CDM

Proofs

Klaus Sutner

Carnegie Mellon University

50-proofs 2017/12/15 23:21

1 What is a Proof?

- Formal Proofs
- Provers

Proofs are Old

Mathematical proofs in nearly modern form have been around for a long time, at least since Euclid was active some 2300 years ago.

His aximatization of geometry is brilliant, and it took till 1899 for Hilbert to write the definitive text on the subject (Grundlagen der Geometrie).

Euclid's proof of the infinitude of primes holds up very well (at least if you are willing to accept classical logic).

For the sake of a contradiction, assume there are only finitely many primes $p_1, p_2, \ldots, p_n$. Consider the number

$$q = p_1 \cdot p_2 \cdot \ldots \cdot p_n + 1.$$

If q is prime, it is different from all the p_i. Otherwise, q must have a prime factor p. But p cannot be any one of the p_i. In either case, there is a contradiction.

A “Proof” by Euler

Euler had incredibly strong intuition, which gave him the amazing ability to concoct arguments that were eminently plausible, and led to correct results, but were exceedingly difficult to justify in the modern sense (polite speak for: wrong). Here is an example:

Problem: Find a way to calculate e^x for positive reals x.

We know that for $x > 0$ reasonably small we can write

$$e^x = 1 + x + error$$

with the error term being small. Alas, we have no idea what exactly the error is.

Infinitesimals

Euler considers an infinitesimal $\delta > 0$. Then

$$e^{\delta} = 1 + \delta$$

This is justified by Leibniz's *lex homogeneorum transcendentalis*, the transcendental law of homogeneity (proof by higher authority).

Then, by the laws of exponentiation,

$$e^{x} = (e^{\delta})^{x/\delta}$$

Now comes the interesting part ...

Binomial Theorem

$$\begin{aligned} e^x &= (e^\delta)^{x/\delta} \\ &= (1+\delta)^{x/\delta} \\ &= 1 + \binom{x/\delta}{1}\delta + \binom{x/\delta}{2}\delta^2 + \dots \\ &= 1 + x + 1/2\, x(x-\delta) + \dots \\ &= 1 + x + 1/2\, x^2 + \dots \\ &= \sum_{i \ge 0} x^i/i! \end{aligned}$$

Headache?

The result is perfectly correct. But the argument . . . oy vey.

Incidentally, Euler also had an ingenious argument to show that

$$1 + 2 + 3 + \ldots + n + \ldots = -\frac{1}{12}$$

No, that's not nuts.

Look up ζ functions and analytic continuations.

Food for Thought

Exercise

Find all the places where Euler's reasoning is dubious from a modern day perspective.

Exercise (Very Hard)

Fix all the problems with Euler's argument.

Exercise

Figure out why it makes sense to claim that $1 + 2 + 3 + \ldots = -1/12$*. Similarly* $1 + 1 + 1 + \ldots = -1/2$*.*

Proof Defined

Borrowing from E. Bishop, let us say:

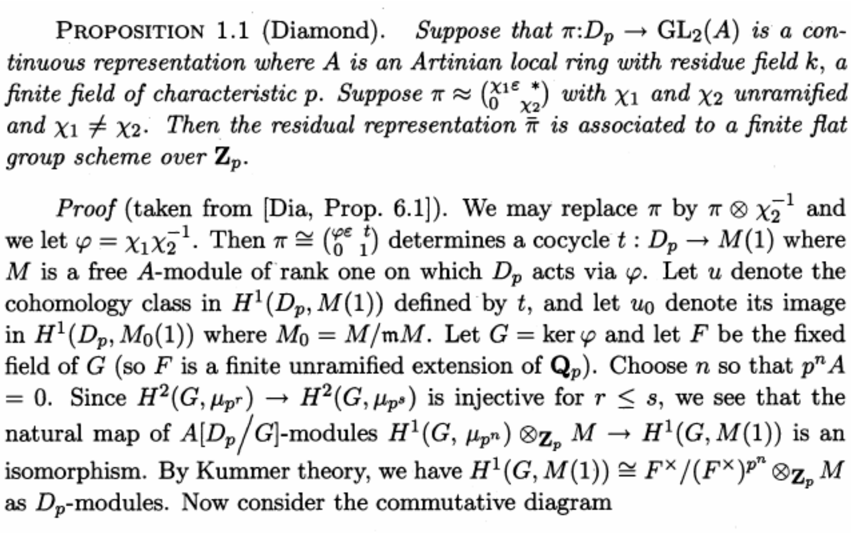
A proof is any completely compelling argument.

