Mathematical Definitions, Formally Speaking

Thomas Hales

May 27, 2019

In memory of Vladimir Voevodsky and Mike Gordon

Formal Abstracts in Mathematics

FAB
STRACTS

ALFRED P. SLOAN FOUNDATION
1934
Logo of the Alfred P. Sloan Foundation,
Alfred P. Sloan
FOUNDATION

UNIVERSITY OF PITTSBURGH
VERITAS 1787 VIRTUS

Carnegie
Mellon
University

THANG LONG
UNIVERSITY

VIỆN TOÁN HỌC
Hanoi

Sphere Packings

Forum of Mathematics, Pi

Search F

Article Metrics

Volume 5 2017 , e2

A FORMAL PROOF OF THE KEPLER CONJECTURE

THOMAS HALES [a1], MARK ADAMS [a2] [a3], GERTRUD BAUER [a4], TAT DAT DANG [a5] ...

https://doi.org/10.1017/fmp.2017.1 Published online: 29 May 2017

Abstract

This article describes a formal proof of the Kepler conjecture on dense sphere packings in a combination of the HOL Light and Isabelle proof assistants. This paper constitutes the official published account of the now completed Flyspeck project.

THOMAS HALES [a1], MARK ADAMS [a2] [a3], GERTRUD BAUER [a4], TAT DAT DANG [a5], JOHN HARRISON [a6], LE TRUONG HOANG [a7], CEZARY KALISZYK [a8], VICTOR MAGRON [a9], SEAN MCLAUGHLIN [a10], TAT THANG NGUYEN [a7], QUANG TRUONG NGUYEN [a1], TOBIAS NIPKOW [a11], STEVEN OBUA [a12], JOSEPH PLESO [a13], JASON RUTE [a14], ALEXEY SOLOVYEV [a15], THI HOAI AN TA [a7], NAM TRUNG TRAN [a7], THI DIEP TRIEU [a16], JOSEF URBAN [a17], KY VU [a18] and ROLAND ZUMKELLER [a19]

The formal proof of the Kepler conjecture, which was finally published in 2017 uncovered and corrected hundreds of errors in the proof.

where `the_kepler_conjecture` is defined as the following term

```
`(!V. packing V
          ==> (?c. !r. &1 <= r
                     ==> &(CARD(V INTER ball(vec 0,r))) <=
                         pi * r pow 3 / sqrt(&18) + c * r pow 2))`
```

In standard mathematical language, this states that for every packing V (which is identified with the set of centers of balls of radius 1), there exists a constant c controlling the error term, such that for every radius r that is at least 1, the number of ball centers inside a spherical container of radius r is at most `pi * r^3 / sqrt(18)`, plus an error term of smaller order. As r tends to infinity, this gives the density bound `pi / sqrt(18) = 0.74+`, which is the density of the face-centered-cubic packing.

The term `the_nonlinear_inequalities` is defined as a conjunction of several hundred nonlinear inequalities. The domains of these inequalities have been partitioned to create more than 23,000 inequalities. The verification of all nonlinear inequalities in HOL Light on the Microsoft Azure cloud took approximately 5000 processor-hours. Almost all verifications were made in parallel with 32 cores, hence the real time is about 5000 / 32 = 156.25 hours. Nonlinear inequalities were verified with compiled versions of HOL Light and the verification tool developed in Solovyev's 2012 thesis.

To check that no pieces were overlooked in the distribution of inequalities to various cores, the pieces have been reassembled in a specially modified version of HOL Light that allows the import of theorems from other sessions of HOL light. In that version, we obtain a formal proof of the theorem

