Nominal Henkin Semantics

Murdoch J. Gabbay and Dominic P. Mulligan

Thanks to the LFMTP organisers

August 26, 2011

This paper in context

- 1999. Gabbay & Pitts nominal sets (LICS'99).
  Idea: nominal sets.
- 2003. Urban, Pitts, & Gabbay nominal terms (CSL'03).
  Idea: nominal terms.
- 2004. Fernández & Gabbay nominal rewriting (PPDP'04).
  Idea: non-trivial equalities (not just α-equivalence, not just induction).
- 2005. Gabbay & Mathijssen nominal algebra (CANS'05).
  Idea: 'real' logic over nominal terms.
- 2006. Gabbay & Mathijssen nominal algebra axiomatisations of substitution & first-order logic (ICTAC,PPDP'06).
  Idea: first-order ax. of substitution, logic & computation.

  ...
- 2011. Gabbay & Mulligan, this talk (LFMTP'11).
  Idea: Henkin models of the axioms.

Why is this interesting?

To be unfair to my own research, I'm replaying classical mathematics using a richer semantics.

Unfair since we can use this semantics to go further; e.g. in meta-programming and model theory.

But let's start at the beginning.

Nominal semantics can interpret variables as names/atoms. We obtain an alternative to Tarski-style valuations.

In Tarski style, an open term r gets denotation in $(\mathit{vars} \to U) \to U$ where U is some domain of denotation. This is higher-order, involves very large sets, and makes it hard to award variables an independent existence.

Why is this interesting?

In nominal style, an open term r gets denotation in U where U is some nominal domain of denotation. Variables map to names-with-properties.

What properties? Substitution. So the 'large higher-order' structure $(\mathit{vars} \to \text{-}) \to \text{-}$ is replaced by a 'small first-order' structure: a nominal substitution algebra $x[a \mapsto u]$.

This paper is one part of an exploration of the application of the ICTAC'06 'nominal algebra axiomatisation of substitution' paper to replace Tarski's valuations with nominal substitution algebra.

Substitution axiomatised (types elided)

$$
\begin{array}{lll}
(\textbf{Suba}) & & a^{\mathcal{J}}_{\phi}[a{\mapsto}x] = x \\
(\textbf{Sub}\#) & a\#z \Rightarrow & z[a{\mapsto}x] = z \\
(\textbf{SubApp}) & & (z' \bullet z)[a{\mapsto}x] = (z'[a{\mapsto}x]) \bullet (z[a{\mapsto}x]) \\
(\textbf{Sub}\lambda) & c\#x \Rightarrow & ([c{:}\chi]z)[a{\mapsto}x] = [c{:}\chi](z[a{\mapsto}x]) \\
(\textbf{SubId}) & & z[a{\mapsto}a^{\mathcal{J}}] = z
\end{array}
$$

This seems reasonable. Facts of substitution are now modelled, in algebraic style, as abstract operations-satisfying-axioms.

In the paper we take $z[a{\mapsto}x]$ as shorthand for $([a{:}\chi]z) \bullet x$. Thus by 'sleight of hand' we get β for free. Just a design choice.

Bit more detail; definition of a model

Assume simple types ϕ and type environments Γ.

A model $\mathcal{I}$ is an assignment for each type environment Γ and type ϕ of a finitely-supported set $[\![\phi]\!]^{\mathcal{I}}_{\Gamma}$ together with the following data:

Bit more detail; definition of a model

1. For every $a{:}\phi \in \Gamma$ an element $a^{\mathcal{J}}_{\phi} \in [\![\phi]\!]^{\mathcal{J}}_{\Gamma}$.
   Variables interpreted as names.
2. For every constant C an element $C^{\mathcal{J}} \in [\![type(C)]\!]^{\mathcal{J}}_{\Gamma}$.
   As usual.
3. If $x \in [\![\psi]\!]^{\mathcal{J}}_{\Gamma,\, a:\phi}$, an element $[a{:}\phi]x \in [\![\phi{\rightarrow}\psi]\!]^{\mathcal{J}}_{\Gamma}$.
   Nominal atoms-abstractor models function abstraction.
4. For $x \in [\![\phi{\rightarrow}\psi]\!]^{\mathcal{J}}_{\Gamma}$ and $y \in [\![\phi]\!]^{\mathcal{J}}_{\Gamma}$, an element $x \bullet y \in [\![\psi]\!]^{\mathcal{J}}_{\Gamma}$.
   Application function as usual for Henkin semantics.
5. $[\![\phi]\!]^{\mathcal{J}}_{\Gamma \cap \Gamma'} = [\![\phi]\!]^{\mathcal{J}}_{\Gamma} \cap [\![\phi]\!]^{\mathcal{J}}_{\Gamma'}$.
   Presheaves.
6. If $x \in [\![\phi]\!]^{\mathcal{J}}_{\Gamma}$ then $\mathrm{supp}(x) \subseteq dom(\Gamma)$.
   Presheaves.