Compelling means totally lucid and leaving no room whatsoever for doubt or prevarication.

Of course, the argument is compelling only once it is completely understood. To develop this understanding may take years of training and many hours of concentrated effort.

Published proofs are almost always directed at trained mathematicians, and incomprehensible to everyone else.

Impenetrable



PROPOSITION 1.1 (Diamond). *Suppose that $\pi : D_p \to \mathrm{GL}_2(A)$ is a continuous representation where A is an Artinian local ring with residue field k, a finite field of characteristic p. Suppose $\pi \approx \left(\begin{smallmatrix} \chi_1\varepsilon & * \\ 0 & \chi_2 \end{smallmatrix}\right)$ with χ_1 and χ_2 unramified and $\chi_1 \neq \chi_2$. Then the residual representation $\bar{\pi}$ is associated to a finite flat group scheme over $\mathbf{Z}_p$.*

Proof (taken from [Dia, Prop. 6.1]). We may replace π by $\pi \otimes \chi_2^{-1}$ and we let $\varphi = \chi_1\chi_2^{-1}$. Then $\pi \cong \left(\begin{smallmatrix} \varphi\varepsilon & t \\ 0 & 1 \end{smallmatrix}\right)$ determines a cocycle $t : D_p \to M(1)$ where M is a free A-module of rank one on which D_p acts via φ. Let u denote the cohomology class in $H^1(D_p, M(1))$ defined by t, and let u_0 denote its image in $H^1(D_p, M_0(1))$ where $M_0 = M/\mathfrak{m}M$. Let $G = \ker\varphi$ and let F be the fixed field of G (so F is a finite unramified extension of $\mathbf{Q}_p$). Choose n so that $p^nA = 0$. Since $H^2(G, \mu_{p^r}) \to H^2(G, \mu_{p^s})$ is injective for $r \leq s$, we see that the natural map of $A[D_p/G]$-modules $H^1(G, \mu_{p^n}) \otimes_{\mathbf{Z}_p} M \to H^1(G, M(1))$ is an isomorphism. By Kummer theory, we have $H^1(G, M(1)) \cong F^\times/(F^\times)^{p^n} \otimes_{\mathbf{Z}_p} M$ as D_p-modules. Now consider the commutative diagram

Wishy-Washy

Our compelling argument "definition" is all about human cognitive abilities, and even social interactions–after all, general agreement of what type of argument meets these criteria has shifted over time (and probably will continue to do so).

What does any of this have to do with mathematics?

A lot. We will give a formal definition of proof in a while, but that is no more than an attempt to import the idea expressed in our informal definition into a strictly mathematical setting.

Informally, always think about a compelling argument. Test it on your peers to see if it holds up.

Purpose?

A (good) proof has two clear and sometimes distinct purposes:

Certification	It establishes the truth of an assertion beyond reasonable doubt.
Explanation	It helps to understand why the assertion is true.

The explanation part is particularly tricky, a great many published proofs fail miserably in this sense–though they do provide the requisite certification.

Certification?

One might think that a published result by a strong mathematician is almost always correct, but things are bit more complicated.

> If Gauss says he has proved something, it seems very probable to me; if Cauchy says so, it is about as likely as not; if Dirichlet says so, it is certain.
>
> Carl Gustav Jacob Jacobi

For example, seemingly intuitive notions in analysis such as continuity and differentiability are much more subtle than they might seem. It is very, very easy to trip up–ask Cauchy.

Bieberbach Conjecture

Any analytic function on the unit disk

$$f(z) = z + \sum_{k \geq 2} a_k z^k$$

that is injective (schlicht function) must have $|a_k| \leq k$.

This was conjectured in 1931, and proven by Louis de Branges in 1985.

