Preuves Interactives et Applications

Burkhart Wolff

http://www.lri.fr/~wolff/teach-material/2020-2021/M2-CSMR

Université Paris-Saclay

Foundations: Deduction in HOL

Overview

- Context and Motivation
- Foundations : Deduction
- Deduction Rules for HOL
- Formal Proofs
- Proof Construction
- Constructing Proofs in Isabelle
- Apply-style Proofs in Isabelle

Foundation: Introduction to Deduction

Motivation

- "Logic Whirl-Pool" of the 20ies (Girard) as response to foundational problems in Mathematics

- growing uneasiness over the question:

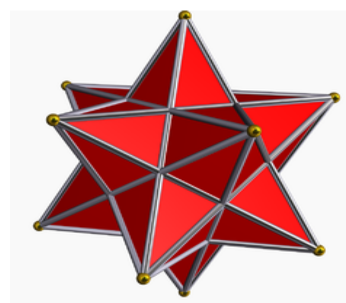
  - What is a logic / a proof ?

  - What is a consistent logic ?

  - Are there limits of provability ?

Deduction

- Historical context in the 20ies:
  - 1500 false proofs of „all parallels do not intersect in infinity"
  - lots of proofs and refutations of „all polyhedrons are eularian" (Lakatosz)



E = F + K - 2 ???

  - Frege's axiomatic set theory proven inconsistent by Russel
  - Science vs. Marxism debate (Popper)

Deduction

- Historical context in the 20ies:
  - this seemed quite far away from Leipnitz

    „Calculemus !"

    (We don`t agree ? Let`s calculate ...)

  - of what constitutes, well, the heart of

    Science ...

Deduction

- Historical context in the 20ies:
  - attempts to formalize the intuition of „deduction“ by Frege, Hilbert, Russel, Lukasiewics, ...
  - 2 Calculi presented by Gerhard Gentzen in 1934.
    - „natürliches Schliessen“

      (natural deduction):

$$\begin{array}{c}[P]\\ \vdots\\ Q\\ \hline R\end{array}$$

    - „Sequenzkalkül“ (sequent calculus)

$$\frac{\Gamma \vdash A \vee B \quad \Gamma \cup \{A\} \vdash C \quad \Gamma \cup \{B\} \vdash C}{\Gamma \vdash C}$$

Deduction

- An Inference System (or Logic) allows to infer formulas from a set of elementary judgements (axioms) and inferred judgements by rules:

$$\frac{A_1 \quad \ldots \quad A_n}{A_{n+1}}$$

"from the assumptions A_1 to A_n, you can infer the conclusion A_{n+1}." A rule with n=0 is an elementary fact. Variables occurring in the formulas A_n can be arbitrarily substituted.

Deduction

- judgements discussed in this course (or elsewhere):

$\Sigma, \Gamma \vdash t :: \tau$ "term t has type τ"

$\Gamma \vdash \phi$ "formula ϕ is valid under assumptions Γ"

$\vdash \{P\}\ x:= x+1\ \{Q\}$. "Hoare Triple"

ϕ prop "ϕ is a property"

ϕ valid "ϕ is a valid (true) property"

X mortal $\Longrightarrow$ sokrates mortal

--- judgements with free variable

etc …

Representing Logics

- An Inference System for the equality operator (presented in typed λ-calculus in Σ_{Pure}) looks like this:

$$\frac{}{(s = s)prop} \qquad \frac{(s = t)prop}{(t = s)prop} \qquad \frac{(r = s)prop \quad (s = t)prop}{(r = t)prop}$$

$$\frac{(s(x) = t(x))prop}{(s = t)prop}\, where\ x\ is\ fresh \qquad \frac{(s = t)prop \quad (P(s))prop}{(P(t))prop}$$

(where *prop* is *Trueprop* and " ―――― " is _⟹_).

