Fraenkel-Mostowski sets and the NEW model of abstraction

Murdoch J Gabbay

Tel Aviv University, Israel 28/5/2006

This talk. . .

. . . is the fourth and final talk in a mini-course describing (most!) of the mathematics I've done over the past six years.

Thank you all for coming to this talk and to the others.

Fraenkel-Mostowski

In this talk I'll discuss the Fraenkel-Mostowski semantics of abstraction, which was presented in my thesis and has been developed since then in many directions.

Write FM for Fraenkel-Mostowski.

Fraenkel and Mostowski. . .

Two mathematicians who in the early part of the 20th century proved the independence of the axiom of choice from the other axioms of set theory by producing a model which

- satisfied the other axioms of set theory
- and did not satisfy choice!

Their technique was: build a cumulative hierarchy model in the ‘normal’ way, but then choose the sets invariant under some group of permutations. Do it right, and you get another model which satisfies most axioms — but not choice, because choice may require some arbitrary choices which the permutation can detect.

FM sets

ZF sets is the dominant notion of set and is used in foundations of mathematics (apologies to Quine's New Foundations!). It is just a first-order theory with a binary predicate $\in$, and some axioms.

Easy!

A model is a collection on which we can interpret $\in$ such that the axioms are satisfied.

FM sets

Let's be concrete.

The cumulative hierarchy model:

- Take one element, write it $\emptyset$ or $\{\}$, call it the emptyset and put it in the collection.
- Take all subcollections of the collection so far, and throw those in too.
- Iterate transfinitely.

FM sets

It is possible to go insane worrying: do we not use the notion of set to build the cumulative hierarchy model, only we called it ‘a collection’.

Yes, it is well-known that a system cannot prove its own consistency. If we accept that mathematics is consistent then (probably) ZF is consistent, so it has a model, so we may proceed.

Your typical ZF set looks like this:

$$\{\{\},\{\{\}\},\{\{\},\{\{\}\}\}\}.$$

Sets are simple

It is very important to not be confused on this simple point:

- Proving consistency of a first-order theory is hard (we must show a model exists).
- Working with an arbitrary model if we know that at least one such exists so that we know we're not talking complete garbage, is easy — its structure is precisely the structure we assume of a model that it be a model.

We assume 'set inclusion' of ZF plus appropriate axioms.

So if we leave proofs of consistency to the experts we can just say:

ZF sets are just sets.

Descartes

But ZF is supposed to be a foundational theory. So what about Descartes. Is Descartes a set?

Oops.

(Descartes walks into a bar. The barkeeper says ‘Do you want a beer?’. Descartes says ‘I think not’, and disappears.)

ZFA

This motivates ZFA — just like ZF, but now we admit into the model a few extra things other than $\emptyset$.

Now we have some extra sets:

$$\{\{\text{Descartes}\}, \{\{\}\}, \{\{\}, \{\{\text{Descartes}\}\}\}\}.$$

Call these extra things atoms and asssume infinitely many of them $a, b, c, \ldots$. Introduce a new constant into the language of ZFA $\mathbb{A}$ plus axioms to make it collect atoms into a set.

Permutations

Define a swapping action on the model by:

- $(a\ b) \cdot a = b$.
- $(a\ b) \cdot b = a$.
- $(a\ b) \cdot c = c$.
- $(a\ b) \cdot x = \{(a\ b) \cdot x' \mid x' \in x\}$.

For example:

$$(a\ b) \cdot \{a, \{b\}\} = \{b, \{a\}\} \qquad (a\ b) \cdot \{a, b\} = \{a, b\}$$
$$(a\ b) \cdot \{a, c, d, \ldots\} = \{b, c, d, \ldots\}$$

What is this good for?

(Somebody always asks this; it's code for: 'I still don't see how this justifies the way I waste my time'.)

Note that $(b\ a) \cdot x = (b\ a) \cdot y$ if and only if $x = y$.

Note that $(b\ a) \cdot x \in (b\ a) \cdot y$ if and only if $x \in y$.

Note that $(b\ a) \cdot \mathbb{A} = \mathbb{A}$.

So we have the principle of equivariance. For any predicate Φ we can write in the language of ZFA:

$$\Phi(x_1, \ldots, x_n) \Leftrightarrow \Phi((b\ a) \cdot x_1, \ldots, (b\ a) \cdot x_n).$$

Power of Φ

Set theory is a foundational system.

