Computer-Checked Meta-Logic

Jørgen Villadsen

jovi@dtu.dk

Abstract

Over the past decades there have been several impressive results in computer-checked meta-logic, including the correctness of provers for first-order and higher-order logic as well as the first-ever formalization of Gödel’s Incompleteness Theorems, recently published by Larry Paulson as a 250+ pages document for the Isabelle prover. I give a brief overview of the research area and discuss the prospects for verified and self-verified systems.

The overall aim is to develop verified systems....

... and in particular self-verified systems — or getting more or less close

The verification is done using a prover for a logic

Computer-Checked Logic

For self-verification the verified system is a prover for the logic itself

But often the verified system is a prover for a part of the logic

Like a prover for first-order logic in a prover for higher-order logic

Computer-Checked Meta-Logic

A Computer-Checked Proof of the Four Colour Theorem

Georges Gonthier
Microsoft Research Cambridge
2005

`http://research.microsoft.com/en-us/um/people/gonthier/4colproof.pdf`

... The other 60,000 or so lines of the proof can be read for insight or even entertainment, but need not be reviewed for correctness. That is the job of the Coq proof assistant, a job for computers...

One alleged proof was given by Alfred Kempe in 1879

The proof by Appel and Haken in 1976 was informal but used a computer program for some calculations

Isabelle Proof Assistant — Since 1985...

It may appear abstract but it is really concrete in the Isabelle prover

More about Isabelle and other provers in a moment

The turnstile symbol $\vdash$ is used for a prover (possibly with a label)

It was first used by Gottlob Frege in 1879 for the first prover: *Begriffsschrift*

$\Gamma \vdash \phi$ means that the prover can use the theory Γ to prove the theorem ϕ

In practice more information is required and produced

Isabelle requires a THY file and produces a PDF file if it can prove all theorems

Powerful automation is available and normally only conservative extensions are allowed for the theory

Isabelle Theory Files

A THY file is an extension of a TEX file

From the manual:

> This document is for those Isabelle users who have mastered the art of mixing LaTeX text and Isabelle theories and never want to typeset a theorem by hand anymore because they have experienced the bliss of writing
>
> `@{thm[display]setsum_cartesian_product[no_vars]}`
>
> and seeing Isabelle typeset it for them:
>
> $$(\sum x{\in}A.\ \sum y{\in}B.\ f\ x\ y) = (\sum (x,\ y){\in}A \times B.\ f\ x\ y)$$
>
> No typos, no omissions, no sweat.

Isabelle: Open Source Prover

Isabelle is a generic proof assistant

It allows mathematical formulas to be expressed in a formal language and provides tools for proving those formulas in a logical calculus

Isabelle is developed by Larry Paulson, Tobias Nipkow, Makarius Wenzel, ...

`http://isabelle.in.tum.de/`

Isabelle has millions of lines of code but there are also useful tiny provers

leanCoP: compact automated theorem prover for classical first-order logic

`http://www.leancop.de/`

Smallest ECLiPSe program is just 199 bytes

Isabelle installed with classical higher-order logic is 1.371.569.324 bytes

Alternatives to Isabelle

Coq is a formal proof management system

It provides a formal language to write mathematical definitions, executable algorithms and theorems together with an environment for semi-interactive development of machine-checked proofs

The prover uses a special intuitionistic higher-order logic

`http://coq.inria.fr/`

Other systems: PVS, HOL4, HOL Light...

`http://pvs.csl.sri.com/`

`http://hol.sourceforge.net/`

`http://code.google.com/p/hol-light/`

The Drinker Paradox — Logic Formulation

The paradox was introduced by Smullyan in 1978 who referred to it as the “drinking principle” but nowadays it is usually called the drinker paradox:

> There is someone in the pub such that, if that individual is drinking, everyone in the pub is drinking

$$(\exists x . Dx \rightarrow (\forall x . Dx))$$

Either everyone in the pub is drinking or at least one individual in the pub is not drinking

The formula is clearly true in the first case

Consider the second case and choose a non-drinking individual — the formula is vacuously true due to the semantics of implication since the condition is false

Either way the formula is true

The Drinker Paradox — A Textbook Proof

The sequent calculus is one-sided and the comma corresponds to disjunction

$\vdash \neg DA, DB, \neg DB, (\forall x.Dx)$	Axiom
$\vdash \neg DA, DB, (DB \rightarrow (\forall x.Dx))$	Rule $\rightarrow$
$\vdash (\exists x.Dx \rightarrow (\forall x.Dx)), \neg DA, DB$	Rule $\exists$
$\vdash (\exists x.Dx \rightarrow (\forall x.Dx)), \neg DA, (\forall x.Dx)$	Rule $\forall$
$\vdash (\exists x.Dx \rightarrow (\forall x.Dx)), (DA \rightarrow (\forall x.Dx))$	Rule $\rightarrow$
$\vdash (\exists x.Dx \rightarrow (\forall x.Dx))$	Rule $\exists$

