tree. It is therefore important to understand that sets are more general than syntax, and in particular that $a \notin X$ and $a\#X$ are *not* the same thing. $supp(x)$ measures how 'conspicuous' a is in x, either by its set-membership or *lack* of set membership. For example:

$$a \in \mathbb{A} \text{ and } a\#\mathbb{A} \qquad a \notin \emptyset \text{ and } a\#\emptyset \qquad a \notin a \text{ and } a \in supp(a)$$
$$a \in \{a\} \text{ and } a \in supp(\{a\}) \qquad a \notin \mathbb{A}\setminus\{a\} \text{ and } a \in supp(\mathbb{A}\setminus\{a\})$$

Remark 7. Not every collection has finite support. $\{a, c, e, g, \ldots\}$ (the set of 'every other atom') is not finitely supported, and is excluded from the cumulative hierarchy model of Remark 8 below. There is no finite $S \subseteq \mathbb{A}$ such that if $\pi \in \mathsf{fix}(S)$ then $\pi\{a, c, e, g, \ldots\} = \{a, c, e, g, \ldots\}$.

Remark 8. FM is a theory in first-order logic. As is often the case, we have a clear intuition in mind for a standard model; the *cumulative hierarchy* model is the collection $\mathcal{U}$ defined as follows:

$$\begin{aligned}\mathcal{U}_0 &= \mathbb{A}\\ \mathcal{U}_{i+1} &= \mathcal{U}_i \cup \{X \subseteq \mathcal{U}_i \mid X \text{ has a finite supporting set}\}\end{aligned}$$

Then $\mathcal{U} = \bigcup_i \mathcal{U}_i$. The reader can imagine all our constructions taking place in this model and no harm will come of it.

Theorem 9. *If S and T support x and are finite, then so does $S \cap T$. As a corollary, $supp(x)$ is the unique smallest set supporting x.*

Proof. The corollary follows by calculations and by (**Fresh**).

Suppose κ fixes $S \cap T$ pointwise. We must show $\kappa x = x$.

Write K for $\{a \mid \kappa(a) \neq a\}$. Choose an injection ι of $T \setminus S$ into $\mathbb{A} \setminus (S \cup T \cup K)$ (we can say 'ι freshens $T \setminus S$'). Let $\pi(a) = \iota(a)$ and $\pi(\iota(a)) = a$ for $a \in T \setminus S$, and $\pi(a) = a$ otherwise. Note that $\pi \circ \pi = \mathbf{Id}$, so $\pi = \pi^{-1}$. π fixes S pointwise so $\pi x = x$. Also $\pi \circ \kappa \circ \pi$ fixes T pointwise so $(\pi \circ \kappa \circ \pi)x = x$. We apply π to both sides and simplify and conclude that $\kappa x = x$ as required. □

Theorem 9 says πx depends *only* on the values of π on atoms in $supp(x)$. Support goes back to Fraenkel and Mostowski [17, Chapter 4]; applications in computer science followed later [16, 12].

Theorem 10. *S supports x if and only if πS supports πx. As a corollary, $\pi supp(x) = supp(\pi x)$.*

Proof. From Theorem 3. □

A calculation cannot 'create support' not in its inputs:

Theorem 11. $supp(F(x_1, \ldots, x_n)) \subseteq supp(x_1) \cup \cdots \cup supp(x_n)$.

Proof. If $\pi \in \mathsf{fix}(\bigcup supp(x_i))$ then $\pi \in \bigcap \mathsf{fix}(x_i)$. By Theorem 3 $\pi F(x_1, \ldots, x_n) = F(\pi x_1, \ldots, \pi x_n)$. The result follows. □

2.4 α-abstraction in models of FM set theory

Substitution is interesting and hard to characterise because of its interaction with α-*equivalence*, itself deceptively complex. For example, we distinguish x and y in Px and Py but not in $\forall x.Px$ and $\forall y.Py$. We now show how to α-abstract an atom a in a set x. With sets, it is standard to abstract by taking an equivalence class. For example the concept 'even number' can be modelled as the collection of even numbers. Intuitively Definition 12 defines an equivalence class resulting from renaming atoms not in A, and thus α-abstracts over atoms in $supp(x)\setminus A$. This is then exploited in Definition 16.

Definition 12. *Suppose $A \subseteq \mathbb{A}$. Write*

$$u\|_A \quad \textit{for} \quad \{\pi u \mid \pi \in \mathsf{fix}(A)\}.$$

(Recall that $\mathsf{fix}(A) = \{\pi \mid \forall a \in A.\pi(a) = a\}$.) For example:

$$\{a\}\|_\emptyset = \{a,\ b,\ c,\ d,\ e,\ f,\ \ldots\} \qquad \{a\}\|_{\{a\}} = \{a\}$$
$$\{b\}\|_{\{a\}} = \{b,\ c,\ d,\ e,\ f,\ \ldots\}$$
$$\{a, b\}\|_{\{a,c\}} = \{\{a, b\},\ \{a, d\},\ \{a, e\},\ \{a, f\},\ \ldots\}$$

Since $\mathsf{fix}(A)$ is a group we have:

Lemma 13. *If $\pi \in \mathsf{fix}(A)$ then $u\|_A = (\pi u)\|_A$.*

In words: $u\|_A$ is an equivalence class of sets which are equal 'up to renaming atoms not in A'.

Theorem 14. *Suppose A is a finite set of atoms. Then:*

- *If $supp(u) \subseteq A$ then $supp(u\|_A) = supp(u)$.*
- *$supp(u\|_A) \subseteq A$ always.*

As a corollary, if $supp(u) \setminus A \neq \emptyset$ then $supp(u\|_A) = A$.

Proof.
- If $supp(u) \subseteq A$ then $u\|_A = \{u\}$. For example, $supp(a\|_{\{a\}}) = supp(\{a\}) = \{a\}$ and $supp(a) = a$.
- If $supp(u) \subseteq A$ then we use the first part. If there is some $a \in supp(u) \setminus A$ then the result follows by an easy calculation illustrated by the following example:

$$a\|_{\{b\}} = \{a,\ c,\ d,\ e,\ f,\ \ldots\} = \mathbb{A} \setminus \{b\}.$$

The corollary follows. □

Definition 15. *Write* (x,y) *for* $\{\{x\},\{x,y\}\}$ *(a set implementation of ordered pairs [18]).*

Definition 16. *Let* **atoms abstraction** *be* $[c]z = (c,z)\|_{supp(z)\setminus\{c\}}$.