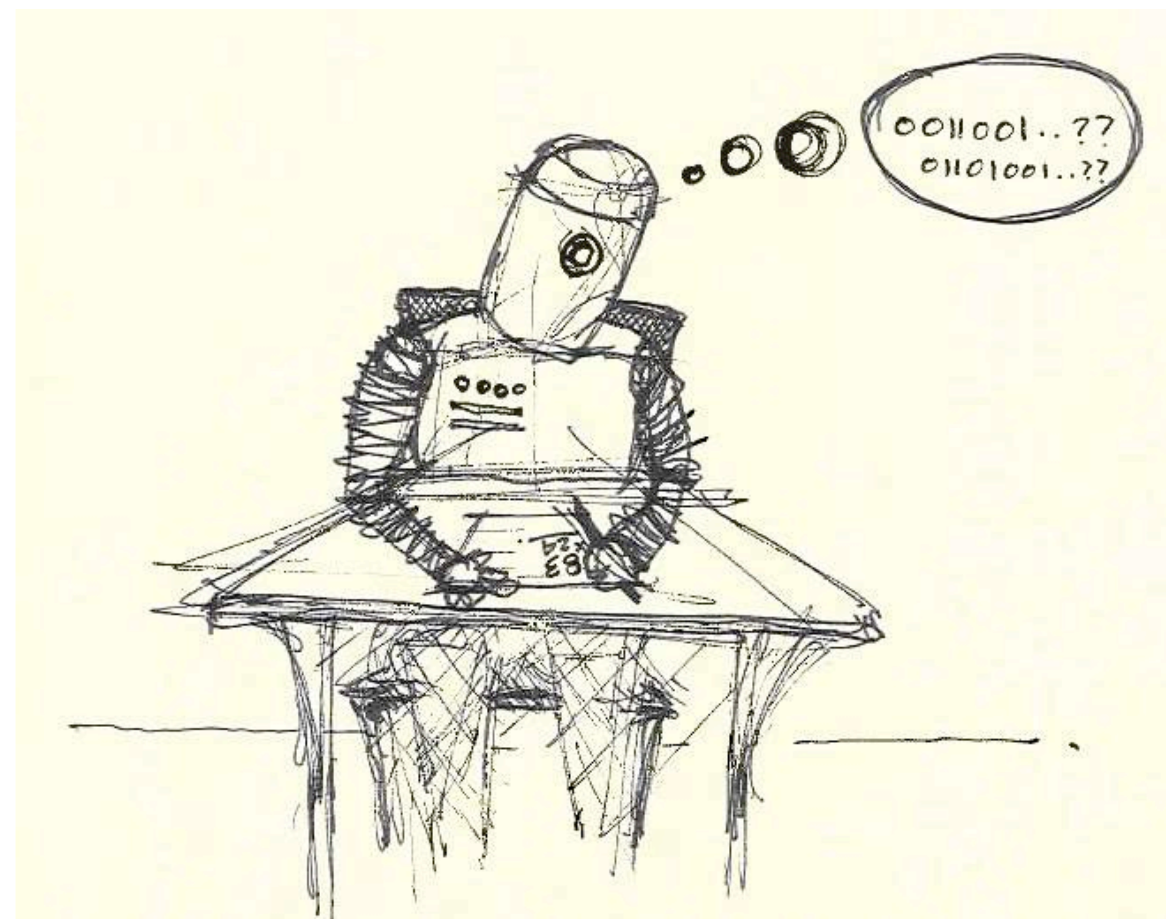
```
|- the_nonlinear_inequalities
```

Big Proof and Formal Proof

- Size and materials

Computers were once human

Referees were once human



Kaliszyk-Chollet-Szegedy

Published as a conference paper at ICLR 2017

HolStep: A Machine Learning Dataset for Higher-order Logic Theorem Proving

Cezary Kaliszyk
University of Innsbruck
cezary.kaliszyk@uibk.ac.at

François Chollet, Christian Szegedy
Google Research
{fchollet,szegedy}@google.com

Abstract

Large computer-understandable proofs consist of millions of intermediate logical steps. The vast majority of such steps originate from manually selected and manually guided heuristics applied to intermediate goals. So far, machine learning has generally not been used to filter or generate these steps. In this paper, we introduce a new dataset based on Higher-Order Logic (HOL) proofs, for the purpose of developing new machine learning-based theorem-proving strategies. We make this dataset publicly available under the BSD license. We propose various machine learning tasks that can be performed on this dataset, and discuss their significance for theorem proving. We also benchmark a set of simple baseline machine learning models suited for the tasks (including logistic regression, convolutional neural networks and recurrent neural networks). The results of our baseline models show the promise of applying machine learning to HOL theorem proving.

Kaliszyk-Chollet-Szegedy

1.1 Contribution and Overview

First, we develop a dataset for machine learning based on the proof steps used in a large interactive proof section 2. We focus on the HOL Light (Harrison, 2009) ITP, its multivariate analysis library (Harrison, 2013), as well as the formal proof of the Kepler conjecture (Hales et al., 2010). These formalizations constitute a diverse proof dataset containing basic mathematics, analysis, trigonometry, as well as reasoning about data structures such as graphs. Furthermore these formal proof developments have been used as benchmarks for automated reasoning techniques (Kaliszyk & Urban, 2014).

The dataset consists of 2,013,046 training examples and 196,030 testing examples that originate from 11,400 proofs. Precisely half of the examples are statements that were useful in the currently proven conjectures and half are steps that have been derived either manually or as part of the automated proof search but were not necessary in the final proofs. The dataset contains only proofs of non-trivial theorems, that also do not focus on computation but rather on actual theorem proving. For each proof, the conjecture that is being proven as well as its dependencies (axioms) and may be exploited in machine learning tasks. Furthermore, for each statement both its human-readable (pretty-printed) statement and a tokenization designed to make machine learning tasks more manageable are included.

The relationship between the computer and mathematics is decisively different from the relationship between the computer and the empirical sciences. The essential difference is that mathematics is capable of exact representation by computer, but the external world only admits approximate representation by computer. This difference has enormous implications for the correct architecture of mathematical databases. A database of formal math abstracts can capture true mathematical content in a way that say a database of chemical compounds never will.

