A NEW semantics of substitution

Samuel Rota Bulò, Andrea Marin

Work supervised by Murdoch J. Gabbay

University of Venice, 1/3/2006

Capture-avoiding substitution

Capture-avoiding substitution is extremely important for mathematical computer science.

Terms Λ of the untyped λ-calculus

$$t ::= a \quad | \quad \lambda a.t \quad | \quad tt$$

(quotient by α-equivalence for simplicity).

Substitution has the usual definition, e.g.

$$(\lambda a.ba)[b \mapsto a] \equiv \lambda a'.aa'.$$

The untyped λ-calculus is canonical for a wide range of related systems (logic, rewriting, unification, matching, pi-calculus, and so on).

What is substitution?

In this talk we will give a NEW axiomatisation of substitution.

We hope these new axioms may throw new light on old and well-established systems.

More on that later.

Fix some countably infinite set of atoms $a, b, c \in \mathbb{A}$.

Substitution sets

A substitution set $\mathbb{X}$ is a tuple of:

- An underlying set $|\mathbb{X}|$, write elements $X, Y, Z \in |\mathbb{X}|$.
- A substitution action $|\mathbb{X}| \times \mathbb{A} \times |\mathbb{X}| \longrightarrow |\mathbb{X}|$.
- (Optionally) a interpretation function $\mathbb{A} \longrightarrow |\mathbb{X}|$.

If $\mathbb{X}$ has an interpretation function call it concrete.

Otherwise call it abstract.

Write the substitution action as $Z[a \mapsto X]$.

Axioms of substitution sets

$\mathbb{X}$ should satisfy:

$$(\#{\mapsto}) \qquad a\#Z \;\vdash\; Z[a{\mapsto}X] = Z$$
$$(sub{\mapsto}) \quad a\#Y \;\vdash\; Z[a{\mapsto}X][b{\mapsto}Y] = Z[b{\mapsto}Y][a{\mapsto}X[b{\mapsto}Y]]$$

If $\mathbb{X}$ is concrete it should additionally satisfy:

$$(var{\mapsto}) \qquad [\![a]\!][a{\mapsto}X] = X$$
$$(ren{\mapsto}) \quad b\#Z \;\vdash\; Z[a{\mapsto}[\![b]\!]] = (b\,a)Z$$

$[\![\text{-}]\!]$ is the interpretation function.

Freshness conditions and permutation

This is a nominal algebraic specification.

$b\#Z$ is an abstract notion of ‘free names (atoms) of’ inherited from previous work on Fraenkel-Mostowski techniques (FM techniques).

$(b\ a)Z$ is an abstract notion of ‘swap b and a in Z’. Again inherited from FM techniques.

Freshness conditions and permutation

On concrete sets these behave much as one would expect. In particular if $Z \in \Lambda$ then

$$b\#Z \quad \text{if and only if} \quad b \notin fn(Z)$$

for example $b\#\lambda b.b$ and $\neg(b\#\lambda a.b)$.

$$(b\ a)Z = Z[b{\mapsto}a,\ a{\mapsto}b]$$

(simultaneous renaming), for example $(b\ a)(ba) = ab$.

Λ is a concrete substitution set

Substitution is substitution, and the interpretation of an atom is the atom (considered as a variable symbol in Λ of course).

It suffices to validate the various axioms:

- If $a \notin fn(Z)$ then $Z[a{\mapsto}X] = Z$. ✓
- If $a \notin fn(Y)$ then $Z[a{\mapsto}X][b{\mapsto}Y] = Z[b{\mapsto}Y][a{\mapsto}X[b{\mapsto}Y]]$. ✓
- $a[a{\mapsto}X] = X$. ✓
- If $b \notin fn(Z)$ then $Z[a{\mapsto}b] = (b\ a).Z$ (since there are no bs in Z to change to as). ✓

$\mathbb{A} - list$ is a concrete substitution set

Just one more example, so you don't think this is just for the λ-calculus.

Let $l, m, n \in \mathbb{L}$ be the set of possibly empty finite lists of atoms.

