NICTA

COMP 4161
NICTA Advanced Course

Advanced Topics in Software Verification

Gerwin Klein
Formal Methods

HOL

NICTA

Content

- Intro & motivation, getting started with Isabelle
- **Foundations & Principles**
  - Lambda Calculus
  - **Higher Order Logic, natural deduction**
  - Term rewriting
- Proof & Specification Techniques
  - Datatypes, recursion, induction
  - Inductively defined sets, rule induction
  - Calculational reasoning, mathematics style proofs
  - Hoare logic, proofs about programs

NICTA

Last Time on HOL

- natural deduction rules for $\wedge$, $\vee$ and $\longrightarrow$
- proof by assumption
- proof by intro rule
- proof by elim rule

NICTA

More Proof Rules

Iff, Negation, True and False

$$\frac{A \Longrightarrow B \quad B \Longrightarrow A}{A = B}\ \text{iffI} \qquad \frac{A = B \quad [\![A \longrightarrow B; B \longrightarrow A]\!] \Longrightarrow C}{C}\ \text{iffE}$$

$$\frac{A = B}{A \Longrightarrow B}\ \text{iffD1} \qquad \frac{A = B}{B \Longrightarrow A}\ \text{iffD2}$$

$$\frac{A \Longrightarrow False}{\neg A}\ \text{notI} \qquad \frac{\neg A \quad A}{P}\ \text{notE}$$

$$\frac{}{\mathit{True}}\ \text{TrueI} \qquad \frac{\mathit{False}}{P}\ \text{FalseE}$$

Equality

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

$$\frac{s = t \quad P\ s}{P\ t}\ \text{subst}$$

Rarely needed explicitly — used implicitly by term rewriting

Classical

$$\frac{}{P = True \vee P = False}\ \text{True-False}$$

$$\frac{}{P \vee \neg P}\ \text{excluded-middle}$$

$$\frac{\neg A \Longrightarrow False}{A}\ \text{ccontr} \qquad \frac{\neg A \Longrightarrow A}{A}\ \text{classical}$$

- ➜ **excluded-middle**, **ccontr** and **classical** not derivable from the other rules.
- ➜ if we include True-False, they are derivable

They make the logic “classical”, “non-constructive”

Cases

$$\frac{}{P \vee \neg P}\ \text{excluded-middle}$$

is a case distinction on type $bool$

Isabelle can do case distinctions on arbitrary terms:

apply (case_tac $term$)

Safe and not so safe

Safe rules preserve provability

conjI, impI, notI, iffI, refl, ccontr, classical, conjE, disjE

$\dfrac{A \quad B}{A \wedge B}$ conjI

Unsafe rules can turn a provable goal into an unprovable one

disjI1, disjI2, impE, iffD1, iffD2, notE

$\dfrac{A}{A \vee B}$ disjI1

Apply safe rules before unsafe ones

Demo

Quantifiers

Scope

- Scope of parameters: whole subgoal
- Scope of $\forall, \exists, \ldots$: ends with ; or $\Longrightarrow$

Example:

$$\bigwedge x\ y.\ [\![\forall y.\ P\ y \longrightarrow Q\ z\ y;\ \ Q\ x\ y]\!] \Longrightarrow \exists x.\ Q\ x\ y$$

means

$$\bigwedge x\ y.\ [\![(\forall y_1.\ P\ y_1 \longrightarrow Q\ z\ y_1);\ \ Q\ x\ y]\!] \Longrightarrow (\exists x_1.\ Q\ x_1\ y)$$

Natural deduction for quantifiers

$$\frac{\bigwedge x.\ P\ x}{\forall x.\ P\ x}\ \text{allI} \qquad \frac{\forall x.\ P\ x \quad P\ ?x \Longrightarrow R}{R}\ \text{allE}$$

$$\frac{P\ ?x}{\exists x.\ P\ x}\ \text{exI} \qquad \frac{\exists x.\ P\ x \quad \bigwedge x.\ P\ x \Longrightarrow R}{R}\ \text{exE}$$

- **allI** and **exE** introduce new parameters ($\bigwedge x$).
- **allE** and **exI** introduce new unknowns ($?x$).

Slide 13

Instantiating Rules

apply (rule_tac x = "$term$" in $rule$)

Like **rule**, but $?x$ in $rule$ is instantiated by $term$ before application.