Representing Logics

- the same thing presented a bit more neatly (not pretty-printing *Trueprop*, using $\bigwedge$_._ in Σ_{Pure}):

$$\frac{}{x = x} \qquad \frac{s = t}{t = s} \qquad \frac{r = s \quad s = t}{r = t}$$

$$\frac{\bigwedge x.\ s\ x = t\ x}{s = t} \qquad \frac{s = t \quad P\ s}{P\ t}$$

(equality on functions as above ("extensional equality") is an higher-order principle, and it makes this logic "classic").

Representing Logics

- the same thing presented as core logic in Isabelle/HOL) (not pretty-printing *Trueprop*, using $\bigwedge\_.\_$ in Σ_{Pure}):

$$\frac{}{x = x}\,\text{refl} \qquad \frac{s = t}{t = s}\,\text{sym} \qquad \frac{r = s \quad s = t}{r = t}\,\text{trans}$$

$$\frac{\bigwedge x.\; s\ x = t\ x}{s = t}\,\text{ext} \qquad \frac{s = t \quad P\ s}{P\ t}\,\text{subst}$$

(with the concrete names in Isabelle/HOL).

Foundation: Introduction to Deduction

„Pure": A (Meta)-Language for Deductive Systems

- Pure is a language to write logical rules (a "meta-logic")
- Higher-Order Logic (HOL) is our working logic.
- Equivalent notations for natural deduction rules (Textbook and Isabelle/HOL:)

$$\frac{A_1 \quad \ldots \quad A_n}{A_{n+1}}$$

$A_1 \Longrightarrow (\ldots \Longrightarrow (A_n \Longrightarrow A_{n+1})\ldots)$,

$[\![A_1; \ldots; A_n]\!] \Longrightarrow A_{n+1}$,

theorem
  assumes A_1
  and …
  and A_n
  shows A_{n+1}

„Pure": A (Meta)-Language for Deductive Systems

- Pure allows also to represent and reason over more complex rules involving the concept of "Discharge" of (hypothetical) assumptions:*

$(P \Longrightarrow Q) \Longrightarrow R$:

$$\frac{\begin{array}{c}[P]\\ \vdots \\ Q\end{array}}{R}$$

theorem
 assumes "$P \Longrightarrow Q$"
 shows "R"

* We follow the notation of van Dahlen's Book: "Logic and Structure". Available online.

„Pure": A (Meta)-Language for Deductive Systems

- Pure allows even more complex rules involving "local fresh variables" in sub-proofs:

$\bigwedge x.\ (P\ x \Longrightarrow Q\ x) \Longrightarrow R$:

$$\frac{\begin{array}{c}[P]_x \\ \vdots \\ Q\end{array}}{R}$$

theorem
fix x
assumes "P $\Longrightarrow$ Q"
shows "R"

„Pure": A (Meta)-Language for Deductive Systems

- Pure allows even more complex rules involving "local fresh variables" in sub-proofs:

Important Example:

$$\frac{P(0) \qquad \begin{array}{c}[P(n)]_n \\ \vdots \\ P(Suc\ n)\end{array}}{\forall x. P(x)}$$

Deduction Rules for HOL (in Isabelle/Pure)

Propositional Logic as ND calculus

- Some (almost) basic rules in HOL
  (and the names in Isabelle/HOL)

$$\frac{Q}{\neg\neg Q} \qquad \frac{\neg\neg Q}{Q}\,\text{notnotD} \qquad \frac{}{\neg\neg Q = Q}\,\text{notnot}$$

$$\frac{A}{A \vee B}\,\text{disjI1} \qquad \frac{B}{A \vee B}\,\text{disjI2} \qquad \frac{A \vee B \quad \begin{matrix}[A] \\ \vdots \\ Q\end{matrix} \quad \begin{matrix}[B] \\ \vdots \\ Q\end{matrix}}{Q}\,\text{disjE}$$

