Bob Harper (Lecture 3) [2018/07/18] (1)

REVIEW

So far we've been developing
<u>computational</u> <u>type</u> <u>theory</u>
Types are behavioral specifications

2 Principle Forms of Judgment:

(exact) type equality $A \doteq A'$ (A type means $A \doteq A$)

(exact) member equality $M \doteq M' \in A$ ($M \in A$ means $M \doteq M \in A$)

possible notation: $M \doteq_A M'$

-> expressions do not intrinsically "have" a type!
the same program can satisfy <u>many</u> specs!

e.g. $M \in Nat \to Nat$ $\quad M \in Primes \to Primes$ ...

e.g. $\lambda x. x \doteq \lambda x. |x| \in Nat \to Nat$

$\lambda x. x \not\doteq \lambda x. |x| \in Int \to Int$

Definitions of Types

Bool, Nat } inductive types

1 aka Unit (exercise) $\langle\rangle$ val

$A_1 \times A_2$ product, $A_1 \to A_2$ functions } in terms of A_1, A_2

0 aka Void (exercise) $case\{\}(M)$

$A_1 + A_2$ (exercise) $1 \cdot M,\ 2 \cdot M,\ case\{a_1 M_1\ a_2 M_2\}(M)$

$a : A_1 \times A_2(a)$ dep. product (Σ-type)

$a : A_1 \to A_2(a)$ dep fn (Π-type)

horizontal induction

Bool | true, false
Nat | 0, succ(0), succ(succ(0)), ...
Bool × Nat |
Bool -> Nat |
Nat -> Nat |
⋮

struct of types

To define $a : A_1 \times A_2$
must already have
def'd A_1. $A_2[M_1/a]$ for
every $M_1 \in A_1$

A_1 type $\quad a : A_1 \gg A_2$ type

A_1

$A_2[M_1]$ $A_2[M_1']$ ...

"It's all about how programs are <u>run</u>."

"Interesting types don't always correspond in a natural way to things in logic."

Type theory - a theory of computation (program specification)

Brouwer as a means to give a notion of truth for logical propositions.
"Propositions-as-types" aka semantic correspondence.

Capsule Summary

How to explain this? { ϕ Prop ← (specification or problem)
ϕ True ← (solution or satisfier) }

specification: ϕ^* type
solution: ϕ^* is inhabited i.e. $\exists M \in \phi^*$ } type is primary and more extensive than "mere" logic

Typically we have the following
$\top^*$ the trivially true type 1 aka unit

Equality of types $\phi^* \doteq \psi^*$ (?)
when are two specifications of a problem the same
(or at least spec. the same problem)

$\top^*$ — 1 (aka unit)
$\bot^*$ — 0 (aka void)
$(\phi_1 \wedge \phi_2)^*$ — $\phi_1^* \times \phi_2^*$
$(\phi_1 \vee \phi_2)^*$ — $\phi_1^* + \phi_2^*$ (N.B. can be problematic / controvertial)
$(\phi_1 \supset \phi_2)^*$ — $\phi_1^* \longrightarrow \phi_2^*$
$(\forall a : A . \phi_a)^*$ — $a : A \longrightarrow \phi_a^*$
$(\exists a : A . \phi_a)^*$ — $a : A \times \phi_a^*$ (NB "constructive existence")

The Fact of Choice

in type thy { $[(\forall a : A . \exists b : B . R(a,b))$ "R is total"
$\supset \exists f : A \to B . \forall a : A . R(a, f(a))$ true$]$ ("choice fn") }

in logic { let $F : (a : A \longrightarrow (b : B \times R(a,b)))$
$\longrightarrow (f : A \to B \times (a : A \longrightarrow R(a, f(a))))$
then $f \triangleq \lambda a . (F(a) \cdot 1)$ }

Bob Harper [2018/07/18] 3

what should we do about $(M =_A M')^*$?

In the semantic setting ...

How do we interpret the proposition of equality as a type.

$\boxed{\text{Equality Types}}$

$Eq_A(M_1, M_2) \doteq Eq_{A'}(M_1', M_2')$ iff $A \doteq A'$, and $M_1 \doteq M_1' \in A$ $M_2 \doteq M_2' \in A$

$M \in Eq_A(M_1, M_2)$ iff $M \Downarrow *$ and $M_1 \doteq M_2 \in A$.

"subsingleton" $Eq_{Nat}(2, 3)$ is uninhabited.

$Eq_{Nat}(2, 1+1)$ inhabited.

$M \in Eq_A(M_1, M_2)$ iff $M_1 \doteq M_2 \in A$.

$(M_1 =_A M_2)^* = Eq_A(M_1, M_2)$ "works"

EXERCISE

Show that $Eq_A(\_,\_)$ is the least reflexive relation on A such that

1) $* \in Eq_A(M, M)$ whenever $M \in A$

2) To show $Eq_A(M, N) \to R(M, N)$ it suff. to show R is reflexive $a : A \gg \_ \in R(a, a)$.

"equality induction"

Formalism formal type theory is inductively defined by a collection of rules for deriving:

$\Gamma \vdash A \text{ type}$ $\quad\Gamma \vdash A \equiv A'$ "definitional equality" (up for grabs)

$\Gamma \vdash M : A$ $\quad\Gamma \vdash M \equiv M' : A$

Typical axioms

$$\frac{\Gamma \vdash M : A \quad \Gamma \vdash A \equiv A'}{\Gamma \vdash M : A'} \qquad \frac{}{\Gamma, x : A, \Gamma' \vdash x : A}$$

$$\frac{\Gamma \vdash M : A_1 \times A_2}{\Gamma \vdash M \cdot i : A_i}\,(i=1,2) \qquad \frac{\Gamma \vdash A_1 \text{ type} \quad \Gamma \vdash A_2 \text{ type}}{\Gamma \vdash A_1 \times A_2 \text{ type}}$$

$$\frac{\Gamma \vdash M_1 : A_1 \quad \Gamma \vdash M_2 : A_2}{\Gamma \vdash \langle M_1, M_2\rangle \cdot i \equiv M_i : A_i} \quad ✓ \quad \frac{\Gamma \vdash M_1 : A_1 \quad \Gamma \vdash M_2 : A_2}{\Gamma \vdash \langle M_1, M_2\rangle : A_1 \times A_2}$$

$$\frac{\Gamma \vdash M : A_1 \times A_2}{\Gamma \vdash \langle M\cdot 1, M\cdot 2\rangle = M : A_1 \times A_2} \quad ?$$

Derivation

composition of rules

$\Gamma \vdash A \equiv A'$

} inherently computably enumerable

Design Requirements

- All judgments should be decidable; e.g.
  - type checking
  - definitional equivalence (for some choice of definitional equality)
- structural properties of entailment

$\longrightarrow$ critical point of view: the formalism is a useful approximation to the truth.

Via formal corresp. w/ formal logic
 type-checking proof-checking (derivation-checking)

The formal corresp. is boring (obvious)
The semantic corresp. is interesting (nonobvious).

How to formalize equality ?

1st cut: "equality reflection" (ER)

$$\frac{\Gamma \vdash M : Eq_A(M_1, M_2)}{\Gamma \vdash M_1 \equiv M_2 : A}\ (ER) \qquad \boxed{???}$$

$$\frac{\Gamma \vdash M_1 \equiv M_2 : A_2}{\Gamma \vdash * : Eq_A(M_1, M_2)}$$

$refl_A(M_1)$

How can we check mechanically that $M_1 \equiv M_2$?
Proof search: find an M that inhabits the type $Eq_A(M_1, M_2)$???