If it's mathematical, you can encode it.

So when we say 'any Φ', this really means something.

This solved a problem in inductive reasoning

This solved a big problem in the formal (e.g. mechanised, in Isabelle) theory of inductive datatypes.

Suppose you have some inductive hypothesis $\Phi(z) = \forall b, a, x.\phi(b, z, a, x)$ where ϕ is

$$a \in \mathbb{A} \wedge b \in \mathbb{A} \wedge x \in \Lambda \wedge b \notin fv(z) \wedge b \notin fv(x) \Rightarrow b \notin fv(z[a \mapsto x]).$$

Here Λ is some sets-based implementation of a datatype such as

$$t ::= a \mid tt \mid \lambda a.t.$$

Now you want to prove $\Phi(z)$ implies $\Phi(\lambda b.z)$.

Problem with inductive principles

Unfortunately $b \in x$ so your definition of substitution tells you

$$(\lambda b.z)[a \mapsto x] = \lambda b'.((b'b) \cdot z)[a \mapsto x]$$

for some fixed but arbitrary $b' \notin fv(z, x) \cup \{a, b\}$.

($(b'\ b) \cdot z$ equals $z[b \mapsto b']$, but is more useful, see below.)

So you have $\Phi(z)$ — not $\Phi((b'\ b) \cdot z)$.

Bugger!

Ah — but you do have $\Phi((b'\ b) \cdot z)$, because of equivariance.

Lovely.

Syntax with abstraction

This simple but beautiful point formalises what we mean when we write:

> We may without loss of generality rename b to b' in $\lambda b.z$.

It becomes

> By Fraenkel-Mostowski equivariance we may assume all nice properties of $z[b \mapsto b']$ that we did of z — including our inductive hypothesis, forever and for free.

What happened to Fraenkel-Mostowski sets?

So far so good, but hang on. This happened in ZFA!

What's all this FM stuff then?

Fraenkel-Mostowski sets. . .

. . . enriches ZFA with the ‘fresh axiom’

$$\forall z.И a.\ И b.\ (b\ a)\cdot z = z$$

Here a is an atom and z is any set.

$И a.\ \phi(a)$ means:

- Perhaps $\neg\phi(a)$ for some finite set S.
- However, $\phi(a)$ holds for all atoms $a \notin S$.

The fresh axiom (detail)

$И a.\, И b.\, (b\ a) \cdot z$ unpacks as

$$\exists S_a, S_b . S_a \text{ finite} \wedge S_b \text{ finite} \wedge \forall a \in \mathbb{A} \backslash S_a . \forall b \in \mathbb{A} \backslash S_b . (b\ a) \cdot z = z.$$

Since $S_a \vee S_b$ is finite, this simplifies to:

$$\exists S . S \text{ finite} \wedge \forall a, b \in (\mathbb{A} \backslash S) . (b\ a) \cdot z = z$$

Think of this as a (powerful) generalisation of the property of syntax, that it only mentions finitely many variable symbols so you can always pick a fresh name.

Abstraction sets

This has significant mathematical repercussions.

We obtain the NEW model of abstraction, which is to α-equivalence as functional abstraction is to β-equivalence.

Just as a set Y^X is populated by graphs of functions from elements of X to elements of Y, so . . .

. . . a set $[\mathbb{A}]X$ is populated by elements $[a]x$ for $a \in \mathbb{A}$ (atoms) and $x \in X$, defined by

$$[a]x = \{(b, (b\ a) \cdot x) \quad | \quad b\#x \vee b = a\}$$

Here $b\#x$ when $И b'.\ (b'\ b) \cdot x = x$ is a notion of 'fresh for'.

Fresh for

Examples of ‘fresh for’ in action:

- $b\#\mathbb{A}$ since $(b'\ b)\cdot\mathbb{A}=\mathbb{A}$, since swapping is bijective on atoms.
- $b\#\{a\}$ since for $S=\{a\}$ and $b'\notin S$ we have $(b'\ b)\cdot\{a\}=\{a\}$.
- $\neg(a\#\{a\})$.
- $\neg(b\#\mathbb{A}\backslash\{b\})$ since $(b'\ b)\cdot(\mathbb{A}\backslash\{b\})=\mathbb{A}\backslash\{b'\}$.

