Formal Verification of System FC Using the Coq Proof Assistant

Tiernan Garsys, Tayler Mandel, Lucas Peña, & Noam Zilberstein
Advised by Stephanie Weirich & Richard Eisenberg

Project Outline

> Formally verify type safety of System FC using the Coq Proof Assistant

> Type safety guarantees the absence of security holes like buffer overruns

> Verification of System FC has a direct correspondence to verification of Haskell

Haskell

> High-level, statically-typed, functional language

> Used because of “type safety” guarantees

> System FC, the underlying language of Haskell, is not proven to be type safe

GHC Core & System FC

> GHC Core is the implementation of System FC within GHC

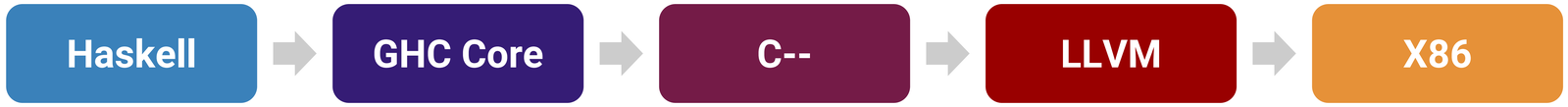
> Haskell code is syntactic sugar for GHC Core programs

The Coq Proof Assistant

> Interactive, automated theorem prover

> Proofs are mechanically verified by the Coq interpreter

Glasgow Haskell Compiler (GHC) Stages



Haskell

```
id :: a -> a
id x = x

y :: Int
y = id 1
```

GHC Core

```
id :: forall a_apE. a_apE -> a_apE
id = \ (@ a_aHK) (x_apF :: a_aHK) -> x_apF

y :: GHC.Types.Int
y = id @ GHC.Types.Int (GHC.Types.I# 1)
```

System FC

$$\texttt{id} := \Lambda T.\ \lambda x : T.\ x \qquad \varnothing \vdash \texttt{id} : \forall T,\ T \to T$$
$$\texttt{y} := \texttt{id}\ [\texttt{Int}]\ 1 \qquad \varnothing \vdash \texttt{y} : \texttt{Int}$$

$$\texttt{y} = (\Lambda T.\lambda x : T.x)\ [\texttt{Int}]\ 1 \Rightarrow (\lambda x : \texttt{Int})\ 1 \Rightarrow 1$$

Progress

```
Theorem progress : forall t T,
     empty |- t \in T ->
     value t \/ exists t', t ==> t'.
```

Preservation

```
Theorem preservation : forall Gamma t t' T,
     Gamma |- t \in T  ->
     t ==> t'          ->
     Gamma |- t' \in T.
```

Type Safety

> **Progress:** any well-typed expression is either a value, or can continue to be evaluated

> **Preservation:** the type of an expression is preserved after evaluation

> **Soundness:** any well-typed term evaluates to a value

Future Work

> Verify translation to GHC Core

> Use formalization to verify compiler optimizations and extensions to GHC Core