Axiom: $\vdash \Gamma, \phi, \neg\phi$

Rule $\rightarrow$: $\vdash \Gamma, \neg\phi, \psi$ yields $\vdash \Gamma, (\phi \rightarrow \psi)$

Rule $\exists$: $\vdash \Gamma, \phi(c)$ yields $\vdash \Gamma, (\exists x.\phi(x))$

Rule $\forall$: $\vdash \Gamma, \phi(c)$ yields $\vdash \Gamma, (\forall x.\phi(x))$ if c does not occur in Γ

Natural Deduction — Inductive Definition

Axiom: $a \in G \leadsto G \vdash a$

$\top$ means true and $\bot$ means false

$\top$Intro: $G \vdash \top$

$\bot$Elim: $G \vdash \bot \leadsto G \vdash a$

$\neg$Intro: $a \cup G \vdash \bot \leadsto G \vdash \neg a$

$\neg$Elim: $G \vdash \neg a \leadsto G \vdash a \leadsto G \vdash \bot$

$\wedge$Intro: $G \vdash a \leadsto G \vdash b \leadsto G \vdash a \wedge b$

$\wedge$Elim1: $G \vdash a \wedge b \leadsto G \vdash a$

$\wedge$Elim2: $G \vdash a \wedge b \leadsto G \vdash b$

$\vee$Intro1: $G \vdash a \leadsto G \vdash a \vee b$

$\vee$Intro2: $G \vdash b \leadsto G \vdash a \vee b$

$\vee$Elim: $G \vdash a \vee b \leadsto a \cup G \vdash c \leadsto b \cup G \vdash c \leadsto G \vdash c$

$\rightarrow$Intro: $a \cup G \vdash b \leadsto G \vdash a \rightarrow b$

$\rightarrow$Elim: $G \vdash a \rightarrow b \leadsto G \vdash a \leadsto G \vdash b$

Natural Deduction — Isabelle Meta-Logic

Only the intuitionistic propositional part shown since the quantifier rules use auxiliary functions which are a bit unusual (de Bruijn indices)

A rule “Classical” must be added as well

The drinker paradox using the natural deduction involves 15 steps

Soundness already shown in Isabelle — a rather simple proof

Completeness also shown in Isabelle — a rather complicated proof

> Stefan Berghofer: *First-Order Logic According to Fitting*
> Archive of Formal Proofs 2007
>
> Melvin Fitting: *First-Order Logic and Automated Theorem Proving*
> Second Edition Springer 1996

CakeML: A Verified Implementation of ML

CakeML is designed to be both easy to program in and easy to reason about formally in proof assistants for higher-order logic, in particular HOL4

A case study is to construct a verified CakeML version of the HOL Light prover

`https://cakeml.org/`

> The CakeML project (and the HOL kernel that is part of that project) is the most interesting verification project of the moment, AFAIC.
>
> Freek Wiedijk (P.C. 2014)

HOL with Definitions: Semantics, Soundness, and a Verified Implementation
Springer LNCS — Conference on Interactive Theorem (ITP 2014)

But be aware of Gödel's Incompleteness Theorems...

Gödel's Incompleteness Theorems 1931

Back in 1986 Shankar made a computer-checked proof of the first theorem

1. Every "strong enough" formal system is *incomplete*, in that at least one formula can neither be proved nor disproved
2. And if such a formal system admits a proof of its own consistency, then it is actually *inconsistent*

And in 2013 Paulson made a computer-checked proof of both theorems using Isabelle (just under 16.000 lines in formal proof document)

- Some highly compressed proofs were finally made explicit
- The entire proof development can be examined interactively
- Today's provers can cope with very large formal proof documents

Ongoing Work

How to teach computer science students and professionals formal logic...?

Use Isabelle and its formalizations of soundness and completeness...!

From first-order logic to higher-order logic eventually... :-)

But even for natural deduction there are many variants:

Handbook of the History of Logic, Volume 11, Chapter 11, Pelletier & Hazen:
A History of Natural Deduction

50 elementary logic textbooks are compared

Perhaps less focus on axiomatics, resolution, tableaux and sequent calculus

NaDeA: Natural Deduction Assistant

A tool for teaching logic with a formalization in Isabelle

`http://nadea.compute.dtu.dk/`

NaDeA runs in a standard browser and is open source software — please find the source code and further information here:

`http://logic-tools.github.io/`