Intuitively $[c]z$ is an α-equivalence class of (c,z) where c is abstracted, i.e. where we read (c,z) like '$\lambda c.z$' or '$\forall c.z$':

$$\begin{aligned}
[a]a &= \{(a,a),\ (b,b),\ (c,c),\ (d,d),\ (e,e),\ (f,f),\ \ldots\}\\
[a]\{a,b\} &= \{(a,\{a,b\}),\ (c,\{c,b\}),\ (d,\{d,b\}),\ (e,\{e,b\}),\ \ldots\}\\
[a](\mathbb{A}\backslash\{a\}) &= \{(a,\mathbb{A}\backslash\{a\}),\ (b,\mathbb{A}\backslash\{b\}),\ (c,\mathbb{A}\backslash\{c\}),\ \ldots\}\\
[a](\mathbb{A}\backslash\{a,b\}) &= \{(a,\mathbb{A}\backslash\{a,b\}),\ (c,\mathbb{A}\backslash\{c,b\}),\ (d,\mathbb{A}\backslash\{d,b\}),\ \ldots\}
\end{aligned}$$

Write U_{ab} for $\{a\}\cup\{\{a\},\{c\},\{d\},\{e\},\ldots\}$ for any a,b. Then:

$$\begin{aligned}
[a]U_{ab} &= \{(a,U_{ab}),(c,U_{cb}),(d,U_{db})(e,U_{eb}),\ldots\}\\
[c]U_{ab} &= \{(c,U_{ab}),(d,U_{ab}),(e,U_{ab}),\ldots\}
\end{aligned}$$

We can read '$[a]x$' as the binding action of '$\lambda a.x$' or '$\forall a.x$', and the sets above coresond with α-equivalence classes of FM sets. There is no *a priori* notion of λ-abstraction or universal quantification in $[a]x$; this is just α-abstraction, on FM sets.

Definition 16 agrees with the definition of $[c]z$ from [16]:

Lemma 17. $[c]z = \{(x,(x\ c)z) \mid x\in\mathbb{A},\ x\neq c,\ x\#z\}\cup\{(c,z)\}$.

2.5 Further properties of support, finite sets, and α-abstraction

Lemma 18. *1.* $supp(X) = \bigcup\{supp(x)\mid x\in X\}$ *if* X *is finite.*
2. $supp(\{x\}) = supp(x)$ *and if* $A\subseteq\mathbb{A}$ *is finite then* $supp(A) = A$.
3. $supp((x,y)) = supp(x)\cup supp(y)$.

Proof. If X is finite then $supp(X)\subseteq\bigcup\{supp(x)\mid x\in X\}$ follows by Theorem 11.

Now suppose $a\in supp(x)$ for some $x\in X$. Choose some b such that $b\#X$ and $b\#x'$ for every $x'\in X$. By Theorem 10 $supp((b\ a)x) = (b\ a)supp(x)$. Since X has no element y such that $b\in supp(y)$, we know that $(b\ a)X\neq X$ and by Theorem 9 it must be that $a\in supp(X)$.

The second part is immediate; the third is by Definition 15. □

Theorem 19. $supp([c]z) = supp(z)\setminus\{c\}$.

Proof. By part 3 of Lemma 18 $supp((c,z)) = supp(z)\cup\{c\}$. By definition $[c]z = (c,z)\|_{supp(z)\setminus\{c\}}$. The result follows by Theorem 14. □

Thus we expect $(a\ d)[a]\{a,b\} = [a]\{a,b\}$:

$$\begin{aligned}
[a]\{a,b\} &= \{(a,\{a,b\}),\ (c,\{c,b\}),\ (d,\{d,b\}),\ (e,\{e,b\}),\ \ldots\}\\
(a\ d)[a]\{a,b\} &= \{(d,\{d,b\}),\ (c,\{c,b\}),\ (a,\{a,b\}),\ (e,\{e,b\}),\ \ldots\}
\end{aligned}$$

Lemma 20. $supp(X)\subseteq\bigcup\{supp(x)\mid x\in X\}$ *need not necessarily hold if* X *is not finite.*

Proof. It suffices to give a counterexample; we give two:
$supp(\mathbb{A}) = \emptyset$ but $\bigcup\{supp(a)\mid a\in\mathbb{A}\} = \mathbb{A}$.
$supp(\mathbb{A}\backslash\{c\})=\{c\}$ but $\bigcup\{supp(a)\mid a\in\mathbb{A}\wedge a\neq c\} = \mathbb{A}\backslash\{c\}$. □

3 The substitution action

We now turn to defining an operation on sets that matches the syntactic operation of substitution. It must interact correctly with the α-abstraction of Definition 16.

Recall that a,b,c range over distinct atoms, A,B,C,S,T range over sets of atoms, x,y,z,u,v range over all elements, and X,Y,Z,U,V range over elements that are not atoms.

3.1 Axioms, naïve substitution action

Definition 21. *A* **substitution action** *on FM set theory is a function* $z[a{\mapsto}x]$ *expressed in the language of FM set theory taking an element* z*, an atom* a*, and an element* x*, and returning an element which we write as* $z[a{\mapsto}x]$*, satisfying:*

$$\begin{aligned}
(\alpha)\quad & b\#z\Rightarrow z[a{\mapsto}x] = ((b\ a)z)[b{\mapsto}x]\\
(\#{\mapsto})\quad & a\#z\Rightarrow z[a{\mapsto}x] = z\\
(\mathbf{var}{\mapsto})\quad & a[a{\mapsto}x] = x\\
(\mathbf{id}{\mapsto})\quad & z[a{\mapsto}a] = z\\
(\mathbf{abs}{\mapsto})\quad & c\#x\Rightarrow([c]z)[a{\mapsto}x] = [c](z[a{\mapsto}x])
\end{aligned}$$

If we read $a\#x$ as 'a is not free in x' and $z[a{\mapsto}x]$ as 'substitute x for a in z' then, clearly, the axioms of Definition 21 are sound for the standard syntactic model. In [13] they are also proved complete.[4]

FM set theory has notions of 'name' and 'free in', and 'abstraction'. We can therefore try to build a function which models 'capture-avoiding substitution' in the sense made precise by the axioms of Definition 21.

Definition 22 is probably what we might first consider:

Definition 22. *Define the* **naïve** *substitution action by*

$$a[a{\mapsto}x]_n = x\quad b[a{\mapsto}x]_n = b\quad Z[a{\mapsto}x]_n = \{z[a{\mapsto}x]_n\mid z\in Z\}.$$

Write $0=\emptyset$ and $i+1 = i\cup\{i\}$, and write $\mathbb{N} = \{0,1,2,3,\ldots\}$.

