# What are variables of first-order logic and the lambda-calculus?

Murdoch J. Gabbay and Michael J. Gabbay

1 August 2016

## Introduction

Thank you all for coming.

This talk is based on the following papers:

- [gabbay:nomua]
  “Nominal universal algebra”
  (coauthored with Aad Mathijssen).
- [gabbay:semooc]
  “Semantics out of context”
- [gabbay:repdul]
  “Representation and duality of the untyped lambda-calculus”
  (coauthored with Michael J. Gabbay).
- [gabbay:conqnf]
  “Consistency of Quine’s NF”

All available from `www.gabbay.org.uk/papers.html`
Type Control+F and search for nomuae, semooc, repdul, conqnf.

## Nominal algebra as a foundation for logical calculi

Recall first-order logic and the $\lambda$-calculus, as specified using derivation rules and reduction rules:

$$\frac{\Gamma, \phi[a{:=}t] \vdash \psi}{\Gamma, \forall a.\phi \vdash \psi}\ (\forall\mathbf{L}) \qquad \frac{\Gamma \vdash \phi \quad (a \notin fv(\phi))}{\Gamma \vdash \forall a.\phi}\ (\forall\mathbf{R})$$

$$(\lambda a.t)s = s[a{:=}t]\ (\beta) \qquad s = (\lambda a.s)a\ (a \notin fv(s))\ (\eta)$$

Axiomatisations using universal algebra (the logic of equality) are possible:

- Cylindric Algebras for first-order logic;
- Lambda-Abstraction Algebras for the $\lambda$-calculus.

I propose a new method to specify logical calculi based on nominal algebra (more on this shortly).

## Advantages of the nominal algebraic method

Before going into details, I will give you some reasons to care:

1. It's a genuinely new view of logic.
2. It's modular.
   For instance, first-order logic and the $\lambda$-calculus share a common kernel, which we'll see later in this talk.
3. Binders like $\forall$ and $\lambda$ are decoupled from substitution: quantifiers characterised simply and independently of substitution.
4. It's a basis for new semantics for first-order logic, the $\lambda$-calculus, and Quine's NF.
5. We obtain representation theorems and Stone duality constructions for first-order logic and the $\lambda$-calculus, and a consistency proof for Quine's NF (cf. my second talk at the LC, this Friday).

## Axiomatise substitution in nominal algebra

Nominal algebra [gabbay:nomuae] is like universal algebra but over nominal sets, so enriched with names, freshness, and binding.

Axiomatisation of substitution $Z[a{\mapsto}X]$:

$$\begin{array}{lrcl}
a\#Z \Rightarrow & Z[a{\mapsto}X] &=& Z \\
 & Z[a{\mapsto}a] &=& Z \\
a\#Y \Rightarrow & Z[a{\mapsto}X][b{\mapsto}Y] &=& Z[b{\mapsto}Y][a{\mapsto}X[b{\mapsto}Y]] \\
b\#Z \Rightarrow & Z[a{\mapsto}X] &=& ((b\ a)\cdot Z)[b{\mapsto}X]
\end{array}$$

A sigma-algebra is a set $\mathbb{S}$ with an operation $\sigma$ satisfying the axioms above. So 'substitution' is axiomatised as an operation

$$\sigma : \mathbb{S} \times \mathbb{A} \times \mathbb{T} \to \mathbb{S}$$

satisfying the axioms above. If $\mathbb{S} = \mathbb{T}$ then call the sigma-action termlike: $\lambda$-calculus has a termlike action $s[a{:=}t]$; first-order logic terms have a termlike action $s[a{:=}t]$; first-order predicates have an action but it's not termlike $\phi[a{:=}t]$.

## Axiomatise substitution in nominal algebra

$$\begin{array}{lrcl}
a\#Z \Rightarrow & Z[a\mapsto X] & = & Z \\
 & Z[a\mapsto a] & = & Z \\
a\#Y \Rightarrow & Z[a\mapsto X][b\mapsto Y] & = & Z[b\mapsto Y][a\mapsto X[b\mapsto Y]] \\
b\#Z \Rightarrow & Z[a\mapsto X] & = & ((b\ a)\cdot Z)[b\mapsto X]
\end{array}$$

- Freshness side-condition $a\#Z$ corresponds to saying 'if $a$ is not free in $Z$'.
- Permutation $(b\ a)\cdot Z$ corresponds to 'swap $b$ and $a$ in $Z$'.

Both have direct, natural interpretions over nominal sets.

As lemmas of the concrete syntactic model of syntactic substitution $:=$, the axioms above are often called: garbage-collection, identity, the substitution lemma, and $\alpha$-renaming.

## Axiomatise substitution in nominal algebra

We have axiomatised substitution in nominal algebra. This nominal algebra theory is significant:

- It has some obvious models, some not-so-obvious models, and some amazing models.
  *(Terms; duality-based models, below; any model of FM sets.)*
- Concrete models of first-order logic and the $\lambda$-calculus flow naturally out of the axiomatisation: "just add powersets" (see below).
- It decouples sigma from alpha.

