Design and Implementation of Andromeda

Andrej Bauer Gaëtan Gilbert
Philipp Haselwarter Matija Pretnar
Christopher A. Stone

TYPES 2016

Type theory with equality reflection

- dependent product
- equality type

$$\textsc{eq-reflection}\quad \frac{\Gamma \vdash\ p\ :\ \mathsf{Eq}_T(e_1, e_2)}{\Gamma \vdash e_1 \equiv e_2 : T}$$

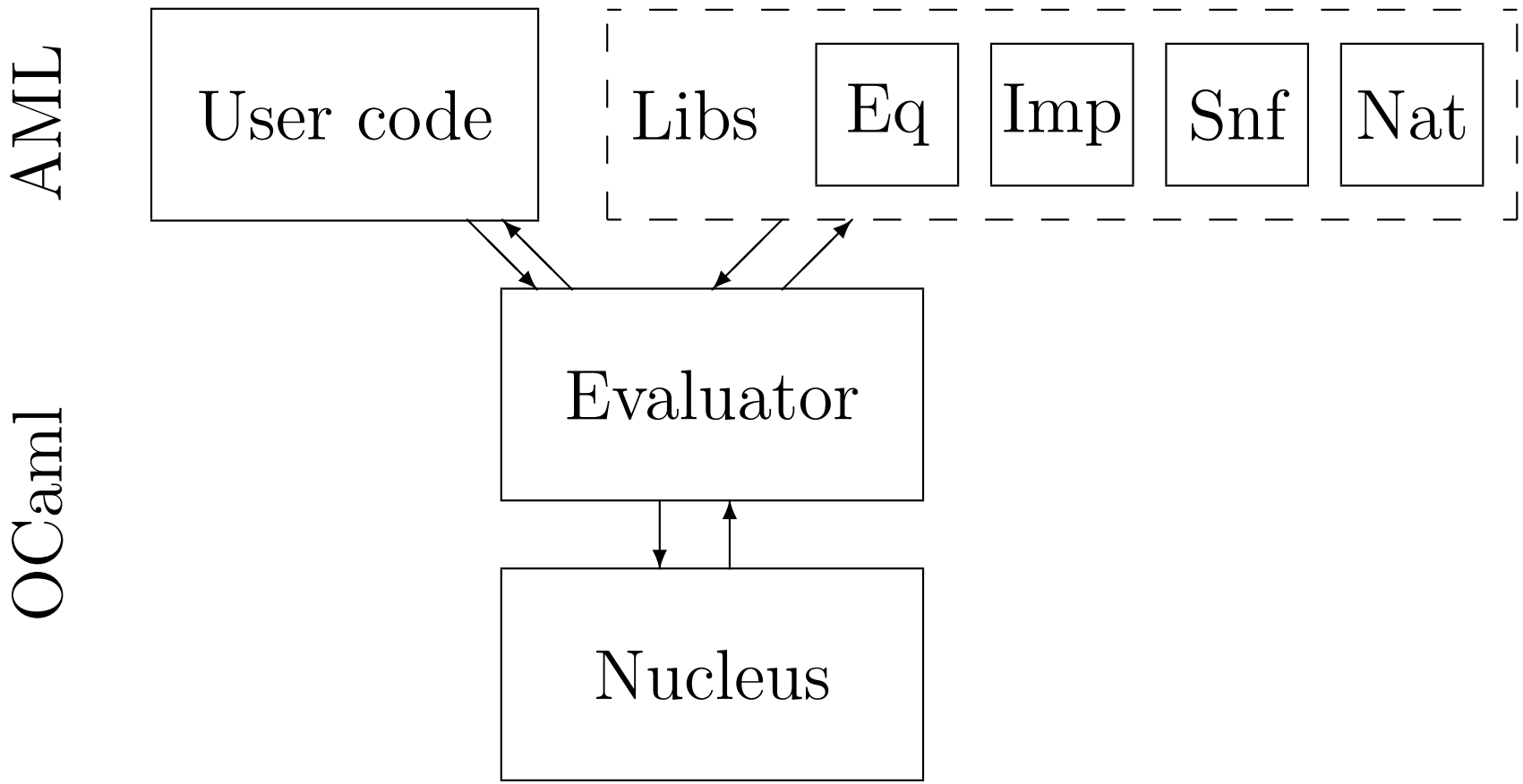
Expressivity

This TT can hypothesise judgemental equalities.

In Andromeda, this is used for

- definitions
- rewriting rules
- extensionality principles
- new types with *judgemental* computation rules:
  - declare constants for types and constructors
  - declare constants of equality types

Architecture



Nucleus

- simple: 1800 lines of pure OCaml
- implements exactly the rules of our TT
- computes *judgements* $\Gamma \vdash e : T$

Andromeda Meta Language (AML)

- ML-style language
- special data type `judgement` :
  - smart constructors via Nucleus
  - pattern matching on syntax
- algebraic effects & handlers à la Eff

Questions asked by the Evaluator

- equality:
  $$\Gamma \vdash (\lambda x{:}T\,.\,x)\,e \equiv e : T$$
- type shape analysis:
  $$T \hookrightarrow \prod_{(x:A)} B$$
- coercions:
  "a has type Field but expected a Ring"

Soundness

The values of type **`judgement`** *are the derivable judgements in type theory with equality reflection.*

We implemented some safe algorithms:

- `equal.m31` (600 lines)
- `hints.m31` (400 lines)

Related work

- rewriting in TT: ‘sprinkles’, Dedukti, CoqMT, ...
- “extensional” type theory: Nuprl
- ML based ITPs: LCF / HOL (light) family

Plans

- improve efficiency
- replace Type : Type with universes
- axiomatise Homotopy Type System
- users!