Lemma 23. *Naïve substitution does not satisfy* (α)*,* $(\#{\mapsto})$*, or* $(\mathbf{abs}{\mapsto})$*, and so is not a substitution action in the sense of Definition 21.*

Proof. It suffices to give counterexamples. We do this for $(\#{\mapsto})$ and $(\mathbf{abs}{\mapsto})$. We expect that $\mathbb{A}[a{\mapsto}1]_n = \mathbb{A}$ since $a\#\mathbb{A}$. We also expect that $([c]a)[a{\mapsto}1]_n = [c](a[a{\mapsto}1])$ since $c\#1$. But:

$$\begin{aligned}
\mathbb{A}[a{\mapsto}1]_n &= (\mathbb{A}\setminus\{a\})\cup\{1\}\\
([c]a)[a{\mapsto}1]_n &= \{(b,a),(c,a),(d,a),(e,a),\ldots\}[a{\mapsto}1]_n\\
&= \{(b,1),(c,1),(d,1),(e,1),\ldots\}\\
[c](a[a{\mapsto}1]_n) &= [c]1\\
&= \{(a,1),(b,1),(c,1),(d,1),(e,1),\ldots\}.
\end{aligned}$$

□

The naïve substitution action does not take into account that substitutions should be capture avoiding and does not interact property with the FM treatment of abstraction. We need a more subtle substitution action that 'unpacks' an FM set to discern the 'free' atoms and equate the 'bound' atoms. The basic units of such an unpacking are the *planes* defined in Definition 24.

3.2 The planes of a set

Definition 24. *If* $A\subseteq\mathbb{A}$ *is finite call* (u,A) *a* **plane** *in* Z *when*

- $u\|_A\subseteq Z$ *and* $A\subseteq supp(Z)$*, and*
- $u\|_A$ *is maximal in that for all* $u'\|_{A'}\subseteq Z$ *where* $A'\subseteq supp(Z)$*,*

$$u\|_A\subseteq u'\|_{A'}\quad\textit{implies}\quad u'\|_{A'} = u\|_A.$$

[4] An equivariance rule from [13] is omitted here because it is guaranteed by Theorem 3. Instead of $(\mathbf{id}{\mapsto})$ we use a rule $(\mathbf{ren}{\mapsto})$ in [13]. A proof that the two formulations are equivalent is not hard (and was observed by an anonymous referee of [13]). The proof is included in a recent work pending publication.

Write $\mathsf{plane}(Z)$ *for the collection of planes in* Z.

(u, A) is a plane in Z when A is a least subset of $supp(Z)$ such that $u\|_A \subseteq Z$. For example:

1. $(a, \{a\}) \in \mathsf{plane}(\{a\})$ and $a\|_{\{a\}} = \{a\} \subseteq \{a\}$.
2. $(a, \{\}) \notin \mathsf{plane}(\{a\})$ because $a\|_{\{\}} = \mathbb{A} \not\subseteq \{a\}$.
3. $(a, \{a\}) \notin \mathsf{plane}(\mathbb{A})$ because $\{a\} \not\subseteq supp(\mathbb{A}) = \emptyset$.
4. $(a, \{a, b\}) \notin \mathsf{plane}(\{a\})$ because $a\|_{\{a,b\}} = \{a\} = a\|_{\{a\}}$ and $\{a\} \not\subseteq \{a, b\}$.
5. $(c, \{a\}) \in \mathsf{plane}(\mathbb{A}\backslash\{a\})$ and $c\|_{\{a\}} = \mathbb{A}\backslash\{a\} \subseteq \mathbb{A}\backslash\{a\}$.
6. $(a, \{\}) \in \mathsf{plane}(\mathbb{A})$ and $a\|_{\{\}} = \mathbb{A} \subseteq \mathbb{A}$.
7. $((c, a), \{a\}) \in \mathsf{plane}([c]a)$ and $(c, a)\|_{\{a\}} = \{(x, a) \mid x \neq a\} = [c]a \subseteq [c]a$.
8. $\mathsf{plane}(\{a\} \cup \{\{a\}, \{c\}, \{d\}, \ldots\}) = \{(a, \{a\})\} \cup \{(\{x\}, \{b\}) \mid x \in \mathbb{A},\ x \neq b\}$.
   $a\|_{\{a\}} = \{a\} \subseteq \{a\} \cup \{\{a\}, \{c\}, \{d\}, \ldots\}$.
   $\{x\}\|_{\{b\}} = \{\{a\}, \{c\}, \{d\}, \ldots\} \subseteq \{a\} \cup \{\{a\}, \{c\}, \{d\}, \ldots\}$.

Definition 25. *If* $S \subseteq \mathbb{A}$ *is finite then define*

$$\mathsf{plane}_S(Z) = \{(u, A) \in \mathsf{plane}(Z) \mid supp(u) \cap S \subseteq supp(u) \cap A\}.$$

We should think of $\mathsf{plane}_S(Z)$ as the planes (u, A) in Z such that $supp(u)$ 'avoids name-clashes' with S. For example

$$(a, \{\}) \in \mathsf{plane}_{\{\}}(\mathbb{A}) \quad \text{but} \quad (a, \{\}) \notin \mathsf{plane}_{\{a\}}(\mathbb{A}) \quad \text{and}$$
$$(c, \{a\}) \in \mathsf{plane}_{\{b\}}(\mathbb{A}\backslash\{a\}) \quad \text{but} \quad (b, \{a\}) \notin \mathsf{plane}_{\{b\}}(\mathbb{A}\backslash\{a\}).$$

The planes of Z 'cover' Z in the following sense:

Lemma 26. *If* $S \subseteq \mathbb{A}$ *is finite then*

$$\bigcup\{u\|_A \mid (u, A) \in \mathsf{plane}_S(Z)\} = Z.$$

As a corollary taking $S = \emptyset$, $\bigcup\{u\|_A \mid (u, A) \in \mathsf{plane}(Z)\} = Z$.

Proof. We prove two set inclusions: The left-to-right inclusion is by construction. For the right-to-left inclusion, choose any $u \in Z$. Let $B = \{b_1, \ldots, b_k\}$ be equal to $supp(u) \setminus A$ and let $B' = \{b'_1, \ldots, b'_k\}$ be some set of entirely fresh atoms (so disjoint from $supp(u)$, A, S, and $supp(Z)$). Let $\pi = (b_1\ b'_1) \circ \ldots \circ (b_k\ b'_k)$.