Let me discuss this final point:

## Decoupling sigma from alpha

Let's rewrite $\alpha$-renaming as one might see it in a textbook (as a lemma, not an axiom):

$$b \notin fv(u) \Rightarrow u[a{:=}s] = u[a{:=}b][b{:=}s] \quad \text{lemma 1}$$

Contrast with nominal axiomatisation:

$$\begin{array}{lll} b\#Z \Rightarrow & Z[a{\mapsto}X] = ((b\ a)\cdot Z)[b{\mapsto}X] & \text{axiom} \\ b \notin fv(u) \Rightarrow & u[a{:=}s] = ((b\ a)\cdot u)[b{:=}s] & \text{lemma 2} \end{array}$$

If $b \notin fv(u)$ then $u[a{:=}b] = (b\ a)\cdot u$.

So modulo this easy fact, lemmas 1 and 2 are equivalent.

## Decoupling sigma from alpha

But there is a crucial difference! If we think in terms of rewriting,

- $u[a{:=}s] \to u[a{:=}b][b{:=}s]$ has two substitutions on the right, because we are managing $\alpha$-renaming using substitution which is the very thing we are trying to define.
- $u[a{:=}s] \to ((b\ a)\cdot u)[b{:=}s]$ has only one substitution on the right, because we manage $\alpha$-renaming using permutations.

In the nominal approach, alpha is decoupled from sigma.

I claim:

- This is The Right Way to do it.
- You should not specify $\alpha$-renaming using substitution.
- Use permutations and nominal techiques instead. It's just better.

## Quantifiers

Assume a nominal lattice (that is, a lattice over nominal sets).
Universal quantification can be axiomatised as follows:

$$\begin{array}{rr@{\;}c@{\;}l}
b\#X \Rightarrow & \forall a.X &=& \forall b.(b\ a)\cdot X \\
& \forall a.X &\leq& X \\
a\#X \Rightarrow & X &\leq& \forall a.X \\
& \forall a.(X \wedge Y) &=& (\forall a.X) \wedge (\forall a.Y)
\end{array}$$

- Axiomatisation above is equivalent to a universal property:

$$\forall a.X = \bigvee \{X' \leq X \mid a\#X'\}.$$

- Equivalent to 'adjoint' property:

$$\frac{Y \leq X \quad (a\#Y)}{\overline{\;Y \leq \forall a.X\;}}$$

- $\forall$ is decoupled from substitution. Second axiom above is not $\forall a.X \leq X[a \mapsto Y]$!

## The big picture (so far)

- $\alpha$ is decoupled from $\sigma$:

$$\cancel{[a]X = [b](X[a\mapsto b])} \qquad [a]X = [b](b\ a)\cdot X.$$

- $\forall$ is also decoupled from $\sigma$:

$$\cancel{\forall a.X = \bigwedge_u X[a{:=}U]} \qquad \forall a.X = \bigvee\{X' \leq X \mid a\#X'\}.$$

- $\sigma$ and $\forall$ do depend on $\alpha = \#{+}\pi$ (freshness + permutation):

$$\begin{aligned} b\#X \Rightarrow &\quad X[a\mapsto U] = ((b\ a)\cdot X)[b\mapsto U] \\ b\#X \Rightarrow &\quad \forall a.X = \forall b.(b\ a)\cdot X. \end{aligned}$$

## What are variables?

*Question:* What are variables of first-order logic?

*Answer 1:* Names in a theory that includes axioms for substitution $\sigma$, BA (Boolean Algebra: $\wedge$ and $\neg$), and $\forall$.

I'll now give you a flavour of Answer 2.

## From sigma to amgis

Assume a sigma-algebra $x, y, z, u, v \in \mathcal{X}$.
Consider its powerset $p, q \in pow(\mathcal{X})$.

Define an amgis-action on sets by:

$$x \in p[u \leftarrow\!\dashv a] \Leftrightarrow x[a \mapsto u] \in p \quad \text{so}$$
$$p[u \leftarrow\!\dashv a] = \{x \mid x[a \mapsto u] \in p\}.$$

The amgis-action is the functional preimage of the sigma-action.

(Amgis-algebras can be axiomatised, as sigma-algebras were axiomatised above. See e.g. [gabbay:semooc].)

Think of $p[u \leftarrow\!\dashv a]$ as

"$p$ reprogrammed to believe that $a$ is equal to $u$."

## From amgis back to sigma

Assume an amgis-algebra $p, q \in \mathcal{P}$.
Consider its powerset $X, Y \in pow(\mathcal{P})$.

Define an action by:

$$\begin{aligned} p \in X[a \mapsto u] &\Leftrightarrow \text{И} b.p[u \leftarrow\!\!\dashv b] \in (b\ a)\cdot X \quad \text{so} \\ X[a \mapsto u] &= \{p \mid \text{И} b.(p[u \leftarrow\!\!\dashv b] \in (b\ a)\cdot X)\}. \end{aligned}$$

The И is the new-quantifier. It means 'for a fresh name'.

