

About my background

TODO



Architecture of compilers

TODO

ExtSTG and STG interpreter

- Architecture of GHC Haskell/Core/STG/Cmm/RTS
- ► How to use STG to generate executable via STG
- ▶ What is ExtSTG and GHC-WPC

Architecture of the Idris2 Compiler

- ► Surface language =? Dependent types
- ► Elaboration =? Type Inference
- Code generation
- Backends

Lazy Haskell and Strict Idris?

- ▶ In STG there is no overhead for lazy computation.
- Case expressions force evaluation.

https://www.reddit.com/r/haskell/comments/ku1zsm/nextgen_haske

A bit on the backend part

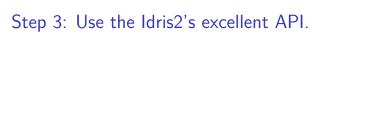
- ► Lifted
- ► ANF
- ► VM

But what is STG?

- Lambda calculus like language.
- ► TODO: Overview image

Step 2: Select the Idris IR

► Why did I select ANF?



Compiler Backend API

Idris2 isn't an optimiser compiler, but

- Optimisation opportunities
- Your backend is responsible for optimisations

Core Monad

- ► What's in the core monad?
- ► How it represents definitions
- Names and Resolved names
- Compiled vs Expr
- How it represents datatypes



Challenges

- ► How to represent Values
- How to represent FFI
- Implement String operations
- Implement arbitrary precession integers
- How to compile TopLevel definitions
- How to get the right arity
- ► Holes

Holes

Holes in idris programs are compiled to runtime crashed. The backend should at least generate some warnings when runtime crash toplevel is found during compilation.

FFI

TODO: What is the difference between %extern and %foreign?

Step 5: Enjoy your Idris program

Step 6: Compare with others

Other backends

- ► Scheme
- ► RefC
- Lua



NOTES:

- ▶ Use screenshots in the presentation when showing IRs.
- ▶ Use color scheme of the Idris ReadTheDocs