By Theorem 10 we can calculate that

$$supp(\pi u) \cap S = (supp(u) \cap A) \cap S \quad \text{and}$$
$$supp(\pi u) \cap A = supp(u) \cap A.$$

Therefore $supp(\pi u) \cap S \subseteq supp(\pi u) \cap A$. Also $(\pi u)\|_A = u\|_A$ by Lemma 13, so $(\pi u, A) \in \mathsf{plane}_S(Z)$. Finally we note that $u \in (\pi u)\|_A$. □

3.3 The substitution action, with examples

We can now define the substitution action. We use Lemma 26 to view an FM set Z as a union of planes; the 'capture-avoiding' aspect of substitution is easy to manage on a 'plane-by-plane basis'.

Definition 27. *If* $A, S \subseteq \mathbb{A}$ *are finite then define*

$$A(a{\mapsto}S) = \begin{cases} (A \setminus \{a\}) \cup S & \text{if } a \in A \\ A & \text{if } a \notin A. \end{cases}$$

Definition 28. *Define the* **substitution action** $z[a{\mapsto}x]$ *and a 'helper' function* $\delta(z, a, x)$ *as follows:*

- $a[a{\mapsto}x] = x$ *and* $b[a{\mapsto}x] = b$, *and*
- *if* $Z \notin \mathbb{A}$ *then*

$$Z[a{\mapsto}x] = \bigcup\{(u[a{\mapsto}x])\|_{A(a\mapsto supp(x))\setminus\delta(u,a,x)} \mid (u, A) \in \mathsf{plane}_{supp(x)\cup\{a\}}(Z)\}$$
$$\delta(u, a, x) = (supp(u)(a{\mapsto}supp(x))) \setminus supp(u[a{\mapsto}x]).$$

We consider some examples.

1. $\{a\}[a{\mapsto}x]$. There is one plane, $(a, \{a\})$.
$$\delta(a, a, x) = \{a\}(a{\mapsto}supp(x)) \setminus supp(x) = \emptyset.$$
$$\{a\}(a{\mapsto}supp(x)) \setminus \delta(a, a, x) = supp(x) \setminus \emptyset = supp(x)$$
$$\{a\}[a{\mapsto}x] = a[a{\mapsto}x]\|_{supp(x)} = x\|_{supp(x)} = x.$$

2. $(\mathbb{A} \setminus \{a\})[a{\mapsto}x]$. One plane is $(b, \{a\})$ where $b\#x$ (the others give the same result).
$$\delta(b, a, x) = \{b\}(a{\mapsto}supp(x)) \setminus \{b\} = \emptyset$$
$$\{a\}(a{\mapsto}supp(x)) \setminus \emptyset = supp(x)$$
$$(\mathbb{A} \setminus \{a\})[a{\mapsto}x] = b\|_{supp(x)} = \mathbb{A} \setminus supp(x)$$

3. $\mathbb{A}[a{\mapsto}x]$. One relevant plane is $(b, \emptyset)$ where $b\#x$ (the others give the same result).
$$\delta(b, a, x) = \emptyset \qquad \emptyset(a{\mapsto}supp(x)) \setminus \emptyset = \emptyset$$
$$\mathbb{A}[a{\mapsto}x] = b\|_{\emptyset} = \mathbb{A}$$

4. $([c]a)[a{\mapsto}x] = \{(b, a), (c, a), (d, a), \ldots\}[a{\mapsto}x]$.
One plane is $((c, a), \{a\})$ where $c\#x$ (if $c \in supp(x)$ then $((c, a), \{a\}) \notin \mathsf{plane}_{supp(x)\cup\{a\}}([c]a)$).
We omit calculations showing that $(c, a)[a{\mapsto}x] = (c, x)$; for a general result see Theorem 32 after these examples.
$$\delta((c, a), a, x) = \{c, a\}(a{\mapsto}supp(x)) \setminus supp((c, x))$$
$$= (supp(x) \cup \{c\}) \setminus (supp(x) \cup \{c\}) = \emptyset$$
$$\{a\}(a{\mapsto}supp(x)) \setminus \emptyset = supp(x)$$
$$([c]a)[a{\mapsto}x] = (c, x)\|_{supp(x)} = [c]x$$
The other planes give the same result.

5. $U_b[a{\mapsto}\{b\}]$ where $U_b = \{a\} \cup \{\{a\}, \{c\}, \{d\}, \ldots\}$ for each b.
Two planes are $a\|_{\{a\}}$ (a plane for $\{a\}$) and $\{a\}\|_{\{b\}}$ (a plane for $\{\{a\}, \{c\}, \{d\}, \ldots\}$).
By calculations similar to the examples above, we calculate that
$$a[a{\mapsto}\{b\}] = \{b\} \quad \text{and}$$
$$\{\{a\}, \{c\}, \{d\}, \ldots\}[a{\mapsto}\{b\}] = \{\{a\}, \{c\}, \{d\}, \ldots\}$$
and that $U_b[a{\mapsto}\{b\}] = U$ where we write
$$U = \{\{a\}, \{b\}, \{c\}, \{d\}, \ldots\}.$$
The other planes give the same results.

6. $([c]U_b)[a{\mapsto}\{b\}] = \{(c, U_b), (d, U_b), \ldots\}[a{\mapsto}\{b\}]$.
One plane is $((c, U_b), \{a, b\})$ (the other planes give the same result). Note that $supp(U) = \emptyset$ and $supp((c, U)) = \{c\}$, so that
$$\delta((c, U_b), a, \{b\}) = \{a, b, c\}(a{\mapsto}\{b\}) \setminus \{c\} = \{b\}$$
$$\{a, b\}(a{\mapsto}\{b\}) \setminus \{b\} = \emptyset$$
$$([c]U_b)[a{\mapsto}\{b\}] = (c, U)\|_{\emptyset} = [c]U.$$

In all other examples δ is equal to $\emptyset$. Here, we see how δ is not equal to $\emptyset$. This corrects for the fact that $supp(U_b[a{\mapsto}\{b\}]) \neq supp(U_b)(a{\mapsto}\{b\})$.

Remark 29. Suppose that $A, S \subseteq \mathbb{A}$ are finite. Note that $A(a{\mapsto}S)$ and $A[a{\mapsto}S]$ do not coincide. For example, $\{a\}(a{\mapsto}\{a\}) = \{a\}$ whereas $\{a\}[a{\mapsto}\{a\}] = \{\{a\}\}$.

