

Checking proofs

Reinhard Kahle¹ & Jesse Alama¹

(1) Center for Artificial Intelligence
New University of Lisbon

`kahle@mat.uc.pt`, `j.alama@fct.unl.pt`

9 September 2011, Ponta Delgada
Philosophy of Computer Science and Artificial Intelligence

- Puzzle: how can one check a proof?
- An epistemological shift in recent philosophy of mathematics.
- Computer-assisted theorem proving.

A puzzle about proofs

- Suppose that we have a proof d of a mathematical theorem.
- Intuitively, d is a proof independent of our knowledge that it is a proof.
- What does it mean to *check* d ?
- If d really is a proof, then checking d can give new knowledge.
- If d really is not a proof, then checking d might give new knowledge (the argument purported to be a proof is not in fact a proof).

- The puzzle is reminiscent of a puzzle about the analysis of knowledge:

If A knows that p , then p is true, so A cannot be wrong about p .

- Thus at first blush, without any further analysis, the concept of knowledge apparently amounts to certainty, and concepts such as *fallible knowledge* come out as incoherent.
- Likewise, intuitively a mathematical proof is fallible, too. But if could be wrong, is it really a proof?

- There are various escape hatches for avoiding the collapse of knowledge and certainty:
 - Bite the bullet and accept the collapse. Or jettison the concept of proof entirely.
 - Make distinctions: e.g., internalism vs. externalism about various epistemic or rhetorical phenomena.
 - Revise the analysis of knowledge (more conditions?)
 - Refine components of the analysis, e.g., justification.

Digging in to justification

- Viewing proofs as warrants of a certain kind.
- Doing so gives us some conceptual breathing room:
- We can ask how mathematical proofs, qua warrants, serve to justify.
- We can distinguish between different kinds of proofs, qua warrants.

Epistemological shift in philosophy of mathematics

- Increased interest lately (last 40 years or so) in the phenomenon/practice of mathematical proof.
- In some quarters, there is a shift from traditional metaphysical question about the stuff of mathematical assertions (numbers, functions, spaces) toward the practice of demonstrating things about these mathematical objects, whatever they are.
- With the intention of understanding the epistemology of mathematics, we turn towards its practice.
- Allied subjects: philosophy of science, argumentation theory, informal logic.

- View proofs as *structured warrants*.
- What structure do they have? Can we give a formal account of the structure of arguments? Can that account give us a way to make sense of the way in which proofs can succeed or fail, as warrants?
- Many proposals and resources are available, coming from mathematical logic, linguistics, and computer science.
 - Proof theory: Hilbert-style deductions, Gentzen-style deductions, natural deduction, . . .
 - Computer science: programming languages
 - Linguistics: discourse representation theory

As a valuable initial position on understanding mathematical proofs, let's take the approach taken by I. Lakatos:

From *Proofs and Refutations*

I propose to retain the time-honoured technical term 'proof' for a *thought-experiment*—or '*quasi-experiment*'—*which suggests a decomposition of the original conjecture into subconjectures or lemmas, thus embedding it in a possibly quite distant body of knowledge.*

- Lakatos's conception of mathematical proof is intuitively attractive because it gives some room for the fallibility of proof. Following Lakatos, proofs even become a kind of conjecture.
- Nonetheless, Lakatos's conception does not relate mathematical proof with desired notions such as logical consequence, validity, and truth given to us by mathematical logic and philosophy.
- A further step is needed. Can we enrich Lakatos's conception, taking seriously the "conjectural" nature of mathematical proof while relating it to standard models of logical consequence?

- “Theorem proving” is a broad subject in computer science aiming to give mechanical assistance in the service of reasoning.
- A basic distinction in the field:
 - *Automated* theorem proving (ATP) covers systems whose primary use is the unassisted, automated exploration of a reasoning problem.
 - *Interactive* theorem proving (ITP) covers systems whose primary use is the exchange of a human and some kind of mechanical reasoning assistant.

- A number of ITPs are thriving. Some of the major ones:
 - **Coq**
 - **Mizar**
 - **Isabelle**
 - **HOL4/HOL light**
- These systems take varying approaches toward representing mathematical arguments:
 - Their underlying logics and motivations differ (some weak, some strong);
 - The formats in which different proofs are written differs (declarative vs. procedural proofs);

- Each of these systems comes with its own library of formalized mathematical knowledge.
- Some libraries are quite large. **Mizar**'s formal library contains, for example:
 - More than 2 million lines of proofs
 - > 10k definitions
 - > 50k theorems
 - > 1100 “articles” analogous to an article in a mathematics journal: a collection of definitions, theorems, and proofs unified around a problem/target theorem(s).

- These proof systems are, to a large extent, quite successful:
- They each give their own analysis of mathematical language.
- “The proof of the pudding is in the eating”: these systems are able to express a wide variety of mathematical proofs.
- We can talk to each other all day long about “being formal”. With these systems you can *actually do that*.
- Moreover, one can actually go places. These are far from toy systems. Many major mathematical theorems have been formalized in complete detail, from beginning to end. The libraries and languages continue to evolve as we gain experience with the everyday practice of what is involved in practically formalizing arguments.





- To be sure, there are some important limitations on the enterprise of understanding mathematical proof by looking at ITPs:
- The systems are of course predisposed to capture only discursive elements of mathematical proof. Non-discursive aspects of proofs, such as reasoning by pictures or physical intuition, is not treated by these systems.
- Theoretical barriers: at one extreme of interactive theorem proving is the practice of using an ITP as an oracle for logical consequence: “tell me whether φ logically follows from Γ ”. For large classes of logics, testing such assertions is undecidable.
- Verifying verifications. Is there no end?

Checking a proof with an ITP

Let's see how this works!

- The process of formalization can reveal:
 - Gaps in the informal argument being formalized.
 - Problems of embedding the informal argument into a formal framework.
 - Gaps in our understanding of what we're actually proving.
- Checking that certain inferences are “obvious” is a problem of automated theorem proving. The power that goes into checking obviousness of an inference is rather modest, but there can be advantages to keeping the proof checker weak.
 - Proof of Cantor's diagonal argument in **Isabelle**: extremely little justification needed (“just do depth-first search”). Is this satisfactory?

- Formal mathematics is a fascinating brew of mathematical logic, computer science, philosophy of mathematics, and artificial intelligence.
- By picking a particular formal framework and checking proofs within it, one can get a concrete picture of what mathematical argumentation that we might not have seen had we worked entirely “informally”.
- Formal proofs offer a kind of conceptual laboratory for experimenting with how formal and informal argumentation converge/diverge, what counts as “obvious”, ...

-  Rudnicki, Piotr, 1987: “Obvious inferences”, *Journal of Automated Reasoning* **3**(4), pp. 383–393.
-  The **Mizar** homepage (<http://www.mizar.org>),
-  Lakatos, Imre, 1976: *Proofs and Refutations*, Cambridge University Press.
-  Wiedijk, F. (2010): “Formal proof—getting started”, *Notices of the American Mathematical Society* **55**(11), pp. 1408–1414.