# Implementing Observational Type theory in Epigram 2.

Wouter Swierstra
(with a lot of help
from my friends)

- slippery tar-pit

two notions of equality

$\equiv$ for typechecking

$=$ for reasoning, traditionally

(similarly for set)

$$\frac{\Gamma \vdash A\ \mathsf{set} \quad \Gamma \vdash a : A \quad \Gamma \vdash a' : A}{\Gamma \vdash a =_A a'\ \mathsf{set}}$$

$$\frac{\Gamma \vdash a : A}{\Gamma \vdash \overline{a}^A : a =_A a}$$

and now the schism

extensional TT $$\frac{\Gamma \vdash q : a =_A a'}{\Gamma \vdash a \equiv a' : A}$$

intensional TT $$\frac{\begin{array}{l}\Gamma; x : A; q : a =_A x \vdash P[x; q]\ \mathsf{set} \\ \Gamma \vdash a' : A \quad \Gamma \vdash p : P[a; \overline{a}^A] \\ \Gamma \vdash r : a =_A a'\end{array}}{\Gamma \vdash r\,(\mathsf{subst}_a^{a'}\ x\ q.\, P[x; q] \mid p) : P[a'; r]}$$

with $\overline{a}^A\,(\mathsf{subst}_a^a\ x\ q.\, P[x; q] \mid p) \mapsto p$

extensional TT
is jolly useful:

$(\Pi x : S.\ f\, x = g\, x)$
$\rightarrow f = g$

but undecidable
(checking $\equiv$ involves
guessing $=$)

In ETT terms are
no longer evidence.

Epigram is a language of

evidence

- No termination checker, but explicit structural recursion (".. rec").

- No coverage checker, but explicit pattern matching (".. case")

- Adding the axiom of extensionality is not in the spirit of the language.

Things get even worse....

Extensional type theory has a clear underlying intuition:

What can you do with functions, but apply them?

But what about:

$$\underline{\text{data}} \quad \frac{f : \mathbb{N} \to \mathbb{N}}{\mathsf{isId}\ f : \star}$$

$$\underline{\text{where}} \quad \frac{}{\mathsf{idIsId} : \mathsf{isId}\ (\lambda n \to n)}$$

Use functions to index data types.

# So what?

Clearly:

idIsId : isId $(\lambda\, n \to n)$

But we can prove:

$$(\lambda\, n \to n) = (\lambda\, n \to n+0)$$

And use substitution to make an inhabitant of

isId $(\lambda\, n \to n+0)$

in the empty context.

But what constructor made this term?

# Story so far:

We have to be very, very, very careful if we want some form of extensional equality for Epigram:

- we want more evidence
- extensionality and indexed data families are tricky, just ask Peter!

# What is observational type theory?

- Thorsten Altenkirch, LICS '99 Extension type theory in intensional type theory. LICS '99
- Thorsten Altenkirch and Conor McBride, Towards observational type theory, 2006.

For one thing, it's not my idea.

Remember isId?

Substitution by non-refl proofs is hairy.
(start with refl, add ext.)

Observational type theory is backwards:

1) what is observable?
(ext.)

2) What about refl?

# Hello John Major!

Start from heterogeneous equality:

$$\frac{S_0 \text{ set} \qquad S_1 \text{ set}}{S_0 = S_1 \text{ set}}$$

$$\frac{s_0 : S_0 \qquad s_1 : S_1}{(s_0 : S_0) = (s_1 : S_1) \text{ set}}$$

so far so good.

Now proceed
structurally over types
thinking about
observable behaviour.

$$\frac{P : (\Pi x : S_0 . T_0[x]) = (\Pi x : S_1 . T_1[x])}{\text{project } P : S_0 = S_1}$$

Similarly.

$$\frac{P : \Pi x{:}S_0.\,T_0[x] = \Pi x{:}S_1.\,T_1[x] \; ; \quad p : (s_0 : S_0 = s_1 : S_1)}{\text{apply } P\ p : T_0[s_0] = T_1[s_1]}$$

And similarly for $\Sigma$.

But what about

extensionality?

The conversion rule converts definitionally equal terms.

In OTT we build an explicit coercion between provably equal terms.

$$\frac{P : S_0 = S_1 ; \quad s_0 : S_0}{\mathrm{coerce}\ P\ s_0 : S_1}$$

Even better:
coercion does not change terms.
We can show that:

$$\frac{P : S_0 = S_1 \ ; \ s_0 : S_0}{\text{coherence } P\ s_0 : (s_0 : S_0) = (\text{coerce } P\ s_0)}$$

Sceptic:

"I don't trust you!"

Me:

"and I don't trust myself!"

1000 lines of Agda on a single slide:

- construct a universe $U$ closed under $\Pi$, $\Sigma$, $W$ with $\emptyset$, $1$, $2$.
- define equality on $U$.
- define equality on $el(U)$.
- prove that these equalities are symmetric and transitive.
- define

$$\mathrm{coerce} : (u_1 : U) \to (u_2 : U) \to (u_1 = u_2) \to el\ u_1 \to el\ u_2$$

- prove

$$\mathrm{coherence} : (u_1 : U) \to (u_2 : U) \to (P : u_1 = u_2) \to (x : el\ u_1) \to x = \mathrm{coerce}\ u_1\ u_2\ P\ x$$

Sceptic:
But you have not proven refl and you use it!

Me:
The meta-theory of Agda says refl is OK - see the paper.

Besides, if I could prove refl, OTT would be little more than a construction in Agda!

# Proof irrelevance

Coercion never inspects the proof:

- kills off the off-diagonal cases
- projects out parts for recursive calls.

That's all (Agda keeps me honest).

# What can we do with Observational type theory?

- derive extensionality!

- prove the induction principle for Nat - and add even more evidence. No more awkward schemas!

- opens the door for type theoretic treatment of quotients, coinduction, ....