Lemma 30. $supp(z[a{\mapsto}x]) \subseteq supp(z)(a{\mapsto}supp(x))$.

Proof. By Theorem 11 $supp(z[a{\mapsto}x]) \subseteq supp(z) \cup \{a\} \cup supp(x)$.

Choose some fresh b (so $b\#z, a, x$). By the axiom (α) $z[a{\mapsto}x] = ((b\ a)z)[b{\mapsto}x]$. By Theorem 11

$$supp(((b\ a)z)[b{\mapsto}x]) \subseteq supp((b\ a)z) \cup \{b\} \cup supp(x).$$

The result follows using Theorem 10. □

Lemma 31. $supp(z[a{\mapsto}x]) \supseteq supp(z)(a{\mapsto}supp(x))$ *need not necessarily hold.*

Proof. A counterexample is $U_b[a{\mapsto}\{b\}]$ above. □

In the terminology of Definition 22, the substitution action is naïve on finite sets:

Theorem 32. *If $Z \notin \mathbb{A}$ and Z is* finite *then* $Z[a{\mapsto}x] = \{z[a{\mapsto}x] \mid z \in Z\}$.

Proof. By definition,

$$Z[a{\mapsto}x] = \bigcup\{(u[a{\mapsto}x])\|_{A(a{\mapsto}supp(x))\setminus\delta(u,a,x)} \mid (u, A) \in \mathsf{plane}_{supp(x)\cup\{a\}}(Z)\}$$

Suppose $(u, A) \in \mathsf{plane}_{supp(x)\cup\{a\}}(Z)$. Since $u\|_A \subseteq Z$ and Z is finite, $u\|_A$ is finite. It follows by part 1 of Lemma 18 that $supp(u) \subseteq A$ and $u\|_A = \{u\}$.

By Lemma 30 $supp(u[a{\mapsto}x]) \subseteq supp(u)(a{\mapsto}supp(x))$, so

$$\delta(u, a, x) = (supp(u)(a{\mapsto}supp(x))) \setminus supp(u[a{\mapsto}x]).$$

It follows by set calculations that

$$supp(u[a{\mapsto}x]) \subseteq A(a{\mapsto}supp(x)) \setminus \delta(u, a, x)$$

and the result follows. □

3.4 The substitution action is a substitution action

We now sketch how substitution satisfies (α), $(\#{\mapsto})$, $(\mathbf{var}{\mapsto})$, $(\mathbf{id}{\mapsto})$, and $(\mathbf{abs}{\mapsto})$, from Definition 21.

Theorem 33 is a useful technical result:

Theorem 33. *$b\#Z$ if and only if for all $(u, A) \in \mathsf{plane}(Z)$ it is the case that $b \notin A$.*

As a corollary $supp(Z) = \bigcup\{A \mid (u, A) \in \mathsf{plane}(Z)\}$.

We may use this result in a slightly different form where we write $b \notin A$ instead of $b\#A$; by part 2 of Lemma 18 these are equivalent.

Proof. By definition if $(u, A) \in \mathsf{plane}(Z)$ then $A \subseteq supp(Z)$. The left-to-right implication follows.

Now suppose that $b\#A$ for every $(u, A) \in \mathsf{plane}(Z)$. Choose any fresh $b'\#Z$. By the first part of this result, $b'\#A$ for every $(u, A) \in \mathsf{plane}(Z)$. Using part 1 of Lemma 26 we reason as follows:

$$\begin{array}{lcl} (b'\ b)Z & = & (b'\ b)\bigcup\{(u\|_A) \mid (u, A) \in \mathsf{plane}(Z)\} \\ & = & \bigcup\{(b'\ b)(u\|_A) \mid (u, A) \in \mathsf{plane}(Z)\} \\ & \overset{\text{Theorem 9}}{=} & \bigcup\{u\|_A \mid (u, A) \in \mathsf{plane}(Z)\} \\ & = & Z \end{array}$$

Now $b \notin supp((b'\ b)Z)$ by Theorem 10 and the fact that $b'\#Z$. The result follows. □

For Theorem 36 we need a technical 'capture-avoidance' result:

Lemma 34. *Suppose that $Z \notin \mathbb{A}$, $a \in \mathbb{A}$, and x is any element.*

Suppose that $B = \{b_1, \ldots, b_n\}$ is a finite set of fresh atoms (so $b_i\#x, Z$ for $1 \le i \le n$). Then

$$Z[a{\mapsto}x] = \bigcup\{(u[a{\mapsto}x])\|_{A(a{\mapsto}supp(x))\setminus\delta(u,a,x)} \mid (u, A) \in \mathsf{plane}_{supp(x)\cup\{a\}\cup B}(Z)\}.$$

Notice the B on the far right subscript.

Proof. A routine calculation demonstrates that if $(u, A) \in \mathsf{plane}(Z)$ and $supp(u)$ 'clashes' with atoms in B, then we can find a $\pi \in \mathsf{fix}(A)$ such that $supp(\pi u)$ does not 'clash' with atoms in B; by Lemma 13 the result follows using Lemma 26. □

Lemma 35. *If (for all b, if $b\#z, x$ then $z[a{\mapsto}x] = ((b\ a)z)[b{\mapsto}x]$), then also (for all b, if $b\#z$ then $z[a{\mapsto}x] = ((b\ a)z)[b{\mapsto}x]$).*

Proof. Choose fresh c (so $c\#z$; also $c\#a, b$ since by our permutative convention c, a, b are distinct atoms). By assumption

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

The result follows by Theorem 9. □

Theorem 36 $((\alpha))$**.** *$Z[a{\mapsto}x] \subseteq ((b\ a)Z)[b{\mapsto}x]$ if $b\#Z$.*
As a corollary, for any z if $b\#z$ then $z[a{\mapsto}x] = ((b\ a)z)[b{\mapsto}x]$.

Proof. We first prove the corollary. Suppose $b\#z$; there are two cases depending on whether $z \in \mathbb{A}$:

- Suppose $z \in \mathbb{A}$. Then there are three subcases:
  - (i) $z = a$. $z[a{\mapsto}x] = a[a{\mapsto}x] = x = ((b\ a)a)[b{\mapsto}x] = x$.
  - (ii) $z = b$. This contradicts $b\#z$ so there is nothing to prove.
  - (iii) $z = c$ (where $c \notin \{a, b\}$). $c[a{\mapsto}x] = c = ((b\ a)c)[b{\mapsto}x]$.
- Suppose $Z \notin \mathbb{A}$. By the first part, $Z[a{\mapsto}x] \subseteq ((b\ a)Z)[b{\mapsto}x]$. Also by Theorem 10 $a\#(b\ a)Z$ and it follows that $((b\ a)Z)[b{\mapsto}x] \subseteq ((b\ a)(b\ a)Z)[a{\mapsto}x]$. The result follows, since $(b\ a)(b\ a)Z = Z$.