Interpret $[\![a]\!]$ by $[a]$ and substitution by list insertion, e.g.

$$[abc][a \mapsto [abc]] = [abcbc].$$

Then $a\#l$ when a does not occur in l, swapping is swapping, and the axioms are very easy to check.

A set model of concrete substitution sets

It is important to generate substitution sets without syntax.

Pick any 'normal' set $\mathcal{X}$ (e.g. $\mathcal{X} = \{\top, \bot\}$ or $\mathcal{X} = \{0, 1, 2, \ldots\}$). Let

$$\begin{aligned} \mathbb{A} \Rightarrow \mathcal{X} &= \{\kappa \mid И a, b.\, \kappa(a) = \kappa(b)\} \\ (\mathbb{A} \Rightarrow \mathcal{X}) \Rightarrow \mathcal{X} &= \{\tau \mid И a.\, \forall x \in \mathcal{X}.\, \tau\kappa = \tau(\kappa\{a \mapsto x\})\}. \end{aligned}$$

Then $(\mathbb{A} \Rightarrow \mathcal{X}) \Rightarrow \mathcal{X}$ is a concrete substitution set.

A set model of concrete substitution sets

$$\mathbb{A} \Rightarrow \mathcal{X} = \{\kappa \mid \text{И}a, b.\ \kappa(a) = \kappa(b)\}$$
$$(\mathbb{A} \Rightarrow \mathcal{X}) \Rightarrow \mathcal{X} = \{\tau \mid \text{И}a.\ \forall x \in \mathcal{X}.\ \tau\kappa = \tau(\kappa\{a{\mapsto}x\})\}.$$

Here

- $\kappa\{a{\mapsto}x\}(a) = x$ and $\kappa\{a{\mapsto}x\}(b) = \kappa(b)$.
- $\text{И}a.\ \Phi(a)$ holds when
  - $\Phi(a)$ is false of some finite (possibly empty) subset of $\mathbb{A}$, BUT
  - for all b not in that finite set, $\Phi(b)$ is true.

  Read this as 'Φ is true for a new/for most atoms'.

Set model of concrete substitution sets

What this means is:

- κ is a valuation, which must for most atoms be constant ('boring') but may vary (be 'interesting') on some finite set.
- τ is a function from valuations as described to elements of $\mathcal{X}$ which only 'cares' about the values of κ on some finite set.

In particular if τ does not care about the values where κ and κ' are interesting, then $\tau\kappa = \tau\kappa'$.

Set model of concerete substitution sets

So we can set

- $[\![a]\!] = \lambda\kappa.\kappa(a)$.
- $\mu[a{\mapsto}\tau](\kappa) = \mu(\kappa\{a{\mapsto}\tau\kappa\})$.

It is a fact that these validate the axioms. We give just one example:

$$[\![a]\!][a{\mapsto}\tau](\kappa) = (\kappa\{a{\mapsto}\tau\kappa\})(a) = \tau(\kappa).$$

Checking the other axioms is very interesting — but read it in the paper!

Function spaces

Suppose $\mathbb{X}$ and $\mathbb{Y}$ are two substitution sets. Set

$$\mathbb{X} \Rightarrow \mathbb{Y} \quad = \quad \{f \mid И a, b.\, ((a\ b) f) = f\}$$

Here

- f varies over functions from the underlying set of $\mathbb{X}$ to the underlying set of $\mathbb{Y}$, and
- $((a\ b) f)(X) = (a\ b)(f((a\ b)X))$
  (the conjugation action).

Function spaces

The conjugation action is very literal in its action.

For example if $\mathbb{X} = \mathbb{Y} = \mathbb{A}$ then...

if $f = \lambda x.a$ then $(a\,b)f = \lambda x.b$,

if $f = \lambda x.\begin{cases} b & x = a \\ c & x = b \\ c & x = c \end{cases}$ then $(a\,c)f = \lambda x.\begin{cases} b & x = c \\ a & x = b \\ a & x = a \end{cases}$.