This generates a sigma-algebra on $pow(\mathcal{P})$! Think of $X[a \mapsto u]$ as

"$X$ reprogrammed to believe that $a$ is equal to $u$ — then hide/bind $a$."

## Sigma and amgis are dual

So taking powersets we alternate: sigma, amgis, sigma.

If $\mathcal{X}$ has sigma then $pow(\mathcal{X})$ has amgis, and $pow(pow(\mathcal{X}))$ has sigma and is also a nominal lattice.

## Equality

Pleasingly, $pow(pow(\mathcal{X}))$ can also interpret FOL equality:

$$u{=}v = \{p{\in}pow(\mathcal{X}) \mid \text{И}c.p[u{\leftarrow\!\dashv}c] = p[v{\leftarrow\!\dashv}c]\}.$$

$\text{И}c.p[u{\leftarrow\!\dashv}c] = p[v{\leftarrow\!\dashv}c]\}$ is a congruence property: it is precisely that which is necessary to derive

$$p \in X[a{\mapsto}u] \quad \text{if and only if} \quad p \in X[a{\mapsto}v].$$

Bearing in mind that $p \in X[a{\mapsto}u] \Leftrightarrow \text{И}c.(p[u{\leftarrow\!\dashv}c] \in (c\ a){\cdot}X)$.

## A beautiful equation

If $\mathcal{X}$ is a $\sigma$-algebra then $pow(pow(\mathcal{X}))$ has rich sets structure:

- It has all the structure of a model of first-order logic.
- It is a sound and complete model of first-order logic.

So we can write the following beautiful equation

$$\text{FOL} = \sigma + \text{powersets}$$

and we can give another answer to our question:

## What are variables of first-order logic?

The paper [gabbay:semooc] states the following:

*Question:* What are variables of first-order logic?

*Answer 1:* Names in a theory that includes axioms for substitution $\sigma$, BA (Boolean Algebra: $\wedge$ and $\neg$), and $\forall$.

*Answer 2:* Names in $pow(pow(\mathcal{X}))$ where $\mathcal{X}$ has a termlike $\sigma$-action.

## Conclusions

What is really happening here?

Nominal techniques takes names as primitive. We can therefore apply standard techniques, such as algebra and taking powersets, to names.

Quantifiers are just connectives that manipulate names!

Quantifier structure flows naturally out of the nominal axioms and dually out of the nominal powersets—just as propositional structure flows naturally out of non-nominal axioms and non-nominal powersets.

I claim: a nominal universe is the natural place to study quantifiers.

Since logical calculi often have binders, a nominal universe is the natural place to study logic!

## And what about the $\lambda$-calculus?

$\lambda$ is 'just' an elaboration of the first-order case.

Assume a binary connective $\circ$ for application. Assume a right-adjoint $\multimap$, so

$$X \circ Y \leq Z \quad \Leftrightarrow \quad X \leq Y \multimap Z.$$

Then $\lambda$ is definable as

$$\lambda a.X = \forall a.(a \multimap X).$$

$\beta$-reduction and $\eta$-expansion easily derived from this one definition.

$\lambda$-structure appears from double-nominal-powersets structure over a set with a $\sigma$-action and a $\circ$.

[gabbay:repdul] does this, and derives a full Stone duality. So $\lambda$-calculus variables are: names in the presence of a suitable axiomatisation, and names in $pow(pow(\mathcal{X}))$ for any $\mathcal{X}$ with a $\sigma$-action and a magma action $\circ$.

## If there's time: closer look at quantifiers

Perhaps you're looking at the 'adjoint' characterisation:

$$a\#Y \Rightarrow Y{\leq}X \Leftrightarrow Y{\leq}\forall a.X$$

and comparing it with this, taken from p29 of "Introduction to Categorical Logic" by Awodey and Bauer, where $\varphi \leq B \times A$ and $\vartheta \leq B$:

$$\frac{\vartheta \times A \leq \varphi}{\vartheta \leq \forall_A \varphi} \qquad \text{resembles} \qquad \frac{Y \leq X \quad (a\#Y)}{Y \leq \forall a.X}$$

Crucial difference: to interpret $\vartheta$ and $\varphi$ we must have objects $A$ and $B$ to hand, and a category including arrows for substitution and so forth.

In the nominal approach to quantification, substitution is not required.

## Two places I use decoupling

- In the Stone dualities for first-order logic and the $\lambda$-calculus, the construction of points precedes the construction of the $\sigma$-action on predicates.
  In a certain critical sense, we build $\forall$ before we build $\sigma$. See Proposition 5.2.8 (eight characterisations of $\forall$) and Definition 6.1.1 of [gabbay:repdul].
- In my claimed ConNF proof, terms and predicates are interleaved (by comprehension: $\{a \mid \phi\}$ is a term if $\phi$ is a predicate).
  We can still give nominal semantics to $\forall$, even though we have 'not finished building' our universe of terms. See Figure 3 axiom (modalI) of [gabbay:conqnf].