We now prove by ϵ-induction that $Z[a{\mapsto}x] \subseteq ((b\ a)Z)[b{\mapsto}x]$.

Suppose $Z \notin \mathbb{A}$ and $b\#Z$. By Lemma 35 we can assume $b\#x$. Suppose the inductive hypothesis of every $u \in Z$. We unpack the definition of substitution, using Lemma 34 to add a $\{b\}$ to the subscript on plane in the first equality (we cannot add $\{a\}$ to the subscript on plane in the second equality because we do not know $a\#x$):

$$Z[a{\mapsto}x] = \bigcup\{(u[a{\mapsto}x])\|_{A(a{\mapsto}supp(x))\setminus\delta(u,a,x)} \mid (u, A) \in \mathsf{plane}_{supp(x)\cup\{a,b\}}(Z)\}$$

$$((b\ a)Z)[b{\mapsto}x] = \bigcup\{(u'[b{\mapsto}x])\|_{A'(b{\mapsto}supp(x))\setminus\delta(u',b,x)} \mid (u', A') \in \mathsf{plane}_{supp(x)\cup\{b\}}((b\ a)Z)\}$$

Suppose $(u, A) \in \mathsf{plane}_{supp(x)\cup\{a,b\}}(Z)$. To prove our set inclusion we exhibit $(u', A') \in \mathsf{plane}_{supp(x)\cup\{b\}}((b\ a)Z)$ such that

$$u[a{\mapsto}x]\|_{A(a{\mapsto}supp(x))\setminus\delta(u,a,x)} = u'[b{\mapsto}x]\|_{A'(b{\mapsto}supp(x))\setminus\delta(u',b,x)}.$$

We choose $u' = (b\ a)u$ and $A' = (b\ a)A$. By Theorem 3 we have $(u', A') \in \mathsf{plane}((b\ a)Z)$. Also by definition of $\mathsf{plane}_{supp(x)\cup\{a,b\}}(Z)$ we know that

$$supp(u) \cap (supp(x) \cup \{a, b\}) \subseteq supp(u) \cap A.$$

Now $b \notin A$ by Theorem 33 and $b \in supp(x) \cup \{a, b\}$. Therefore $b\#u$. It is now not hard to use Theorem 10 and some elementary set calculations to calculate that

$$supp(u') \cap (supp(x) \cup \{b\}) \subseteq supp(u') \cap A'.$$

So $(u', A') \in \mathsf{plane}_{supp(x)\cup\{b\}}(Z)$. Also since $b\#u$ by the inductive hypothesis $u'[b{\mapsto}x] = u[a{\mapsto}x]$.

It remains to show

$$A(a{\mapsto}supp(x))\backslash\delta(u, a, x) = ((b\,a)A)(b{\mapsto}supp(x))\backslash\delta((b\,a)u, b, x).$$

Recall that $b \notin A$. Then $A(a{\mapsto}supp(x)) = ((b\,a)A)(b{\mapsto}supp(x))$ is easily verified. Also

$$\delta((b\,a)u, b, x) = supp((b\,a)u)(b{\mapsto}supp(x))\backslash supp(((b\,a)u)[b{\mapsto}x]).$$

Now $supp((b\,a)u)(b{\mapsto}supp(x)) = supp(u)(a{\mapsto}supp(x))$ is easily verified, and $supp(((b\,a)u)[b{\mapsto}x]) = supp(u[a{\mapsto}x])$ follows by the inductive hypothesis. The result follows. □

Theorem 37 $((\#{\mapsto}))$**.** *For all $a \in \mathbb{A}$, if $a\#z$ then $z[a{\mapsto}x] = z$.*

Proof. We work by ϵ-induction. The interesting case is when $Z \notin \mathbb{A}$ (we adhere to our convention and write capital Z) and $a\#Z$. Suppose the inductive hypothesis of all $u \in Z$. By definition

$$Z[a{\mapsto}x] = \bigcup\{(u[a{\mapsto}x])\|_{A(a{\mapsto}supp(x))\backslash\delta(u,a,x)} \mid (u, A) \in \mathsf{plane}_{supp(x)\cup\{a\}}(Z)\}.$$

For any $(u, A) \in \mathsf{plane}_{supp(x)\cup\{a\}}(Z)$ by assumption

$$supp(u) \cap (supp(x) \cup \{a\}) \subseteq supp(u) \cap A.$$

By Theorem 33 $a\#A$, so $a\#u$ and by inductive hypothesis $u[a{\mapsto}x] = u$. Now $A(a{\mapsto}supp(x)) = A$, and

$$\begin{aligned}\delta(u, a, x) &= supp(u)(a{\mapsto}supp(x)) \setminus supp(u[a{\mapsto}x]) \\ &= supp(u) \setminus supp(u) = \emptyset \quad \text{and} \\ Z[a{\mapsto}x] &= \bigcup\{u\|_A \mid (u, A) \in \mathsf{plane}_{supp(x)\cup\{a\}}(Z)\}.\end{aligned}$$

The result follows by part 2 of Lemma 26. □

Theorem 38 $((\mathbf{abs}{\mapsto}))$**.** *If $c\#x$ then $([c]z)[a{\mapsto}x] = [c](z[a{\mapsto}x])$.*

Proof. If $a\#z$ then by Theorem 11 also $a\#[c]z$ and

$$([c]z)[a{\mapsto}x] = [c]z \qquad \text{and} \qquad [c](z[a{\mapsto}x]) = [c]z$$

follow by Theorem 37. So suppose $a \in supp(z)$. We sketch the rest of the proof: It is a fact that

$$((c, z), supp(z) \setminus \{c\}) \in \mathsf{plane}([c]z).$$

The other planes add nothing to the final result. So

$$\begin{aligned}[c]z &= (c, z)\|_{supp(z)\setminus\{c\}} \\ ([c]z)[a{\mapsto}x] &= (c, z[a{\mapsto}x])\|_{(supp(z)\setminus\{c\})(a{\mapsto}supp(x))\backslash\delta((c,z),a,x)} \\ [c](z[a{\mapsto}x]) &= (c, z[a{\mapsto}x])\|_{supp(z[a{\mapsto}x])\setminus\{c\}}.\end{aligned}$$

It suffices to verify that

$$(supp(z)\backslash\{c\})(a{\mapsto}supp(x))\backslash\delta((c, z), a, x) = supp(z[a{\mapsto}x])\backslash\{c\}.$$

Now

$$\delta((c, z), a, x) = (supp(z)\cup\{c\})(a{\mapsto}supp(x))\backslash(supp(z[a{\mapsto}x])\cup\{c\})$$

(we use part 3 of Lemma 18 to calculate the support of a pairset). The result follows by set calculations. □

Theorem 39 $((\mathbf{id}{\mapsto}))$**.** $z[a{\mapsto}a] = z$.