So ‘fresh for’ does not imply ‘not set-included in’. Corresponds more to ‘does not occur in any distinguished way in’.

The basic theorem of abstraction sets:

$$[a]x = \{(b, (b\ a) \cdot x) \quad | \quad b\#x \vee b = a\}$$

In fact, $b\#[a]x$ if and only if $b\#x$, and $a\#[a]x$.

This exactly replicates the behaviour of $b \notin fv(z)$, with $[a]x$ corresponding to a binder.

So we can build Λ more compactly as

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

A true inductively defined datatype up to binding!

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

So $[a]a = [b]b$. This is a set equality (both are equal to $\{(x, x) \quad | \quad x \in \mathbb{A}\}$).

Yet we can ‘choose’ a name for the abstracted atom, and anything we can prove about the body of the abstraction, by equivariance we also have ‘for free’ of all renamed versions of that body, corresponding to ad hoc choosing a different name for the abstracted atom.

Further work

I implemented this all in Isabelle but the implementation went nowhere; there were problems with it which made it impractical to use within the framework of Isabelle(’s limitations).

Christian Urban and James Cheney have worked very hard to bring FM techniques to theorem-provers and logic-programming.

Andrew Pitts and Mark Shinwell have worked hard to bring FM techniques to functional programming and denotational semantics.

Cardelli and Gardner used *И* in spatial logics.

Nominal logic

Pitts wrote down a hilbert-style axiomatisation of the parts of FM necessary to build abstraction sets and reason about them.

Proof theory developed by Gabbay and Cheney, and semantics developed by Gabbay.

At around the same time Urban wrote down the syntax of nominal terms.

Nominal terms

$$t ::= a \quad | \quad \pi \cdot X \quad | \quad \mathsf{f}t \quad | \quad (t,t) \quad | \quad [a]t.$$

$\mathsf{f}t$ is a term-former.

X is an ‘unknown element’.

$\pi \cdot X$ has a moderating permutation; π is a (finite) permutation on atoms.

Freshness

To express the capture-avoiding aspects of syntax with variable names (and its abstract nominal version) we introduce an intentional notion of freshness:

$$\frac{a\#t}{a\#\mathsf{f}t} \qquad \frac{a\#t\ a\#t'}{a\#(t,t')} \qquad \frac{a\#t}{a\#[b]t} \qquad \frac{}{a\#b} \qquad \frac{}{a\#[a]t} \qquad \frac{\pi^{-1}(a)\#X}{a\#\pi\cdot X}$$

Then the core equality of nominal terms, can be written as

$$a\#X,\ b\#X \vdash (a\ b)\cdot X = X.$$

This simple equality expresses α-equivalence.

The key point of nominal terms

The key point is the two-level structure of atoms and unknowns; instantiation of unknowns does not avoid capture.

For example, we can set X to be $[a]a$ in the core equality, and we obtain

$$[b]b = [a]a$$

which is what you'd expect ($a\#[a]a$ and $b\#[b]b$).

Nominal terms$\neq$ nominal logic

In the presence of an axiom $a = b$ it is not the case that $a\#a$ in nominal terms, but it is the case that $a\#a$ in nominal logic.

Freshness is syntactic in nominal terms, not semantic. This gives good computational properties, see e.g. Nominal Unification (Urban Pitts Gabbay), α-prolog (Cheney Urban), Nominal Rewriting (Fernández Gabbay).

Advantages

What these all have in common is the Fraenkel-Mostowski semantics of abstraction.

We get equivariance.

$[\mathbb{A}]X$ has the same cardinality as X. Note that Y^X does not have the same cardinality as Y or X (in general).

This leads to good computational properties. For example, unification of nominal terms is decidable (higher-order unification is not).

Advantages

Yet we lose no expressivity. E.g. rewriting of nominal terms is just as expressive as higher-order rewriting, because we can express β-reduction as

$$(\lambda[a]Y)X \to Y[a{\mapsto}X].$$

Here we assume term-formers λ, app and sub with sugar

$$\mathsf{app}(t,u) = tu \qquad \mathsf{sub}([a]u,t) = u[a{\mapsto}t].$$

Conclusions

It seems that the Fraenkel-Mostowsi set model is just what we need to do programming on syntax-with-binding. It's just a good fit.