Function spaces

Function spaces $f, g \in \mathbb{X} \Rightarrow \mathbb{Y}$ form a (non-concrete) substitution set!

The substitution action is defined by:

$$(f[a{\mapsto}g])X \;=\; И a'.\,(((a'\;a)f)X)[a'{\mapsto}gX].$$

An equivalent form is:

$$\text{If} \quad a\#g, X \quad \text{then} \quad f[a{\mapsto}g](X) = (f\,X)[a{\mapsto}gX].$$

Here $a\#g$ when $И b.\,(b\;a)g = g$.

Function spaces satisfy...

$$(\#\mapsto) \qquad a\#h \vdash h[a\mapsto f] = h$$

$$(sub\mapsto) \quad a\#g \vdash h[a\mapsto f][b\mapsto g] = h[b\mapsto g][a\mapsto f[b\mapsto g]]$$

Cartesian product

If $\mathbb{X}$ and $\mathbb{Y}$ are substitution sets then $\mathbb{X} \times \mathbb{Y}$ is a substitution set, with substitution action

$$(X, Y)[a \mapsto (X', Y')] = (X[a \mapsto X'], Y[a \mapsto Y']).$$

Some interesting example functions

One-step β-reduction mapping s to s' if $s \rightarrow_\beta s'$ is in $\Lambda \Rightarrow \Lambda$.

The function -$[a \mapsto t]$ mapping s to $s[a \mapsto t]$ is in $\Lambda \Rightarrow \Lambda$.

The substitution function -$[a \mapsto$-$]$ mapping s and t to $s[a \mapsto t]$ is in $(\Lambda \times \Lambda) \Rightarrow \Lambda$.

The context function $\lambda a.$- mapping s to $\lambda a.s$ is in $\Lambda \Rightarrow \Lambda$.

So is the function

$$s \text{ maps to } C[s]$$

for general (possibly binding) contexts $C[$-$]$.

This is a nice semantics for contexts!

Some more interesting example functions

The unordered datatype $\{a.t, b.u\}$ is represented by

$$f \;=\; \text{-}[a{\mapsto}t][b{\mapsto}u]$$

mapping x to $x[a{\mapsto}t][b{\mapsto}u]$ is in $\Lambda \Rightarrow \Lambda$.

This behaves like an unordered datatype with t located at a and u located at b. To retrieve the data it suffices to apply to a or b.

In-place update is function composition. For example if $g = \text{-}[a{\mapsto}t']$ then

$$f \circ g \;=\; \text{-}[a{\mapsto}t'][b{\mapsto}u].$$

Conclusions

Substitution is a fundamental operation. Using FM and Nominal techniques we have managed to tease it apart from syntax, and also from function application.

We have investigated the functional properties of the collection of all models of substitution. It is an interesting world to explore.

Future work

Semantics for first-order logic based on substitution sets (as opposed to environment models).

I.e. instead of a predicate being a function from evaluations to truth values (like $(\mathbb{A} \Rightarrow \mathbb{D}) \Rightarrow \mathbb{B}$), let a predicate be 'just' an element of a substitution set which interprets logical connectives as well!

Semantics for the λ-calculus based on substitution sets.

More future work

Semantics for new logics and programming languages which can manipulate names in data, e.g. as do the context functions $\lambda a.$- and $C[$-$]$.

Semantics for new logics and programming languages in which elements are parameterised over context holes such as -.

For example any element of $\Lambda \Rightarrow \Lambda$ is parametarised over a context hole.

Even more future work

It is possible to build a hierarchy of substitutions, one on top of the other, using sets of atoms $\mathbb{A}^1$, $\mathbb{A}^2$, ...

Imagine for example $f : \Lambda \Rightarrow \Lambda$ as being an element of Λ with a 'stronger' hole of level 2.

In this way function spaces can be 'folded' down.

An advantage of this approach is that functional holes are now named.

For example, $abc \in \Lambda$ represents a function with 3 arguments named a, b, and c. A hierarchically constructed element $\lambda a.X$ in a version of Λ with a stronger substitution action, could represent $\lambda a.$-, and so on.