Proof. By an easy inductive argument which we sketch. The interesting case is of $Z \notin \mathbb{A}$ where we suppose $u[a{\mapsto}a] = u$ for all $u \in Z$ (we adhere to our convention and write capital Z). By definition

$$Z[a{\mapsto}a] = \bigcup\{(u[a{\mapsto}a])\|_{A(a{\mapsto}\{a\})\setminus\delta(u,a,a)} \mid (u, A){\in}\mathsf{plane}_{\{a\}}(Z)\}.$$

This easily simplifies using the inductive hypothesis to

$$Z[a{\mapsto}a] = \bigcup\{u\|_A \mid (u, A) \in \mathsf{plane}_{\{a\}}(Z)\}$$

and we use Lemma 26. □

Theorem 40. *Definition 28 is equivariant and satisfies (α), $(\#{\mapsto})$, $(\mathbf{var}{\mapsto})$, $(\mathbf{id}{\mapsto})$, and $(\mathbf{abs}{\mapsto})$ from Subsection 3.1.*

Proof. Equivariance is automatic by Theorem 3. $(\mathbf{var}{\mapsto})$ is direct from the definition. Each of (α), $(\#{\mapsto})$, $(\mathbf{id}{\mapsto})$, and $(\mathbf{abs}{\mapsto})$ is by one of the theorems proved above. □

3.5 Substitution and abstract syntax

As a sanity check we prove that our substitution action extends the substitution on syntax, if we express syntax in a model of FM set theory as outlined in previous work [15]. In other words: our substitution action coincides with our expectations of what substitution does to syntax, in a sense which we make precise in Theorem 43.

Definition 41. *Let Λ be inductively defined by:*

$$\frac{a \in \mathbb{A}}{a \in \Lambda} \qquad \frac{x, y \in \Lambda}{(x, y) \in \Lambda} \qquad \frac{a \in \mathbb{A} \quad x \in \Lambda}{[a]x \in \Lambda}$$

Lemma 42. *Λ is isomorphic to λ-terms up to α-equivalence.*

Proof. This is the FM standard construction of abstract-syntax-with-binding [16] slightly modified (see Remark 44 below). □

Theorem 43. *If $z, x \in \Lambda$ then $z[a{\mapsto}x] \in \Lambda$ and $z[a{\mapsto}x]$ is equal to what we usually call 'capture-avoiding substitution of x for a in z'.*

Proof. We work by induction on Λ.

- $a[a{\mapsto}x] = x$ and $b[a{\mapsto}x] = b$.
- $(z_1, z_2)[a{\mapsto}x] = (z_1[a{\mapsto}x], z_2[a{\mapsto}x])$ from Definition 15 and Theorem 32.
- $([c]z)[a{\mapsto}x] = [c](z[a{\mapsto}x])$ providing $c\#x$ by Theorem 38. It is not hard to use part 3 of Lemma 18 and Theorem 19 to prove that $c\#x$ corresponds precisely to 'c is not free in x' when $x \in \Lambda$. □

Define $\mathsf{inl}(x) = (x, 0)$ and $\mathsf{inr}(x) = (x, 1)$.

Remark 44. Theorem 43 works generically for any datatype of syntax-with-binding. Note that we must interpret atoms as themselves and not 'wrapped up': our construction is an isomorphic version of the datatype from [16] given by:

$$\frac{a \in \mathbb{A}}{\mathsf{inl}(\mathsf{inl}(a)) \in \Lambda} \qquad \frac{x, y \in \Lambda}{\mathsf{inl}(\mathsf{inr}((x, y))) \in \Lambda} \qquad \frac{a \in \mathbb{A} \quad x \in \Lambda}{\mathsf{inr}([a]x) \in \Lambda}$$

This is not suitable for Theorem 43 because atoms are wrapped up in $\mathsf{inl}(\mathsf{inl}(a))$ and $\mathsf{inl}(\mathsf{inl}(a))[a{\mapsto}x] = \mathsf{inl}(\mathsf{inl}(x)) \neq x$. There is no canonical implementation of the tree-structure of datatypes — the FM substitution action cannot 'guess' which implementation we chose for inl and inr.

Atoms are a distinct class of elements in a model of FM set theory so it does not harm to insert them 'unwrapped' into Λ.

3.6 Commuting substitutions

It is routine to prove the usual commutativity property of substitutions. The proof is generic and would work for any datatype:

Corollary 45. *If $a\#y$ and $x, y, z \in \Lambda$ then*

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

Proof. By induction on z. Only the base case is interesting:

$$a[a{\mapsto}x][b{\mapsto}y] = x[b{\mapsto}y] = a[b{\mapsto}y][a{\mapsto}x[b{\mapsto}y]]. \qquad \square$$

For more complex sets substitutions need not commute. That is:

Lemma 46. *There exist z, a, x, b, y such that $a\#y$ and $z[a{\mapsto}x][b{\mapsto}y] \neq z[b{\mapsto}y][a{\mapsto}x[b{\mapsto}y]]$.*

Proof.
$$\begin{aligned}\{a,\{a\},\{c\},\{d\},\ldots\}[a{\mapsto}\{b\}][b{\mapsto}\{c\}] &= \{\{a\},\{b\},\{c\},\{d\},\ldots\}[b{\mapsto}\{c\}]\\ &= \{\{a\},\{b\},\{c\},\{d\},\ldots\}\end{aligned}$$

$$\begin{aligned}\{a,\{a\},\{c\},\{d\},\ldots\}[b{\mapsto}\{c\}][a{\mapsto}\{\{c\}\}] &= \{a,\{a\},\{b\},\{d\},\ldots\}[a{\mapsto}\{\{c\}\}]\\ &= \{\{a\},\{b\},\{\{c\}\},\{d\},\ldots\}\end{aligned}$$

(The planes of interest here are $a\|_{\{a\}}$, $\{a\}\|_{\{b\}}$, and $\{a\}\|_{\{c\}}$.) $\square$

Intuitively, $\{a,\{a\},\{c\},\{d\},\ldots\}$ can be read as the predicate 'is the variable a, or is $\{x\}$ where x is a variable other than c'; see the Conclusions. Predicates which reflect on their own variables cannot be expressed in standard logics such as first-order logics; non-commutativity of substitution only holds on sets which intuitively 'reflect on atoms' in which case the results obtained may depend on the order in which those atoms are substituted.