Propositional Logic as ND calculus

- Some (almost) basic rules in HOL

$$\frac{A \wedge B \qquad \begin{matrix}[A, B] \\ \vdots \\ Q\end{matrix}}{Q}\text{conjE}$$

$$\frac{A \qquad B}{A \wedge B}\text{conjI}$$

Propositional Logic as ND calculus

- Some (almost) basic rules in HOL

$$\frac{\begin{array}{c}[A]\\ \vdots\\ B\end{array}}{A \to B}\,\text{impI} \qquad \frac{A \to B \quad A}{B}\,\text{mp} \qquad \frac{A \to B \quad A \quad \begin{array}{c}[B]\\ \vdots\\ R\end{array}}{R}\,\text{impE}$$

Propositional Logic as ND calculus

- Some (almost) basic rules in HOL

$$\frac{\begin{array}{c}[A]\\ \vdots \\ B\end{array}}{A \to B}\,\text{impI} \qquad \frac{A \to B \quad A}{B}\,\text{mp} \qquad \frac{A \to B \quad A \quad \begin{array}{c}[B]\\ \vdots \\ R\end{array}}{R}\,\text{impE}$$

HOL Rules

- "Classic" consequences of not not (not true in a constructivistic version of HOL as used in the Coq-System)

$$\frac{\neg A \quad A}{Q}\ \text{notE} \qquad \frac{\begin{matrix}[\neg Q]\\ \vdots \\ False\end{matrix}}{Q}\ \text{contr} \qquad \frac{False}{Q}\ \text{FalseE}$$

HOL Rules

- The quantifier rules of HOL:

$$\frac{\forall x. P\ x \qquad \begin{matrix} [P\ ?t; \forall x. P\ x] \\ \vdots \\ Q \end{matrix}}{Q}$$

alldupE
(unsafe, but
complete)

HOL Rules

- The quantifier rules of HOL:

$$\frac{\forall x. P\ x \qquad \begin{matrix}[P\ ?t] \\ \vdots \\ Q\end{matrix}}{Q}$$

allE
(safe, but
incomplete)

HOL Rules

- The quantifier rules of HOL:

$$\frac{P\ ?t}{\exists x. P\ x}\ \text{exI}$$

$$\frac{\exists x.\ P(x) \qquad \begin{array}{c}[P(x)]_x \\ \vdots \\ Q\end{array}}{Q}\ \text{exE}$$

Formal Proofs

Key Concepts: Rule-Instances

- A Rule-Instance is a rule where the free variables in its judgements were substituted by a common substitution σ:

$$\frac{A \quad B}{A \wedge B}\,\text{conjI} \quad \xrightarrow{\ \sigma\ } \quad \frac{3 < x \quad x \leq y}{3 < x \wedge x \leq y}$$

where σ is $\{A \mapsto 3<x,\ B \mapsto x \leq y\}$.

Key Concepts: Formal Proofs

- A series of inference rule instances is usually displayed as a Proof Tree (or : Derivation or: Formal Proof)

$$\dfrac{\dfrac{\text{sym}\;\dfrac{f(a,b)=a}{a=f(a,b)} \qquad \dfrac{f(a,b)=a \qquad f(f(a,b),b)=c}{f(a,b)=c}\;\text{subst}}{a=c}\;\text{trans} \qquad \dfrac{}{g(a)=g(a)}\;\text{refl}}{g(a)=g(c)}\;\text{subst}$$

- The hypothetical facts at the leaves are called the assumptions of the proof (here $f(a,b) = a$ and $f(f(a,b),b) = c$).

Key Concepts: Discharge

- A key requisite of ND is the concept of discharge of assumptions allowed by some rules (like impl)

$$\dfrac{\begin{array}{c}[A]\\ \vdots \\ B\end{array}}{A \to B}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{[f(a,b)=a]}{a=f(a,b)}\,\text{sym} \qquad \dfrac{[f(a,b)=a] \quad f(f(a,b),b)=c}{f(a,b)=c}\,\text{subst}}{a=c}\,\text{trans} \qquad \dfrac{}{g(a)=g(a)}\,\text{refl}}{g(a)=g(c)}\,\text{subst}}{f(a,b)=a \to g(a)=g(c)}$$

- The set of assumptions is diminished by the discharged hypothetical facts of the proof (remaining: *f(f(a,b),b) = c*).

