Proof Compilation

Christopher Lam

Joint work with Audrey Seo (UW), Talia Ringer (UIUC), and Dan Grossman (UW)

Motivation

Motivation

- gcc is 15 million lines of code!

Motivation

- gcc is 15 million lines of code!
    - We have no reason to even believe this is correct!

Motivation

- gcc is 15 million lines of code!
  - We have no reason to even believe this is correct!
    - We have explicit reason to believe it isn’t correct!

Motivation

- gcc is 15 million lines of code!
  - We have no reason to even believe this is correct!
    - We have explicit reason to believe it isn’t correct!
- Full correctness is too powerful!

Proposed Workflow

Proposed Workflow

1. Prove a property P about a program c in your source language using Hoare logic

Proposed Workflow

1. Prove a property P about a program c in your source language using Hoare logic
2. Compile c

Proposed Workflow

1. Prove a property P about a program c in your source language using Hoare logic
2. Compile c
    A. While c is being compiled, look at the transformations being applied to the program

Proposed Workflow

1. Prove a property P about a program c in your source language using Hoare logic
2. Compile c
   A. While c is being compiled, look at the transformations being applied to the program
   B. Transform the Hoare logic proof of property P along with c

Proposed Workflow

1. Prove a property P about a program c in your source language using Hoare logic
2. Compile c
    A. While c is being compiled, look at the transformations being applied to the program
    B. Transform the Hoare logic proof of property P along with c
3. Retrieve an analogous property P' about the compiled program c'

Two problems!

Two problems!

- Translating Hoare proofs across levels of abstraction

Two problems!

- Translating Hoare proofs across levels of abstraction
- Mangling Hoare proofs with compiler optimizations

Two problems!

- Translating Hoare proofs across levels of abstraction
- Mangling Hoare proofs with compiler optimizations

We've done this one

A Quick Example

A Quick Example

$$\{5 \leq 10\}\,\texttt{x := 5}\,\{x \leq 10\}\,\texttt{z := 99}\,\{x \leq 10\}$$

A Quick Example

$$\begin{tikzcd}
\{5 \leq 10\}\ x := 5\ \{x \leq 10\}\ z := 99\ \{x \leq 10\} \arrow[d] \\
\{5 \leq 10\}\ \#1 := 5\ \{\#1 \leq 10\}\ \#2 := 99\ \{\#1 \leq 10\}
\end{tikzcd}$$

A Quick Example

$$\begin{tikzcd}
\{5 \leq 10\}\ x := 5\ \{x \leq 10\}\ z := 99\ \{x \leq 10\} \arrow[d] \\
\{5 \leq 10\}\ \#1 := 5\ \{\#1 \leq 10\}\ \#2 := \#1 + \#2\ \{\#1 \leq 10\}
\end{tikzcd}$$

A Quick Example

$$\begin{tikzcd}
\{5 \leq 10\}\, x := 5\, \{x \leq 10\}\, z := 99\, \{x \leq 10\} \arrow[d] \\
\{5 \leq 10\}\, \#1 := 5\, \{\#1 \leq 10\}\, \#2 := 42042\, \{\#1 \leq 10\}
\end{tikzcd}$$

A Quick Example

$$\begin{tikzcd}
\{5 \leq 10\}\ x := 5\ \{x \leq 10\}\ z := 99\ \{x \leq 10\} \arrow[d] \\
\{5 \leq 10\}\ \#1 := 5\ \{\#1 \leq 10\}\ \#2 := \mathrm{TOM}\ \{\#1 \leq 10\}
\end{tikzcd}$$

Directions?

Directions?

- Implement compiler optimizations

Directions?

- Implement compiler optimizations
  - Probably need to prove something about Heyting Kleene Algebras with Tests

Directions?

- Implement compiler optimizations
  - Probably need to prove something about Heyting Kleene Algebras with Tests
- Beyond toy models

Directions?

- Implement compiler optimizations
  - Probably need to prove something about Heyting Kleene Algebras with Tests
- Beyond toy models
  - Lustre