4 Conclusions

We have exhibited substitution as an operation in models of FM set theory, with the same status as 'the graph of a function', 'ordered pairs', 'ordinals', and other basic concepts of mathematics. The foundations of computer science are not set in stone, and by paying attention to them, new insights can be gained.

From the philosophical point of view the FM universe provides a basis by which we can obtain a semantics for formal languages where the structure of denotations matches the structure of syntax very closely, also for open terms. In the case that the denotations are *of* formal syntax, the two coincide as exemplified in Section 3.5.

Since our intention here is to the lay the foundations provided by FM set theory, a full semantic treatment of first order logic is beyond the scope of this paper. However we can hint at how it works. An FM model of first order logic maps the sentences $\mathtt{P}, \mathtt{Q}\ldots$ of first order logic, open or closed, into subsets of an FM set U representing the universe of discourse. The model also maps the variables $\mathtt{x}, \mathtt{y}\ldots$ of a first order language into the set $\mathbb{A}$ of atoms. Now, if a first order sentence $\mathtt{P}$ is assigned the set $y \subseteq U$ as its semantic denotation, then the sentence $\forall\mathtt{x}\mathtt{P}$ has as its denotation the set $\bigcap\{y[a{\mapsto}u] \mid u \in U\}$. This is more than just a translation of the syntactic substitution-for-all-terms operation into another language, for $[a{\mapsto}u]$ represents a particular set theoretic operation constructable from the axioms of FM set theory.

Future work is to use the substitution action above as the basis of semantics for formal languages essential to philosophy and computer science — first-order logic and the λ-calculus are two candidates. It is also possible to investigate 'rewriting on sets'; starting with investigating the unifiers of two sets.

REFERENCES

[1] S. Abramsky, D. R. Ghica, A. S. Murawski, C.-H. L. Ong, and I. D. B. Stark, 'Nominal games and full abstraction for the nu-calculus', in *LICS*, pp. 150–159. IEEE, (2004).

[2] Peter Aczel, *Non-wellfounded Set Theory*, number 14 in CSLI lecture notes, CSLI, 1988.

[3] Peter Aczel, 'Generalised set theory', *CSLI lecture notes*, **1**(58), 1–17, (1996).

[4] Peter Aczel and Rachel Lunnon, 'Universes and parameters', *CSLI lecture notes*, **2**, 3–24, (1991).

[5] Nick Benton and Benjamin Leperchey, 'Relational reasoning in a nominal semantics for storage.', in *Proc. of the 7th Int'l Conf. on Typed Lambda Calculi and Applications (TLCA)*, volume 3461 of *LNCS*, pp. 86–101, (2005).

[6] N. Brunner. 75 years of independence proofs by Fraenkel-Mostowski permutation models, 1996.

[7] James Cheney and Christian Urban, 'Alpha-prolog: A logic programming language with names, binding and alpha-equivalence', in *Proc. of the 20th Int'l Conf. on Logic Programming (ICLP 2004)*, eds., Bart Demoen and Vladimir Lifschitz, number 3132 in LNCS, pp. 269–283. Springer-Verlag, (2004).

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

[9] Kit Fine, *Reasoning with Arbitrary Objects*, Blackwell, 1985.

[10] Gottlob Frege, *The Foundations of Arithmetic*, Blackwell, Oxford, 1953. Translated by J. L. Austin.

[11] Gottlob Frege, 'Begriffsschrift, eine der Arithmetischen Nachgebildete Formelsprache des Reinen Denkens', in *From Frege to Gödel: A Source Book in Mathematical Logic, 1879-1931*, ed., J. van Heijenoort, Harvard University Press, (2002). Translated by S. Bauer-Mengelberg as 'Concept Script, a formal language of pure thought modelled upon that of arithmetic'.

[12] Murdoch J. Gabbay, *A Theory of Inductive Definitions with alpha-Equivalence*, Ph.D. dissertation, Cambridge, UK, 2000.

[13] Murdoch J. Gabbay and Aad Mathijssen, 'Capture-avoiding substitution as a nominal algebra', *Formal Aspects of Computing*, (2008). Available online.

[14] Murdoch J. Gabbay and Aad Mathijssen, 'One-and-a-halfth-order logic', *Journal of Logic and Computation*, (2008). Available online.

[15] Murdoch J. Gabbay and A. M. Pitts, 'A new approach to abstract syntax involving binders', in *14th Annual Symposium on Logic in Computer Science*, pp. 214–224. IEEE Computer Society Press, (1999).

[16] Murdoch J. Gabbay and A. M. Pitts, 'A new approach to abstract syntax with variable binding', *Formal Aspects of Computing*, **13**(3–5), 341–363, (2001).

[17] Thomas Jech, 'Set theory', in *The Stanford Encyclopedia of Philosophy*, ed., Edward N. Zalta, (Fall 2002).

[18] P. T. Johnstone, *Notes on logic and set theory*, Cambridge University Press, 1987.

[19] D. Miller, 'Abstract syntax for variable binders: An overview', *Lecture Notes in Articial Intelligence*, **1861**, 239–253, (July 2000).

[20] A. M. Pitts, 'Nominal logic, a first order theory of names and binding', *Information and Computation*, **186**(2), 165–193, (2003).

[21] A. M. Pitts and Murdoch J. Gabbay, 'A metalanguage for programming with bound names modulo renaming', in *Proceedings of the 5th international conference on the Mathematics of Program Construction (MPC2000)*, eds., R. Backhouse and J. N. Oliveira, volume 1837 of *LNCS*, pp. 230–255. Springer, (July 2000).

[22] J. Truss, 'Permutations and the axiom of choice', in *Automorphisms of first order structures*, ed., H.D.Macpherson R.Kaye, 131–152, OUP, (1994).

[23] Christian Urban and Christine Tasson, 'Nominal techniques in Isabelle/HOL', in *CADE 2005*, volume 3632 of *Lecture Notes in Artificial Intelligence*, pp. 38–53, (2005).