But there were problems.

A Developing Proof

First the proof was contained in a book of 350 pages that no one read.

Then de Branges spent a semester in St. Petersburg and worked with a group there to get the argument down to 30 pages.

The manuscript was circulated, and XXXXXXXXXXX found a way to simplify the argument.

Because of a editorial snafu, the XXXXXXXXXXX paper was published before the main paper. De Branges was furious.

Then XXXXXXXXXXX found a 4 page proof.

Zeilberger gave a 1/2 page proof using a computer.

The Book of Proofs

See for yourself.

Martin Aigner, Günter Ziegler
Proofs from THE BOOK
Springer 2010 (4th ed.)

Then pick up a random math journal and compare.

Proofs and Refutations

Along similar lines, there is a mildly amusing booklet that discusses assorted attempts at proving Euler's famous formula $V - E + F = 2$.

Imre Lakatos
Proofs and Refutations
CUP, 1976

Lakatos describes a long and surprisingly tedious process of discovery, presented as a dialogue between a teacher and a group of students. Complete with lots of false leads and dead ends.

- What is a Proof?

2. Formal Proofs

- Provers

Formalization

How would we translate our informal notion of proof into a rigorous mathematical concept?

Sadly, we will have to give up on the explanatory part and focus just on certification of truth.

The main idea is simple: first, we need to work within a carefully defined logic. Second, the whole proof is broken up into a (potentially very long) sequence of single reasoning steps, each one which conforms to some precise rules of inference. Moreover, at each step, only general axioms, specific proof hypotheses or already established results that appear earlier in the sequence can be used.

The Method

In order to design a logic we need to take care of three things:

- Syntax: we have to design a formal language of terms and formulae.
- Semantics: we have to explain the meaning of the formulae; in particular we have to define what truth is.
- Deduction: we have to explain how a proof works in our system and in particular what rules of inference are admissible.

Of course, these three aspects are closely connected, but it is helpful to think of them as separate tasks.

Syntax

This is the easy part. The languages we are dealing with are much simpler than, say, programming languages and it is fairly straightforward to explain their structure.

In fact, it is straightforward to construct parsers that read and analyze a formula in any of our languages.

Needless to say, things can become more complicated when we are interested in implementing sophisticated algorithms. For example, checking satisfiability of a formula in propositional logic requires some thought about data structures.

Semantics

The difficulty of explaining semantics varies greatly with the logic in question, but in general this is a complicated undertaking and requires much more effort than the pure syntax.

In particular for predicate logic and temporal logics the definition of truth will be quite involved.

Also note that, in general, we will be unable to construct algorithms that check for truth. Sadly, even when algorithms exist they may well be unfeasible.

Deduction

To express the notion of a “formal proof” in one of our logics we have to define rules of inference that can be used to deduce new statements from given ones.

For example, if our logic includes implication (if-then) we might allow the rule of modus ponens: if we have a proof of A and a proof of $A \Rightarrow B$, then we also have a proof of B.

It may seem that these rules should be fairly natural and straightforward, but there are several difficulties that one needs to address. As it turns out, in particular in the context of computer science, standard classical logic is not always the best choice.

Gödel Style Proofs

Suppose Γ is a set of assumptions. In a nutshell, a Gödel style proof from Γ in a logic is a sequence of formulae, each of which is either

- an axiom or in Γ,

or

- justified via a rule of inference applied to formulae earlier in the sequence.

Axioms are arbitrary in a sense, though in real applications they are chosen very carefully and encapsulate the salient features of the domain of discourse.

It is fairly clear that one needs to be explicit about the axioms, but it took some effort late in the 19th century to convince mathematicians that the rules of inference also need to be carefully elaborated.

Formal Proofs

Suppose we have fixed a formal system. Consider a set of formulae Γ (possibly empty) that expresses a set of assumptions.

Definition

A proof or derivation of a formula ψ from Γ is a sequence

$$\varphi_1, \varphi_2, \varphi_2, \ldots, \varphi_{n-1}, \varphi_n = \psi$$

of formulae such that

- φ_i is an axiom of the system, or
- φ_i is a formula in Γ, or
- φ_i follows from some of the φ_j, $j < i$, by some rule of inference (such as modus ponens).

