Towards Human-Centered Types & Type Debugging

Kasra Ferdowsi
(He/They)
PhD Student

hci.social/@kasra

@kasra_ferdowsi
Overview

A Critique of Foundations for Human-Centered Types*:

1. Expressiveness v. Usability
2. Tool- v. Human-Centric Types

A Concrete Example:

1. A Problem with Rust
2. A Solution?

* Technically, any static semantics.
Types are mainstream!
Expressiveness v. Usability

Consider Ownership Types*:

- From Object-Oriented Programming, often for Java.
- Evaluated on...
  - Expressiveness
  - Syntactic Overhead

* [Clarke et al. 1998, Aldrich et al. 2002, Boyapati et al. 2003]
Expressiveness v. Usability*

Expressiveness:

Usability:

* [Hage 2020]
Usability in Type Error Messages*

We've addressed Usability in Type Error Messages:

Experience [...] shows that from a type error message it is often hard to deduce the actual cause of the error and understand it [1, 2, 3, 4, 9, 10, 17, 19, 20, 21, 22, 23, 24, 25, 26].

[Chitil 2001]

But they only focus on error localization and correction...

And only with Tool-Centric solutions.

* [Clack and Myers 1995, Lerner et al. 2007, Chen and Erwig 2014, Zhang and Myers 2014, Tirronen et al. 2015, Zhang et al. 2017]
Tool- v. Human-Centric Debugging

Consider Guided Type Debugging*.

- Elicit feedback from the user:
  
  What is the expected type of rR?

- Employ the User to help the tool understand.

* [Chen and Erwig 2014]
Tool- v. Human-Centric Debugging

*Tool-Centric:*

*Human Squid-Centric:*
Overview

A Critique of Foundations for Human-Centered Types¹:

1. Expressiveness v. Usability
2. Tool- v. Human-Centric Types

A Concrete Example:

1. A Problem with Rust
2. A Solution?

¹ Technically, any static semantics.
A Problem with Rust

Rust's Ownership:

- At most *single mutable* access to each value in the program.
- Statically guaranteed by its type system.

Rust's usability:

"Learning Rust Ownership is like navigating a maze where the walls are made of asbestos and frustration, and the maze has no exit, and every time you hit a dead end you get an aneurysm and die"

Student Participant from [Coblenz et al. 2022]
A Problem with Rust

Error localization and correction

Feedback that "aid design and comprehension"
A Solution?

Scalad [Plociniczak 2016]

Compositional Exploration of Types [Chitil 2001]

Expression: (:) (last xs)
Type : [a]→[a]
with xs [[a]]
  because
Expressions: (:) last xs
Types: a→[a]→[a] b
with xs [[b]]
A Solution?

Expression: `(:) (last xs)`
Type : `[a]→[a]`
with xs : `[a]`
Isn't this your expression?  
Yup.

And this is your type annotation?  
Yup.

They're irreconcilable. And if that's the case, one must be incorrect.

That makes sense to me.
A Solution?

Larger architectural implications?
Summary
Also Happy Valentine's Day!!!
Feedback / Questions / Favorite Love Song?