Similar: **erule_tac**

! x **is in** $rule$**, not in goal** **!**

Slide 14

Two Successful Proofs

1. $\forall x.\ \exists y.\ x = y$

apply (rule allI)

1. $\bigwedge x.\ \exists y.\ x = y$

best practice	exploration
apply (rule_tac x = "x" in exI)	**apply** (rule exI)
1. $\bigwedge x.\ x = x$	1. $\bigwedge x.\ x = ?y\ x$
apply (rule refl)	**apply** (rule refl)
	$?y \mapsto \lambda u.u$
simpler & clearer	**shorter & trickier**

Slide 15

Two Unsuccessful Proofs

1. $\exists y.\ \forall x.\ x = y$

apply (rule_tac x = ??? in exI)	**apply** (rule exI)
	1. $\forall x.\ x = ?y$
	apply (rule allI)
	1. $\bigwedge x.\ x = ?y$
	apply (rule refl)
	$?y \mapsto x$ yields $\bigwedge x'.x' = x$

Principle:

$?f\ x_1 \ldots x_n$ **can only be replaced by term** t
if $params(t) \subseteq x_1, \ldots, x_n$

Slide 16

Safe and Unsafe Rules

Safe allI, exE

Unsafe allE, exI

Create parameters first, unknowns later

Slide 17

Demo: Quantifier Proofs

Slide 18

Parameter names

Parameter names are chosen by Isabelle

1. $\forall x.\ \exists y.\ x = y$

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

apply (rule_tac x = "x" in exI)

Brittle!

Slide 19

Renaming parameters

1. $\forall x.\ \exists y.\ x = y$

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

apply (rename_tac N)
1. $\bigwedge N.\ \exists y.\ N = y$

apply (rule_tac x = "N" in exI)

In general:
(rename_tac $x_1 \ldots x_n$) renames the rightmost (inner) n parameters to $x_1 \ldots x_n$

Slide 20

Forward Proof: frule and drule

apply (frule $< rule >$)

Rule:	$[\![A_1; \dots; A_m]\!] \Longrightarrow A$
Subgoal:	1. $[\![B_1; \dots; B_n]\!] \Longrightarrow C$
Substitution:	$\sigma(B_i) \equiv \sigma(A_1)$
New subgoals:	1. $\sigma([\![B_1; \dots; B_n]\!] \Longrightarrow A_2)$
	$\vdots$
	m-1. $\sigma([\![B_1; \dots; B_n]\!] \Longrightarrow A_m)$
	m. $\sigma([\![B_1; \dots; B_n; A]\!] \Longrightarrow C)$

Like **frule** but also deletes B_i: **apply** (drule $< rule >$)

Examples for Forward Rules

$$\frac{P \wedge Q}{P}\ \text{conjunct1} \qquad \frac{P \wedge Q}{Q}\ \text{conjunct2}$$

$$\frac{P \longrightarrow Q \quad P}{Q}\ \text{mp}$$

$$\frac{\forall x.\ P\ x}{P\ ?x}\ \text{spec}$$

Forward Proof: OF

r [**OF** $r_1 \dots r_n$]

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

Rule r	$[\![A_1; \dots; A_m]\!] \Longrightarrow A$
Rule r_1	$[\![B_1; \dots; B_n]\!] \Longrightarrow B$
Substitution	$\sigma(B) \equiv \sigma(A_1)$
r [OF r_1]	$\sigma([\![B_1; \dots; B_n; A_2; \dots; A_m]\!] \Longrightarrow A)$

Forward proofs: THEN

r_1 [THEN r_2] means r_2 [OF r_1]

NICTA

Demo: Forward Proofs

NICTA

Hilbert's Epsilon Operator

(David Hilbert, 1862-1943)

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

ε also known as **description operator**.
In Isabelle the ε-operator is written SOME $x.\ P\ x$

$$\frac{P\ ?x}{P\ (\text{SOME}\ x.\ P\ x)}\ \text{someI}$$

NICTA

More Epsilon

ε implies Axiom of Choice:

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

Existential and universal quantification can be defined with ε.

Isabelle also knows the definite description operator **THE** (aka ι):

$$\frac{}{(\text{THE}\ x.\ x = a) = a}\ \text{the\_eq\_trivial}$$

NICTA

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 tableaux prover (works well on predicate logic)
apply fast	another automatic search tactic

NICTA

EPSILON AND AUTOMATION DEMO

Slide 29

NICTA

WE HAVE LEARNED SO FAR...

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

Slide 30