φ is said be provable from Γ (in the system), written $\Gamma \vdash \varphi$.

Quality Control

How do we know that a particular logic is properly constructed? There are two basic properties one would like to have:

Soundness	The system derives only true formulae.
Completeness	The system derives all true formulae.

Soundness is really non-negotiable: we cannot allow false statements to creep in. In fact, typically one false statement like $0 = 1$ allows one to prove absolutely anything in the system.

On the other hand, completeness may be a bit of a luxury; we may be better off with less than full completeness.

Quoi?

The last comment may seem surprising, if not insane: why would be content with an incomplete system that misses some of the true statements we are interested in?

The answer is algorithms: ultimately, we would like to write programs that are capable of computing in a logic. In particular we would like to be able to find derivations automatically, or at the least verify a given alleged derivation.

To this end it may be desirable to keep things simple and give up on completeness, in exchange for better algorithms – algorithms that presumably would work well on the actual inputs that we are interested in. More later.

Disclaimer

In the following we will discuss a simple example, a so-called Hilbert style system for propositional logic.

There is really only one justification for this: these systems are extremely easy to describe, and the axioms and rules of inference are very straightforward.

The drawback of this simplicity is huge and actually debilitating: no one uses these systems in the real world (actual proofs in mathematics, theorem provers/proof assistants, formal verification). There are *much* better systems for real applications. Alas, they are significantly more complicated, so we start with something really simple.

A Hilbert Style System

For simplicity, the following system $\mathcal{H}$ uses only two connectives: $\neg$ and $\vee$ (the others could be defined from these).

Logical Axioms:

tertium non datur	$\neg\varphi \vee \varphi$

Rules of Inference:

expansion	$\dfrac{\varphi}{\psi \vee \varphi}$
contraction	$\dfrac{\varphi \vee \varphi}{\varphi}$
associativity	$\dfrac{\varphi \vee (\psi \vee \chi)}{(\varphi \vee \psi) \vee \chi}$
cut rule	$\dfrac{\varphi \vee \psi \qquad \neg\varphi \vee \chi}{\psi \vee \chi}$

Sample Derivation

Claim (modus ponens)

$p, \neg p \lor q \quad \vdash \quad q$

Here is a derivation, annotated by proof rules:

1	p	$prem$
2	$q \lor p$	$e, 1$
3	$\neg q \lor q$	$axiom$
4	$p \lor q$	$cut, 2, 3$
5	$\neg p \lor q$	$prem$
6	$q \lor q$	$cut, 4, 5$
7	q	$c, 6$

Note that this annotation makes it trivial to check correctness.

In Tree Form

Alas, linear proofs are hard to read, even with annotations. For humans, a corresponding proof tree is often much easier to deal with.

$$\dfrac{\dfrac{\dfrac{\dfrac{p}{q \lor p}\ (e) \qquad \neg q \lor q}{p \lor q}\ (cut) \qquad \neg p \lor q}{q \lor q}\ (cut)}{q}\ (c)$$

The root is the proven formula, and the hypotheses and axioms are at the leaves. The labels indicate the type of rule used and there is no need for back-references.

Exercise

Explain the connection between the tree and the linear form of the proof.

Local/Global Problem

A formal proof like this is blindingly boring: it is comprised of a long, long sequence of tiny little steps that all tend to be trivial individually – though the whole proof may well be incomprehensible.

But note the good news: the steps are so simple that they can easily be verified by a computer (proof checker).

In fact, checking an alleged proof for accuracy essentially comes down to a bit of word processing: all we need to do is pattern matching.

Soundness

Soundness is also easy: we have to make sure that only tautologies can be derived (without premisses).

Theorem (Soundness Theorem)

System $\mathcal{H}$ is sound: only tautologies can be derived in it.

Soundness is usually easy to see for any Hilbert system by induction:

- The axioms are tautologies, and
- application of any proof rule to given tautologies produces another tautology.

Completeness

But completeness is more complicated: we need to show that the system produces proofs for all tautologies.