We write $x[a{\mapsto}y]$ as sugar for $([a{:}\phi]x) \bullet y$. In addition, $\mathcal{J}$ is a nominal algebra for substitution, satisfying the rules of a previous slide.

Semantics

Define an interpretation mapping r to $[\![r]\!]^{\mathcal{I}}_{\Gamma}$, by induction on r:

$$
\begin{aligned}
[\![a]\!]^{\mathcal{I}}_{\Gamma} &= a^{\mathcal{I}}_{\phi} && (a{:}\phi \in \Gamma) \\
[\![C]\!]^{\mathcal{I}}_{\Gamma} &= C^{\mathcal{I}} \\
[\![\lambda a{:}\phi.r]\!]^{\mathcal{I}}_{\Gamma} &= [a{:}\phi][\![r]\!]^{\mathcal{I}}_{\Gamma} \\
[\![rs]\!]^{\mathcal{I}}_{\varsigma} &= [\![r]\!]^{\mathcal{I}}_{\Gamma} \bullet [\![s]\!]^{\mathcal{I}}_{\Gamma}
\end{aligned}
$$

Discussion

Names inhabit the denotation directly: $a^{\mathcal{J}}_{\phi} \in [\![\phi]\!]^{\mathcal{J}}_{\Gamma}$. Think of $a : \phi$ as a constant which must be interpreted 'as itself' by $a^{\mathcal{J}}_{\phi}$.

But $a^{\mathcal{J}}_{\phi}$ also behaves like a variable: It can be renamed, bound (by $[a{:}\phi]x$) and substituted for (by $z[a{\mapsto}x]$).

(**Suba**) to (**Sub**λ) do the job that valuations do in Tarski-style models. They replace a name $a^{\mathcal{J}}_{\phi}$ by an(other) element of the model.

In Tarski models we pick a valuation and then form a denotation; in nominal models we form a denotation and then—if we wish—substitute for the free variables.

Just as for Henkin models, we specify what a model must look like but do not build one (but we can using e.g. syntax or Tarski-style valuations).

What's the pay-off?

- New class of models.
- Evidence that ICTAC'06 $\approx$ Tarski-style valuations.
- Nominal Henkin models have better structure. We get ξ and well-pointedness, not true of classical algebraic axiomatisations. Inclusion of names in denotation gives extra structure which simply cannot be expressed without names.
- Since names inhabit the denotation, we can play with them. Thus: inspiration for new languages. The application in the LFMTP'11 paper is to level 2 variables in the style of nominal terms.

Level 2 variables

Nominal terms feature two kinds of names: atoms a and unknowns X. These model e.g. 'object-level variables' x and 'meta-level variables' r or ϕ.

Also related to CMTT variables, except that these are closed (though box types allow finite 'lifting') whereas nominal unknowns are open (and do not require box types). Also related to Jojgov's thesis, Sato's context calculi, and more.

Closest other work in nominal techniques: nominal terms, hierarchical nominal rewriting, lambda context-calculus, all of which extend this hierarchy to ω. If you don't know these, that's fine (but check out my webpage).

The variant of nominal terms used in the LFMTP'11 paper is permissive, meaning that we split the set of atoms into $\mathbb{A}^<$ and $\mathbb{A}^>$ and unknowns X have a permission set which is $\mathbb{A}^<$. This eliminates nominal freshness contexts; see e.g. permissive-nominal algebra (LFMTP'09).

Syntax

Terms are defined by:

$$r ::= a \mid C \mid X[b_i{:=}s_i]_{i=1}^{n} \mid \lambda c{:}\phi.s \mid rs \quad (\{b_i \mid 1{\leq}i{\leq}n\}{\subseteq}\mathbb{A}^{<},\ c{\in}\mathbb{A}^{>})$$

$[b_i{:=}s_i]$ is a (level 1) substitution, which is a finite partial function from atoms to terms, mapping b_i to s_i. We call $X[b_i{:=}s_i]_1^n$ a moderated unknown.

Clearly a generalisation of nominal terms' $\pi{\cdot}X$.

The condition $c \in \mathbb{A}^{>}$ may seem odd—so $\lambda a{:}\phi.a$ is not well-formed syntax if $a \in \mathbb{A}^{<}$—but since a is supposed to be bound we can intuitively always α-convert it to be in $\mathbb{A}^{>}$. This is a useful 'hygiene' simplification; the models don't know about it.