Key Concepts: Global Assumptions

- The set of (proof-global) assumptions gives rise to the notation:

$$\{f(a,b) = a, f(f(a,b),b) = c\} \vdash g(a) = g(c)$$

written:

$$A \vdash \varphi$$

or when emphasising the global theory
(also called: global context):

$$A \vdash_E \phi$$

Sequent-style calculus

- Gentzen introduced and alternative "style" to natural deduction: Sequent style rules.
  - Idea: using the tuples $A \vdash \varphi$ as basic judgments of the rules.
  - impl and impE look then like this:

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B} \qquad \frac{\Gamma \vdash A \to B \quad \Gamma \vdash A}{\Gamma \vdash B}$$

Sequent-style calculus

- in contrast to:

$$\frac{\begin{array}{c}[A]\\ \vdots \\ B\end{array}}{A \rightarrow B} \qquad \frac{A \rightarrow B \quad A}{B}$$

Sequent-style vs. ND calculus

- Both styles are linked by two transformations called "lifting over assumptions" Lifting over assumptions transforms:

$$\frac{A_1 \quad \ldots \quad A_n}{A_{n+1}}$$

$$\Updownarrow$$

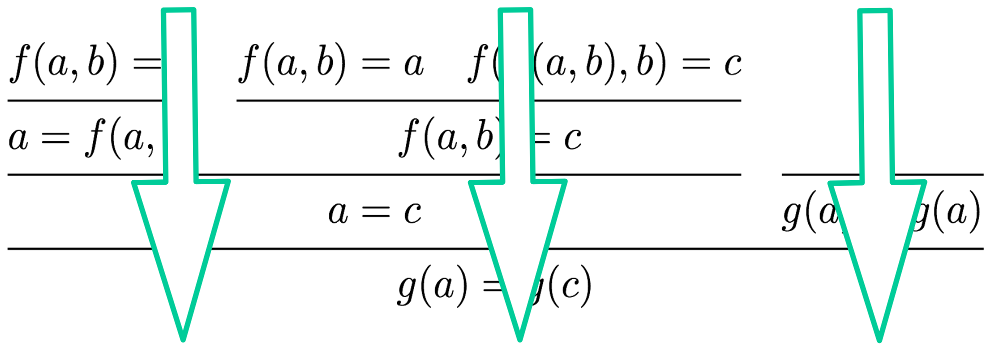
$$\frac{\Gamma \vdash A_1 \quad \ldots \quad \Gamma \vdash A_n}{\Gamma \vdash A_{n+1}}$$

where we consider for the moment $\vdash$ just equivalent to meta implication $\Longrightarrow$

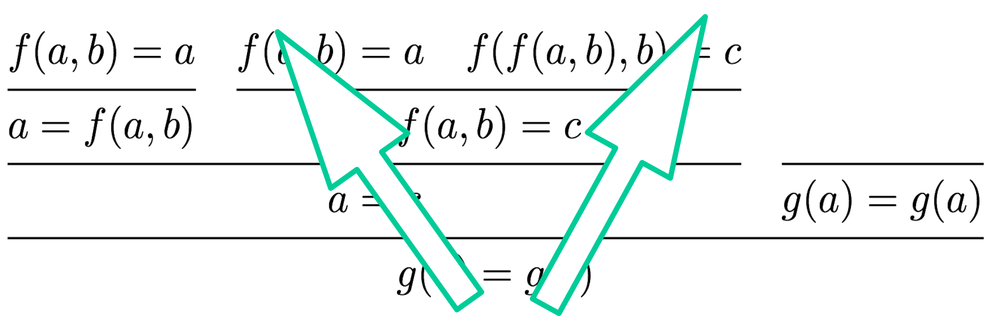
Constructing Proofs

Proof Construction

- Proofs can be constructed in two ways
  - Top down,
    from assumptions
    to conclusions
    (Forward chaining)



