Calculating Correct Compilers

Cameron Wong

CS-252R

2022-11-03

Calculating Correct Compilers

Calculating Correct Compilers

Compilers?

- Let S and T be the source and target languages
- We define a compiler to be a function

$$\texttt{compile} : \textsc{Syntax}_S \to \textsc{Syntax}_T \tag{1}$$

Compilers?

- For our purposes, a compiler is a function from syntax to syntax
- Means we ignore practical concerns like linking, parsing, etc

Compilers?

- Also, assume that the input is well-formed
  - No type errors, etc

Compilers?

- As a last matter of terminology, the language that `compile` is written in is called the host language.

Calculating Correct Compilers

Correct?

- Languages are defined by their syntax and semantics
- If `compile` acts on syntax, its *correctness* should talk about semantics

Correct?

- How do we talk about a relationship between semantics?

Correct?

- Intuitively, e and `compile` e should "mean the same thing"
- If we only worry about runtime, then e and `compile` e should "do the same thing"

Correct?

- This is actually very difficult to state formally!

Correct?

- What parts of behavior need to be preserved?
  - Performance?
  - Memory?

Correct?

- Even just looking at "return values" takes some machinery
- Suppose
  - $e \rightsquigarrow^* v$ and
  - `compile` $e \rightsquigarrow^* \overline{v}$
- v is in S, but $\overline{v}$ is in T!
- For example, if S is Java and T is assembly, v could be some a complex object!

Correct?

- One way is to define a relation R on values(S) $\times$ values(T)
- Can then define correctness as
  - If $e \rightsquigarrow^* v$ and `compile` $e \rightsquigarrow^* \overline{v}$, then $R(v, \overline{v})$
- Could generalize and define the relation over $\textsc{Syntax}_S \times \textsc{Syntax}_T$, but this is typically much harder

Correct?

- Another way: define some common semantic domain D, with denotation functions
  - $[\![\cdot]\!]_S$: expressions$(S) \to D$
  - $[\![\cdot]\!]_T$: expressions$(T) \to D$
- Then correctness is stated as

$$[\![e]\!]_S = [\![\texttt{compile } e]\!]_T \tag{2}$$

[0] This is often called

Correct?

- For us: Our approach will look like the denotation method, embedded into the host language
  - That is, $[\![\cdot]\!]_S$ and $[\![\cdot]\!]_T$ will be host language functions, rather than purely metatheoretical

Calculating Correct Compilers

Setup