COMP 4161
NICTA Advanced Course

Advanced Topics in Software Verification

Gerwin Klein, June Andronick, Toby Murray

HOL

Slide 1

Content

Rough timeline
- Intro & motivation, getting started [1]

- Foundations & Principles
  - Lambda Calculus, natural deduction [2,3,4*a]
  - Higher Order Logic [5,6,7]
  - Term rewriting [8,9,10*c]

- Proof & Specification Techniques
  - Isar [11,12*d]
  - 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]

*a1 out; *a1 due; *a2 out; *a2 due; *session break; *a3 out; *a3 due

Slide 2

Slide 3

Quantifiers

Scope
- Scope of parameters: whole subgoal
- Scope of \( \forall, \exists, \ldots \) ends with \( ; \) or \( \Rightarrow \)

Example:

\[ \Lambda x y. \{ \forall y. P y \Rightarrow Q z y ; \ Q x y \} \Rightarrow \exists x. \ Q x y \]

means

\[ \Lambda x y. \{ \forall y_1. P y_1 \Rightarrow Q z y_1 ; \ Q x y \} \Rightarrow \exists x_1. \ Q x_1 y \]

Slide 4
Natural deduction for quantifiers

\[ \forall x. P x \quad \forall x. \neg P x \quad \forall x. P x \rightarrow R \quad \forall x. \neg P x \rightarrow R \quad \forall x. \neg P x \rightarrow R \quad \forall x. P x \rightarrow R \]

- \text{allI} and \text{allE} introduce new parameters (\(\forall x\)).
- \text{exI} and \text{exE} introduce new unknowns (\(\exists x\)).

Two Successful Proofs

1. \( \forall x. \exists y. x = y \)
   apply (rule \text{allI})

1. \( \forall x. \exists y. x = y \)
   apply (rule \text{exI})

1. \( \forall y. x = y \)
   apply (rule \text{refl})

1. \( \forall x. x = y \)
   apply (rule \text{refl})

Simpler & clearer
Shorter & trickier

Instancing Rules

apply (rule \text{Jac} x = "term" in line)

Like rule, but \(\exists x\) in rule is instantiated by term before application.

Similar: erule \text{Jac}

\[ x \text{ is in rule, not in goal } \]

Two Unsuccessful Proofs

1. \( \exists y. \forall x. x = y \)
   apply (rule \text{Jac} x = ??? in exl)

1. \( \forall x. x = y \)
   apply (rule \text{refl})

Principle:

\( \text{if } x_1 \ldots x_n \text{ can only be replaced by term } t \) 

\( \text{if } \text{params}(t) \subseteq x_1 \ldots x_n \)
Safe and Unsafe Rules

Safe: allI, exE

Unsafe: allE, exI

Create parameters first, unknowns later

Parameter names

Parameter names are chosen by Isabelle

1. \( \forall x. \exists y. x = y \)
   apply (rule allI)

1. \( \exists x. \forall y. x = y \)
   apply (rename_tac x = "x" in exI)

Brittle!

Renaming parameters

1. \( \forall x. \exists y. x = y \)
   apply (rule allI)

1. \( \forall x. \exists y. x = y \)
   apply (rename_tac N)

1. \( \exists N. \forall y. N = y \)
   apply (rename_tac x = "N" in exI)

In general:

(rename_tac \( x_1 \ldots x_n \)) renames the rightmost (inner) \( n \) parameters to \( x_1 \ldots x_n \)
Forward Proof: frule and drule

apply (frule < rule >)

Rule:

\[ [A_1; \ldots; A_m] \implies A \]

Subgoal:

1. \([B_1; \ldots; B_n] \implies C\)

Substitution:

\(\sigma(B_i) \equiv \sigma(A_1)\)

New subgoals:

1. \(\sigma([B_1; \ldots; B_n] \implies A_2)\)
   
   : 
   
   m-1. \(\sigma([B_1; \ldots; B_n] \implies A_m)\)
   
   m. \(\sigma([B_1; \ldots; B_n; A] \implies C)\)

Like frule but also deletes \(B_i\):

\(\text{apply (drule < rule >)}\)

Examples for Forward Rules

\[
\begin{align*}
P \land Q & \quad \text{conjunct1} \\
P \land Q & \quad \text{conjunct2} \\
\end{align*}
\]

\[
\begin{align*}
P \implies Q & \quad \text{mp} \\
\forall x. P(x) & \quad \text{spec} \\
\end{align*}
\]

Slide 13

Forward Proof: OF

\(r(\text{OF } r_1 \ldots r_n)\)

Prove assumption 1 of theorem \(r\) with theorem \(r_1\), and assumption 2 with theorem \(r_2\), and ...

Rule \(r\): \([A_1; \ldots; A_m] \implies A\)

Rule \(r_1\): \([B_1; \ldots; B_n] \implies B\)

Substitution \(\sigma(B) \equiv \sigma(A_1)\)

\(r(\text{OF } r_1)\): \(\sigma([B_1; \ldots; B_n; A_1; \ldots; A_m] \implies A)\)

Slide 15

Forward proofs: THEN

\(r_1 \text{ [THEN } r_2\) means \(r_2 \text{ [OF } r_1]\)

Slide 16
**DEMO: FORWARD PROOFS**

**Hilbert's Epsilon Operator**

(David Hilbert, 1862-1943)

\( \varepsilon x. P x \) is a value that satisfies \( P \) (if such a value exists)

\( \varepsilon \) also known as **description operator**.

In Isabelle the \( \varepsilon \)-operator is written \( \text{SOME} \times P x \)

\[ P \varepsilon x. P x \]

**More Epsilon**

\( \varepsilon \) implies Axiom of Choice:

\( \forall x. \exists y. Q x y \iff \exists f. \forall x. Q x (f x) \)

Existential and universal quantification can be defined with \( \varepsilon \).

Isabelle also knows the definite description operator \( \text{THE} \) (aka \( \iota \)):

\[ \text{THE} \times x. x = a \]

**Some Automation**

**More Proof Methods:**

- **apply (intro < intro-rules >)** repeatedly applies intro rules
- **apply (elim < elim-rules >)** repeatedly applies elim rules
- **apply clarify** applies all safe rules that do not split the goal
- **apply safe** applies all safe rules
- **apply blast** an automatic tableau prover (works well on predicate logic)
- **apply fast** another automatic search tactic
We have learned so far...

- Proof rules for predicate calculus
- Safe and unsafe rules
- Forward Proof
- The Epsilon Operator
- Some automation

We said that $\varepsilon$ implies the Axiom of Choice:

$$\forall x. \exists y. Q x y \implies \exists f. \forall x. Q x (f x)$$

Prove the axiom of choice as a lemma, using only the introduction and elimination rules for $\forall$ and $\exists$, namely $\text{allI}$, $\text{exI}$, $\text{allE}$, $\text{exE}$, and the introduction rule for $\varepsilon$, $\text{someI}$, using only the proof methods $\text{rule_tac}$, $\text{erule}$, $\text{erule}$, $\text{erule_tac}$ and $\text{assumption}$.