Content

- Intro & motivation, getting started [1]
- Foundations & Principles
  - Lambda Calculus, natural deduction [2, 3, 4]
  - Higher Order Logic [5, 6, 7]
  - Term rewriting [8, 9, 10]
- Proof & Specification Techniques
  - Isar [11, 12]
  - Inductively defined sets, rule induction [13, 15]
  - Datatypes, recursion, induction [16, 17, 18, 19]
  - Calculational reasoning, mathematics style proofs [20]
  - Hoare logic, proofs about programs [21, 22, 23]

Slide 1

Slide 2

Slide 3

HOW INDUCTIVE DEFINITIONS WORK

Last Time

- Sets
- Type Definitions
- Inductive Definitions
The Nat Example

\[ 0 \in \mathbb{N}, \quad n \in \mathbb{N} \implies n + 1 \in \mathbb{N} \]

\[ \mathbb{N} \text{ is the set of natural numbers} \]

But why not the set of real numbers?

\[ 0 \in \mathbb{R}, \quad n \in \mathbb{R} \implies n + 1 \in \mathbb{R} \]

\[ \mathbb{N} \text{ is the } \textit{smallest} \text{ set that is consistent with the rules.} \]

Why the smallest set?

\[ \text{Objective: no junk. Only what must be in } X \text{ shall be in } X. \]

\[ \text{Gives rise to a nice proof principle (rule induction)} \]

---

The Set

**Definition:** \( B \) is \( R \)-closed if \( R B \subseteq B \)

**Definition:** \( X \) is the least \( R \)-closed subset of \( A \)

This does always exist:

**Fact:** \( X = \bigcap \{ B \subseteq A. B \text{ \( R \)-closed}\} \)

---

Formally

**Rules**

\[ a_1 \in X \quad \ldots \quad a_n \in X \]

define set \( X \subseteq A \)

**Formally:** set of rules \( R \subseteq A \times A \) (\( R, X \) possibly infinite)

**Applying rules** \( R \) to a set \( B \):

\[ R B = \{ x. \exists H. (H, x) \in R \land B \subseteq \mathcal{R} \} \]

**Example:**

\[ R = \{ (\{ \}, 0) \cup (\{ n \}, n + 1). n \in \mathbb{R} \} \]

\[ R \{ 3, 6, 10 \} = \{ 0, 4, 7, 11 \} \]

---

Generation from Above
Rule Induction

\[ \forall n \in \mathbb{N}, \quad n \in \mathbb{N} : n + 1 \in \mathbb{N} \]

induces induction principle

\[ \{ P 0; \bigwedge n. P n \bigwedge P (n + 1) \} = \forall x \in X. \ P x \]

In general:

\[ \forall(\{ a_1, \ldots, a_n \}, a) \in R. \ P a_1 \land \ldots \land P a_n \Rightarrow P a \]

\[ \forall x \in X. \ P x \]

Why does this work?

\[ \forall(\{ a_1, \ldots, a_n \}, a) \in R. \ P a_1 \land \ldots \land P a_n \Rightarrow P a \]

\[ \forall x \in X. \ P x \]

\[ \forall(\{ a_1, \ldots, a_n \}, a) \in R. \ P a_1 \land \ldots \land P a_n \land C_1 \land \ldots \land C_m \land \{ a_1, \ldots, a_n \} \subseteq X \Rightarrow P a \]

\[ \forall x \in X. \ P x \]

Rules with side conditions

\[ a_1 \in X \quad \ldots \quad a_n \in X \quad C_1 \quad \ldots \quad C_m \]

\[ a \in X \]

induction scheme:

\[ \forall(\{ a_1, \ldots, a_n \}, a) \in R. \ P a_1 \land \ldots \land P a_n \land C_1 \land \ldots \land C_m \land \{ a_1, \ldots, a_n \} \subseteq X \Rightarrow P a \]

\[ \Rightarrow \forall x \in X. \ P x \]

X as Fixpoint

How to compute \( X \)?

\( X = \cap \{ B \subseteq A. \ B \text{ \( R \)}} \text{ closed} \} \)

hard to work with.

Instead: view \( X \) as least fixpoint, \( X \) least set with \( \hat{R} X = X \).

Fixpoints can be approximated by iteration:

\[ X_0 = R^0 \{ \} = \{ \} \]

\[ X_1 = R^1 \{ \} = \text{rules without hypotheses} \]

\[ \vdots \]

\[ X_n = R^n \{ \} \]

\[ X_\omega = \bigcup_{x \in \mathbb{N}} (R^n \{ \}) = X \]
Knaster-Tarski Fixpoint Theorem:
Let \((A, \leq)\) be a complete lattice, and \(f : A \rightarrow A\) a monotone function. Then the fixpoints of \(f\) again form a complete lattice.

Lattice:
Finite subsets have a greatest lower bound (meet) and least upper bound (join).

Complete Lattice:
All subsets have a greatest lower bound and least upper bound.

Implications:
- least and greatest fixpoints exist (complete lattice always non-empty).
- can be reached by (possibly infinite) iteration. (Why?)

Exercise
Formalize the this lecture in Isabelle:
- Define \(\text{closed} f\) and \(\text{closed} f\) for some set operations.
- Show that \(\text{closed} f\) and \(\text{closed} f\) for some set operations.
- Define \(\text{lfp} f\) as the intersection of all \(f\) closed sets.
- Show that \(\text{lfp} f\) is a fixpoint of \(f\) if \(f\) is monotone.
- Show that \(\text{lfp} f\) is the least fixpoint of \(f\).
- Declare a constant \(R : (\alpha \times \alpha)\) set.
- Define \(\hat{R}\) as \(\alpha\) set in terms of \(R\).
- Show soundness of rule induction using \(\hat{R}\) and \(\text{lfp} \hat{R}\).

Rule Induction in ISAR
Inductive definition in Isabelle

```isar
inductive X :: α ⇒ bool
where
rule1: "[X s; A] ⇒ X s'"
... | rule_n: ...
```

Slide 17

Abbreviations

```isar
show "X x ⇒ P x"
proof (induct rule: X.induct)
case rule1
... show ?case
next
... next
  case rule_n
... show ?case
qed
```

Slide 19

Rule induction

```isar
show "X x ⇒ P x"
proof (induct rule: X.induct)
  fix s and s' assume "X s" and "A" and "P s"
... show "P s'"
next
  qed
```

Slide 18

Implicit selection of induction rule

```isar
assume A: "X x"
... show "P x"
using A proof induct
... qed
lemma assumes A: "X x" shows "P x"
using A proof induct
... qed
```

Slide 20
Renaming free variables in rule

```plaintext
case (rule, x₁ ... xₖ)
```

Renames first \( k \) variables in rule, to \( x₁ ... xₖ \).

---

A remark on style

- `case (rule, s y) ... show ?case`
  - is easy to write and maintain

- `fix x y assume formula ... show formula'`
  - is easier to read:
    - all information is shown locally
    - no contextual references (e.g. ?case)

---

**DEMO: RULE INDUCTION IN ISAR**

---

We have learned today ...

- Formal background of inductive definitions
- Definition by intersection
- Computation by iteration
- Formalisation in Isabelle
- Rule Induction in Isar