NICTA

**COMP 4161**

NICTA Advanced Course

**Advanced Topics in Software Verification**

Gerwin Klein
Formal Methods

**Slide 1**

## ORGANISATORIALS

NICTA

| | | |
|---|---|---|
| **When** | Mon | 13:00 – 14:30 |
| | Wed | 13:00 – 14:30 |
| **Where** | Mon: | Webst 250 |
| | Wed: | Law Th G23 |

`http://www.cse.unsw.edu.au/~cs4161/`

**Slide 2**



## WHAT YOU WILL LEARN

NICTA

- ➜ how to use a theorem prover
- ➜ background, how it works
- ➜ how to prove and specify
- ➜ how to reason about programs

**Health Warning**

**Theorem Proving is addictive**

**Slide 3**

## CONTENT — USING THEOREM PROVERS

NICTA

- ➜ Intro & motivation, getting started (today)
- ➜ 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

**Slide 4**

## Credits

some material (in using-theorem-provers part) shamelessly stolen from

Tobias Nipkow, Larry Paulson, Markus Wenzel

David Basin, Burkhardt Wolff

**Don't blame them, errors are mine**



## What is a proof?

**to prove** (Marriam-Webster)

- ➜ from Latin probare (test, approve, prove)
- ➜ to learn or find out by experience (archaic)
- ➜ to establish the existence, truth, or validity of (by evidence or logic)
  *prove a theorem, the charges were never proved in court*

**pops up everywhere**

- ➜ politics (weapons of mass destruction)
- ➜ courts (beyond reasonable doubt)
- ➜ religion (god exists)
- ➜ science (cold fusion works)



## What is a mathematical proof?

**In mathematics, a proof is a demonstration that, given certain axioms, some statement of interest is necessarily true.** (Wikipedia)

**Example:** $\sqrt{2}$ is not rational.

Proof: assume there is $r \in \mathbb{Q}$ such that $r^2 = 2$.

Hence there are mutually prime $p$ and $q$ with $r = \frac{p}{q}$.

Thus $2q^2 = p^2$, i.e. $p^2$ is divisible by $2$.

$2$ is prime, hence it also divides $p$, i.e. $p = 2s$.

Substituting this into $2q^2 = p^2$ and dividing by $2$ gives $q^2 = 2s^2$. Hence, $q$ is also divisible by $2$. Contradiction. Qed.



## Nice, but..

- ➜ still not rigorous enough for some
  - what are the rules?
  - what are the axioms?
  - how big can the steps be?
  - what is obvious or trivial?
- ➜ informal language, easy to get wrong
- ➜ easy to miss something, easy to cheat

**Theorem.** A cat has nine tails.
**Proof.** No cat has eight tails. Since one cat has one more tail than no cat, it must have nine tails.

## What is a formal proof?

**A derivation in a formal calculus**

**Example:** $A \wedge B \longrightarrow B \wedge A$ derivable in the following system

**Rules:** $\frac{X \in S}{S \vdash X}$ (assumption) $\frac{S \cup \{X\} \vdash Y}{S \vdash X \longrightarrow Y}$ (impl)

$\frac{S \vdash X \quad S \vdash Y}{S \vdash X \wedge Y}$ (conjI) $\frac{S \cup \{X, Y\} \vdash Z}{S \cup \{X \wedge Y\} \vdash Z}$ (conjE)

**Proof:**

| | | |
|---|---|---|
| 1. | $\{A, B\} \vdash B$ | (by assumption) |
| 2. | $\{A, B\} \vdash A$ | (by assumption) |
| 3. | $\{A, B\} \vdash B \wedge A$ | (by conjI with 1 and 2) |
| 4. | $\{A \wedge B\} \vdash B \wedge A$ | (by conjE with 3) |
| 5. | $\{\} \vdash A \wedge B \longrightarrow B \wedge A$ | (by impl with 4) |





## What is a theorem prover?

**Implementation of a formal logic on a computer.**

- ➜ fully automated (propositional logic)
- ➜ automated, but not necessarily terminating (first order logic)
- ➜ with automation, but mainly interactive (higher order logic)

- ➜ based on rules and axioms
- ➜ can deliver proofs

There are other (algorithmic) verification tools:

- ➜ model checking, static analysis, ...
- ➜ usually do not deliver proofs







## Why theorem proving?

- ➜ Analysing systems/programs thoroughly
- ➜ Finding design and specification errors early
- ➜ High assurance (mathematical, machine checked proof)
- ➜ it's not always easy
- ➜ it's fun





## Main theorem proving system for this course

Isabelle

- ➜ used here for applications, learning how to prove

## WHAT IS ISABELLE?

**A generic interactive proof assistant**

➜ **generic:**
not specialised to one particular logic
(two large developments: HOL and ZF, will mainly use HOL)

➜ **interactive:**
more than just yes/no, you can interactively guide the system

➜ **proof assistant:**
helps to explore, find, and maintain proofs





## WHY ISABELLE?

➜ free
➜ widely used systems
➜ active development
➜ high expressiveness and automation
➜ reasonably easy to use
➜ (and because I know it best ;-))