Theorem (Completeness Theorem)

System $\mathcal{H}$ is complete: all tautologies can be derived in it.

Given the fact that our example system has only one kind of axiom and relatively simple rules of inference it is by no means clear that we do not miss out on some tautologies.

What if the formula has thousands of variables and the parse tree is hundreds of levels deep?

We give a sketch of the completeness proof, make sure you are able to supply all the necessary details.

Sketch

Lemma

Let $1 \leq i_1, i_2, \ldots, i_m \leq n$*.*

$$\varphi_{i_1} \lor \varphi_{i_2} \lor \ldots \lor \varphi_{i_m} \vdash \varphi_1 \lor \varphi_2 \lor \ldots \lor \varphi_n$$

Proof. Ugly induction on m. □

Note that our notation in the lemma is actually not quite right: $\lor$ is a binary operation and we must use parens to obtain a syntactically correct formula.

But, because of the associativity rule, the parens do not matter as far as derivations are concerned–but this fact takes work to check.

Sketch II

To lighten notation, write $\psi \Rightarrow \varphi$ for $\neg\psi \vee \varphi$.

Theorem (Deduction Theorem)

$\Gamma, \psi \vdash \varphi$ *implies* $\Gamma \vdash \psi \Rightarrow \varphi$.

Proof.

Induction on the length of the derivation of φ from Φ, ψ.

$\square$

This is really a sanity check: in our system implications properly reflect derivability.

Sketch III

For the completeness proof it suffices to show that any tautology of the form

$$\varphi = \varphi_1 \vee \varphi_2 \vee \ldots \vee \varphi_n$$

where $n \geq 1$ is provable in the system. This is done by induction on the total number of connectives in φ.

Note: not induction on n, there is no hope for that.

Exercise

Give a detailed proof of the deduction theorem.

Exercise

Carry out the details of the soundness and completeness proof.

- What is a Proof?

- Formal Proofs

3. Provers

Reasonable Logic

As already mentioned, we should augment our core requirements for a reasonable logic to include decidability.

- Soundness: only valid formulae can be derived.
- Completeness: all valid formulae can be derived.
- Effectiveness: proofs (as combinatorial objects) are decidable.

It is clear that effectiveness holds in $\mathcal{H}$ (and all similar systems): pattern matching is all that is needed to check whether a sequence of formulae is a proof.

Provers versus Checkers

Programs that determine whether an alleged proof is actually correct are called proof checkers: they don't construct proofs, they just make sure that all the rules were properly obeyed.

Checkers can be quite interesting when used in conjunction with provers, programs that actually try to generate proofs: if we do not entirely trust the prover, we use a checker (potentially written by someone else) to make sure the results are indeed correct.

This is a nail in the coffin of claims that machine-generate proofs are somehow less reliable than old-fashioned hand-written ones. Quite the contrary.

Proof Assistants

Proof search is exceedingly difficult, so it makes sense to build “weak provers” that expect interactive input from a human being in the construction of a proof.

In essence, the human has to sketch the general structure of the proof and provide hints whenever the machine gets stuck. With a bit of luck, the assistant can fill in all the gaps and construct a complete formal proof from the sketch.

At present, assistants are the most plausible tools in this area; fully automatic theorem provers tend to be too limited in scope.

Warning: Long Proofs

We are not making any claims about the decidability of derivability:

$$\Gamma \vdash \varphi$$

is usually a much more complicated relation.

In fact, in most interesting cases derivability will be complete semidecidable. The problem here is that it is exceedingly difficult to determine bounds on the length of a potential proof in terms of the data Γ and φ.

The Real World

Proof length is also a huge problem when dealing with actual theorems, rather than little toy examples: interesting proof tend to be very, very long when formalized.

It is often claimed in mathematics that any published proof could be translated into a formal proof, using some proof system like classical first-order logic together with Zermelo-Fraenkel set theory. Alas, it is not so clear what is actually meant when someone says “in principle we could rewrite this as a formal proof”.

See `http://mizar.org` for a major effort to reconstruct large parts of mathematics in a formal environment:

> . . . a database which includes more than 9400 definitions of mathematical concepts and more than 49000 theorems.