Concepts of programming languages

Gaurish Sharma

September 18, 2023

Contents

Lecture 2

Mon 18 Sep 2023 12:08

1 Formalization

Definition 1. Formalization is the process of giving a language a formal, mathematical description.

Typically described in another language (meta-language.) For implementation it may be a programming language. Eg. Haskell

Definition 2. Object language: the language we're describing.

Definition 3. Meta-language - The language we're using to describe the object language.

Things to formalize in programming languages:

- Static semantics - scoping, typing.
- Dynamic Semantics - cost models, runtime behavior.
- Syntax - parsing, grammar.

There is a correspondence with formlising logics.

What logicians needed to formalize logic.

- Well formedness - scoping, typing
- Logical Models - proof models, truth models.
- Syntax - Ambiguity, grammar etc.

2 Judgements

Definition 4. A judgement is a statement asserting a certain property for an object.

Note: not necessarily a true assertion.

Notation. judgement for property A holds for object s by writing sA.

2.1 Proving Judgements

we prove judgements by inference rules. Inference rules are crafting recipes for judgements.

Notation. Inference rule is written as:

$$\frac{J_1\;J_2\;\dots\;J_n}{J}.$$

In order to prove J (conclusion), we must prove J_i (premises) where $i \in \{1, 2, \dots n\}$. Rules with no premises are called axioms.

Continually decompose your conclusion by applying inference rules. Until you reach a base case inference rule.

When we define judgement by inference rules. we keep in mind that what is the smallest set that is closed under inference rules or equivalently, everything we can derive from the rules using finite proof trees. This is also what it means for a definition to be inductive.

Definition 5. Backward-proof: To prove judgement sA holds.

- Find a rule whose conclusion matches sA
- Preconditions of applied rule become new proof obligations.
- Repeat until all obligations are proven until axioms.

2.2 Derivability

Definition 6. A rule

$$\frac{J_1\ J_2\ \dots J_n}{J}.$$

is derivable if, using the original (defining) set of rules, we can construct a proof of J.

Definition 7. A rule is admissible if, by adding it, we don't change the meaning of the judgement i.e does not add the set of statements which satisfy the judgement.

Definition 8. by adding more rules, you make at least as many judgements derivable as before.

Notation. $A \vdash B$ is equivalent to $\frac{A}{B}$.

2.3 Hypothetical Derivation

$$\frac{A \vdash B}{C}.$$

Read as: If assuming A we can derive B, then we can derive C.

Definition 9. The smallest set of rules to define every string in a language is it's minimal definition.

It is also an inductive definition of the language.

3 Rule Induction

Informal process:

To show all strings are valid in M

- prove $P(base)$
- prove premise satisfies $P \Rightarrow$ post condition satisfies P.

Definition 10. Rule induction: Given a set of rules R, we may prove a property P inductively for all judgements that can be inferred with R by showing, for each rule of the form:

$$\frac{J_1\ J_2\ \ldots J_n}{J}.$$

that if P holds for premise then P holds for conclusion

Note: axioms are the base cases of the induction, all other rules form inductive cases, and the premise for each rule give rise to inductive hypothesis.
Structural induction is a special case of rule induction.

Tutorial 3: Tut2

Mon 18 Sep 2023 15:08

1. State the formal properties of the left identity, right identity and associativity for ++.
Sol.

- $[\,] + +ys = ys$
- $ys + +[\,] = ys$
- $(xs + +ys) + +zs = xs + +(ys + +zs)$

Proof.

- $[\,] + +ys = ys$ (given)
- To prove: $ys + +[\,] = ys$
  LHS $= [\,] ++ [\,] = [\,]$ therefore true for base case.
  Assume $xs + +[\,] = xs$ Prove true for inductive case

□