Examples

An incomplete term. The typing

$$a, b{:}\phi, X{:}\phi \vdash \lambda a{:}\phi.X[b{:=}a] : \phi{\to}\phi$$

where $a \in \mathbb{A}^{>}$ and $b \in \mathbb{A}^{<}$ represents an incomplete typing '$\lambda x{:}\phi.t$ where t has type ϕ'.

Without unknowns, both the incomplete and the complete terms would be represented by $\lambda a{:}\phi.fa$ for a higher-order $f : \phi{\to}\phi$ (whereas X has type ϕ).

An incomplete HOL predicate. Assume base types ι and o and constants $\dot{\Rightarrow} : o{\to}o{\to}o$ and $\dot{\forall} : (\iota{\to}o){\to}o$. The typing

$$X : o,\ \ Y : \iota,\ \ b{:}\iota \vdash (\dot{\forall}\lambda b{:}\iota.X)\dot{\Rightarrow}X[b{:=}Y] : o$$

represents an incomplete HOL predicate.

Without level 2 variables, both the incomplete and the complete terms would be represented by $b{:}\iota,\ \ f : \iota{\to}o \vdash (\dot{\forall}\lambda b{:}\iota.fa)\dot{\Rightarrow}fa$.

Semantics

We add a Tarski-style valuation for the X so that $\varsigma(X)$ is an element of the denotation.

But didn't we just eliminate Tarski-style valuations? Yes, but:

- If we read a as 'universal' and X as 'existential', then a says 'I could be replaced by anything' whereas X says 'I must be replaced by something'. That's what Tarski style valuations express. Indeed, one way of reading our Henkin semantics is as a 'universal' instead of 'existential' interpretation of variables.
- It's easy to use Tarski-style valuations because everybody is familiar with them. To see what happens if we reject them at level 2 as well, see Two-level nominal sets ($\approx$TAASN'09, MSCS'11).

Semantics

Define an interpretation mapping r to $[\![r]\!]^{\mathcal{J}}_{\varsigma;\Gamma}$, by induction on r:

$$
\begin{aligned}
[\![a]\!]^{\mathcal{J}}_{\varsigma;\Gamma} &= a^{\mathcal{J}}_{\phi} && (a{:}\phi \in \Gamma) \\
[\![C]\!]^{\mathcal{J}}_{\varsigma;\Gamma} &= C^{\mathcal{J}} \\
[\![\lambda a{:}\phi.r]\!]^{\mathcal{J}}_{\varsigma;\Gamma} &= [a{:}\phi][\![r]\!]^{\mathcal{J}}_{\varsigma;\Gamma} \\
[\![rs]\!]^{\mathcal{J}}_{\varsigma} &= [\![r]\!]^{\mathcal{J}}_{\varsigma;\Gamma} \bullet [\![s]\!]^{\mathcal{J}}_{\varsigma;\Gamma} \\
[\![X\,[b_i{:=}s_i]_i]\!]^{\mathcal{J}}_{\varsigma;\Gamma} &= \varsigma(X)\,[b_i{\mapsto}[\![s_i]\!]^{\mathcal{J}}_{\varsigma;\Gamma}]_i && (X{:}\phi \in \Gamma)
\end{aligned}
$$

We see a benefit of the nominal semantics: we hardly do any extra work; just one extra line above, for X. Adding unknowns to normal semantics is hard work. Adding them to nominal semantics is fairly easy.

Summary

We can identify three relevant waves of nominal research:

1. Nominal sets developed a universe in which names are first-class objects, with excellent mathematical properties.
2. Nominal terms languages and axiomatisations with intended denotations in this universe were then developed: nominal rewriting, nominal algebra, permissive-nominal algebra, permissive-nominal logic ...
3. Most recently, these axiomatisations have been used to explore new models and syntaxes with new and interesting structure.

The pay-off:

- Beauty.
- Improved mathematical structure.
- Ease of use: it is (relatively) easy to cook up a language and use the nominal models as a guide and very quick consistency proof.

Reference bibliography

- Nominal sets (newaas,newaas-jv).
- Nominal terms (nomu,nomu-jv).
- Nominal algebra & axiomatisations (capasn,capasn-jv,forcie,nomuae,nomalc).
- Permissive-nominal algebra (unialt,nomtnl).
- Two-level nominal sets (twolns).
- Concrete representations (this paper,stodnb,frenrs).

If interested, search my webpage `gabbay.org.uk/papers.html` for the BibTex codes.