**If I prove it on the computer, it is correct, right?**





## IF I PROVE IT ON THE COMPUTER, IT IS CORRECT, RIGHT?

**No, because:**

① hardware could be faulty
② operating system could be faulty
③ implementation runtime system could be faulty
④ compiler could be faulty
⑤ implementation could be faulty
⑥ logic could be inconsistent
⑦ theorem could mean something else

## If I prove it on the computer, it is correct, right?

**No, but:**

probability for

- ➜ 1 and 2 reduced by using different systems
- ➜ 3 and 4 reduced by using different compilers
- ➜ faulty implementation reduced by right architecture
- ➜ inconsistent logic reduced by implementing and analysing it
- ➜ wrong theorem reduced by expressive/intuitive logics

**No guarantees, but assurance immensly higher than manual proof**

Slide 17

## If I prove it on the computer, it is correct, right?

**Soundness architectures**

| | |
|---|---|
| careful implementation | PVS |
| LCF approach, small proof kernel | HOL4<br>Isabelle |
| explicit proofs + proof checker | Coq<br>Twelf<br>Isabelle<br>HOL4 |

Slide 18

## Meta Logic

**Meta language:**
The language used to talk about another language.

**Examples:**
English in a Spanish class, English in an English class

**Meta logic:**
The logic used to formalize another logic

**Example:**
Mathematics used to formalize derivations in formal logic

Slide 19

## Meta Logic – Example

**Syntax:**

Formulae: $F ::= V \quad | \quad F \longrightarrow F \quad | \quad F \wedge F \quad | \quad False$

$V ::= [A - Z]$

Derivable: $S \vdash X$ $\quad X$ a formula, $S$ a set of formulae

logic / meta logic

$$\frac{X \in S}{S \vdash X} \qquad \frac{S \cup \{X\} \vdash Y}{S \vdash X \longrightarrow Y}$$

$$\frac{S \vdash X \quad S \vdash Y}{S \vdash X \wedge Y} \qquad \frac{S \cup \{X, Y\} \vdash Z}{S \cup \{X \wedge Y\} \vdash Z}$$

Slide 20

## ISABELLE'S META LOGIC

NICTA

$\bigwedge \qquad \Longrightarrow \qquad \lambda$

Slide 21

## $\bigwedge$

NICTA

**Syntax:** $\bigwedge x.\ F$ ($F$ another meta level formula)

in ASCII: `!!x. F`

- ➜ universal quantifier on the meta level
- ➜ used to denote parameters
- ➜ example and more later

Slide 22



## $\Longrightarrow$

NICTA

**Syntax:** $A \Longrightarrow B$ ($A, B$ other meta level formulae)

in ASCII: `A ==> B`

**Binds to the right:**

$$A \Longrightarrow B \Longrightarrow C \quad = \quad A \Longrightarrow (B \Longrightarrow C)$$

**Abbreviation:**

$$[\![A; B]\!] \Longrightarrow C \quad = \quad A \Longrightarrow B \Longrightarrow C$$

- ➜ read: $A$ and $B$ implies $C$
- ➜ used to write down rules, theorems, and proof states

Slide 23

## EXAMPLE: A THEOREM

NICTA

**mathematics:** if $x < 0$ and $y < 0$, then $x + y < 0$

**formal logic:** $\vdash\ x < 0 \wedge y < 0 \longrightarrow x + y < 0$

variation: $x < 0; y < 0\ \vdash\ x + y < 0$

**Isabelle:** **lemma** "$x < 0 \wedge y < 0 \longrightarrow x + y < 0$"

variation: **lemma** "$[\![x < 0; y < 0]\!] \Longrightarrow x + y < 0$"

variation: **lemma**

assumes "$x < 0$" and "$y < 0$" shows "$x + y < 0$"

Slide 24

## EXAMPLE: A RULE

NICTA

**logic:** $\dfrac{X \quad Y}{X \wedge Y}$

**variation:** $\dfrac{S \vdash X \quad S \vdash Y}{S \vdash X \wedge Y}$

**Isabelle:** $[\![X; Y]\!] \Longrightarrow X \wedge Y$

Slide 25

## EXAMPLE: A RULE WITH NESTED IMPLICATION

NICTA

**logic:** $\dfrac{X \vee Y \quad \begin{matrix} X \\ \vdots \\ Z \end{matrix} \quad \begin{matrix} Y \\ \vdots \\ Z \end{matrix}}{Z}$

**variation:** $\dfrac{S \cup \{X\} \vdash Z \quad S \cup \{Y\} \vdash Z}{S \cup \{X \vee Y\} \vdash Z}$

**Isabelle:** $[\![X \vee Y; X \Longrightarrow Z; Y \Longrightarrow Z]\!] \Longrightarrow Z$

Slide 26

## $\lambda$

NICTA

**Syntax:** $\lambda x.\ F$ ($F$ another meta level formula)

in ASCII: `%x. F`

- ➜ lambda abstraction
- ➜ used for functions in object logics
- ➜ used to encode bound variables in object logics
- ➜ more about this in the next lecture

Slide 27

NICTA

## ENOUGH THEORY!
## GETTING STARTED WITH ISABELLE

Slide 28