  - Bottom up,
    decomposing conclusions
    to necessary assumptions
    (Backward Chaining)



Proof Construction

- Forward Chaining / Forward Reasoning
  - Often intuitive for humans
  - Needs decomposition of assumptions
  - Needs "hindsight" towards the ultimate proof goal "guessing" the right substitutions for rule-instances
  - Forward Reasoning is done by elimination rules
  - In Isabelle indexed by _E :
    notE, conjE, disjE, impE

$$\frac{A \vee B \quad \begin{array}{c}[A]\\ \vdots \\ Q\end{array} \quad \begin{array}{c}[B]\\ \vdots \\ Q\end{array}}{Q} \qquad \frac{A \to B \quad A \quad \begin{array}{c}[B]\\ \vdots \\ R\end{array}}{R}$$

  - A destructive variant of eliminations are destruction-rules. They allows transformations in assumptions.
  - In Isabelle (usually) indexed by _D:

$$\frac{\neg\neg Q}{Q} \qquad \frac{A \to B \quad A}{B}$$

Proof Construction

- Backward Chaining / Backward Reasoning
  - Often deterministic in a logic:
    we know which rules to apply from the
    syntactic structure of the root goal
  - Rule instances can often be constructed automatically
  - Schematic variables may help to delay decisions
  - Backward reasoning can lead in a loop
  - Backward reasoning is done by introduction rules
  - Suited rules are indexed by _I in Isabelle:
    conjI, disjI1, impI, …

$$\frac{A \quad B}{A \land B}\,\text{conjI} \qquad \frac{A}{A \lor B}\,\text{disjI1} \qquad \frac{\begin{matrix}[A]\\ \vdots \\ B\end{matrix}}{A \to B}\,\text{impI}$$

Proof Construction: Quantifiers

- For exI, allE, Isabelle allows schematic variables ?X, ?Y, ?Z that represent „holes" in a term that can be filled in later by substitution; Coq requires the instantiation when applying the rule.

- Isabelle uses a built-in ("meta")-quantifier $\bigwedge x.\ P\ x$ already seen; Coq uses internally a similar concept not explicitly revealed to the user.

Constructing apply-style Proofs in Isabelle

Apply-Style Proofs

- Isabelle supports a proof language for step-wise backwards proofs: "apply style" proofs
- General format:

```
lemma <name> : "<formula>"
      apply(<method>)
      …
      apply(<method>)
      done
```

- Abbreviation:

by(<method>) is apply(<method>) done

Apply-Style Proofs

- Isabelle displays intermediate steps in a format inspired by a sequent-calculus:
  - Each open "branch" is represented by a "subgoal"
  - Each subgoal is represented as a rule, meaning:

under assumptions $A_1 \ldots A_n$, it remains to show A_{n+1}

- A method is usually applied to the first "subgoal"
- "done" closes a proof (if possible) and stores the lemma as theorem (a "<thm>")
- Isabelle manages a data-base of theorems (recall "find_theorem "name"" or "find_theorem "pattern" for search)

Apply-Style Proofs

- core - methods at a glance

assumption	— discharge conclusion
rule <thm>	— introduction rules
erule <thm>	— elimination rules
drule <thm>	— destruction rule

- Variants avec substitution

```
rule_tac <substitution> in <thm>
erule_tac <substitution> in <thm>
drule_tac <substitution> in <thm>
```

Apply-Style Proofs

- Useful operation:

```
unfolding <thm> … <thm>
prefer n                    — rearraging goals
defer n
```

- Derived methods for one-step rewrites of an eqn:

```
subst <thm>
subst <thm>[symmetric]     — "fold"
subst (asm) <thm>
```

Conclusion

- Higher-Order Logic can be easily represented in typed λ-calculus,
- … that includes also its rules
- Rules can be derived in Pure;
  HOL rules are "first-order citizens"(and not built-in)
- Isabelle supports backward and forward reasoning
- … actually in several proof languages.