
DTU

DTU

Lessons of Teaching Formal Methods with Isabelle

Frederik Krogsdal Jacobsen Jørgen Villadsen

Technical University of Denmark

Introduction

Experiences teaching two courses

Isabelle/Pure and Isabelle/HOL

Why is it hard to learn Isabelle?

Tools interfacing with Isabelle

Our undergraduate course

Logical systems and logic programming

Run for many years

Recently introduced more and more Isabelle

Topics: propositional and first-order logic, Hilbert systems, tableaux, sequent calculus, resolution

. . . and programming in Prolog

80+ students

Our graduate course

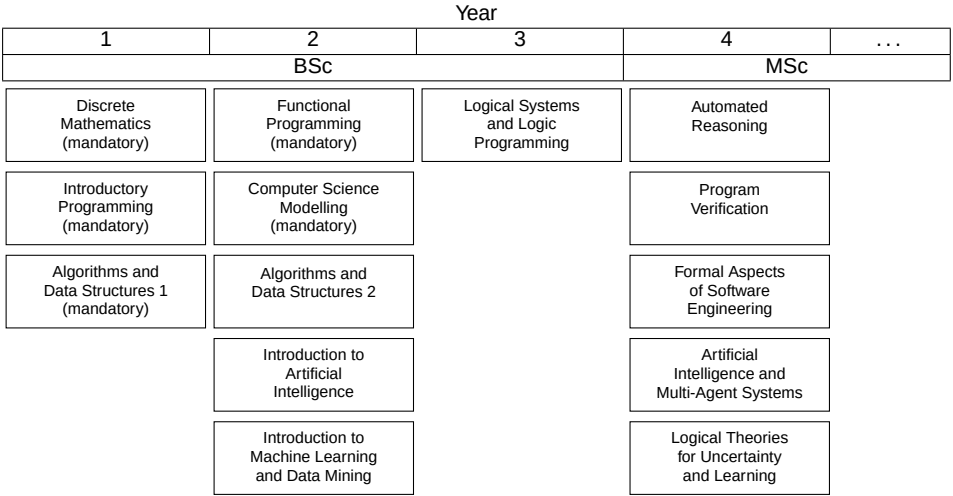
Automated reasoning

Has run only a few times

Topics: Isabelle, higher-order logic, type theory, practical formal proofs, verification of functional programs, automated theorem provers

40+ students

The surrounding curriculum



Natural Deduction Assistant

Graphical interface for natural deduction proofs

Classical first-order logic with functions

Metatheory formalized in Isabelle

Impossible to make syntax mistakes, and suggests applicable proof rules automatically (i.e. impossible to apply a rule wrong)

Easy to use, but annoyingly slow after a while

Sequent Calculus Verifier

Textual interface for sequent calculus proofs

Same logic and metatheory as NaDeA

Possible to make syntax mistakes, does not suggest proof rules, user must write out result of applying rule manually

No special characters or order of precedence (all parentheses required)

Still gives good warnings/errors if proof rules are applied wrong

Slightly harder to use, but quite fast

Looks quite a lot like “manual” proofs in Isabelle

Intuitionistic propositional logic

Formalization in Isabelle/Pure (heavily inspired by Makarius' examples)

Why? No clutter, just the rules

No automation

Students are forced to write structured proofs and think about which rules to use

Intuitionistic higher-order logic

Introduce higher-order logic

More involved examples

Learning how to work with quantifiers

Classical higher-order logic

Essentially just Isabelle/HOL, but with no automation

Learning how to approach proofs by contradiction through various possible rules

Quite involved examples

Builds a good understanding of what automation does under the hood

The basics of Isabelle

In parallel with learning logic, we teach formal verification of functional programs

Getting used to the syntax of Isabelle and the use of Isabelle/jEdit (or Isabelle/VSCode)

Getting back up to speed on functional programming

Students generally have a hard time for the first several weeks

The documentation they get by searching online (i.e. the Isar reference manual) is difficult for them to understand

Automation

We introduce basic automation (i.e. auto) quite quickly, but do not explain what it does immediately

When students have worked through the Isabelle/Pure section of the course, they can understand what the automation does

We exhibit some basic automated theorem provers (for SeCaV) to give students an idea of how proof search can be implemented

The exercises

Widely varying difficulty

Some are too hard for any students to finish without substantial help

Summary

Isabelle is very complex, so it is difficult for beginners to jump in

We start outside of Isabelle and slowly build up to the “full experience”

It seems like we need more relatively easy exercises

Students need to get their hands dirty to succeed

Ideas and future work

Beginner version of the reference manual

More easy exercises in the tutorial

“Project”-based exercises that guide students along

Better integration between our tools and Isabelle

We’re working on it!