A concrete proposal: mathematical FABSTRACTS (formal abstracts)

Given today's technology, it is not reasonable to ask for all proofs to be formalized. But with today's technology, it seems that it should be possible to create a formal abstract service that

- Gives a statement of the main theorem(s) of each published mathematical paper in a language that is both human and machine readable,
- Links each term in theorem statements to a precise definition of that term (again in human/machine readable form), and
- Grounds every statement and definition is the system in some foundational system for doing mathematics.

On Digital Math Libraries

We should not compromise rigorous mathematical standards as we move from paper to computer. In fact, this is an opportunity to drastically improve standards. Many computer bugs are simply slips in logical and mathematical reasoning made by programmers and software designers.

- Mathematics influences the standards of scientific discourse, in the statistical sciences, in computer science, and throughout the sciences. If we promote sloppy platforms, the entire world will be worse off.
- Bugs in computer systems can lead to disaster: Intel Pentium FDIV bug, Ariane V explosion, . . .
- Bugs and design weaknesses in cryptographic software can be exploited by adversaries: Heartbleed, Logjam, Freak bug, . . .

Why?

- bring the benefits of proof assistants to the general mathematical community;
- set standards for the sciences;
- set the stage for applications to ML in mathematical proofs;
- move math closer to the computer.

HOL Light

HOL Light has an exquisite minimal design. It has the smallest kernel of any system. John Harrison is the sole

Mizar

Once the clear front-runner, it now shows signs of age. Do not expect
to understand the inner workings of this system unless you have been

Coq

Coq is built of modular components on a foundation of dependent type theory. This system has grown one PhD thesis at a time.

Isabelle

Designed for use with multiple foundational architectures, Isabelle's early
development featured classical constructions in set theory. However,

Metamath

Does this really work? Defying expectations, Metamath seems to function
shockingly well for those who are happy to live without plumbing.

Lean

Lean is ambitious, and it will be massive. Do not be fooled by the name.
"*Construction area keep out*" signs are prominently posted on the perimeter fencing.

HOL Light

HOL Light has an exquisite minimal design. It has the smallest kernel of any system. John Harrison is the sole

Lean

Lean is ambitious, and it will be massive. Do not be fooled by the name. “*Construction area keep out*” signs are prominently posted on the perimeter fencing.

Lean Theorem Prover

- Lean has a small kernel.
- Its logical foundations are similar to those of Coq.
- Lean is its own metalanguage.

This example illustrates how Lean is both a programming language and a theorem prover, allowing formal mathematics and its metadata to be combined seamlessly into a single document. We stress that the mathematics is machine readable by a computer proof assistant. We display the formal abstract in its raw (computer) form, but we anticipate that viewing tools will convert this raw format into English text, Mathematica notebook data, user friendly web browser display, MathSciNet data, and so forth:

```
-- the statement of Fermat's Last Theorem
axiom fermats_last_theorem :
∀ (x y z n : ℕ), x > 0 → y > 0 → n > 2 → x ^ n + y ^ n ≠ z ^ n

def paper : document := {
authors := [ {name := "Andrew Wiles"} ],
title := "Modular elliptic curves and Fermat's last theorem",
doi := "10.2307/2118559"
}

definition fabstract : fabstract := {
description := "This theorem bearing Fermat's name
was stated without proof by Pierre de Fermat in 1637
in the margins of his copy of Diophantus' Arithmetica.
Andrew Wiles announced a proof in 1994,
and his corrected proof was published in 1995."
sources := [cite.Document paper],
results := [result.Proof fermats_last_theorem]
}
```

Here is a fragment of the formal abstract for the statement of the Riemann hypothesis. The full formal abstract will include links to each of the definitions (such as the specification of the field of complex numbers):

```
def holomorphic_on (domain : set ℂ) (f : subtype domain → ℂ) :=
(∀ z : subtype domain, ∃ f'z,
has_complex_derivative_at (extend_by_zero domain f) f'z z)

class holomorphic_function :=
(domain : set ℂ)
(f : subtype domain → ℂ)
(open_domain : is_open domain)
(has_derivative : holomorphic_on domain f)

-- notation f(z), for holomorphic functions
instance : has_coe_to_fun holomorphic_function :=
{ F := λ h, subtype h.domain → ℂ, coe := λ h, h.f }

-- converges for Re(s) > 1
def riemann_zeta_sum (s : ℂ) : ℂ :=
Σ  (λ n,  complex.pow n  (-s) )

-- trivial zeros at -2, -4, -6,...
def riemann_zeta_trivial_zero (s : ℂ) : Prop :=
(∃ n : ℕ, n > 0 ∧ s = (-2)*n)

-- analytic continuation of Riemann zeta function.
axiom riemann_zeta_exists :
(∃! ζ : holomorphic_function, ζ.domain = (set.univ \ {1}) ∧
∀ s : subtype ζ.domain, re(s) > 1 → ζ(s) = riemann_zeta_sum s)

def ζ := classical.some riemann_zeta_exists

--  (s ≠ 1) implicit in the domain constraints:
def riemann_hypothesis :=
(∀ s, ζ(s) = 0 ∧ ¬(riemann_zeta_trivial_zero s) →
re (s) = 1/2)
```

What is great about LEAN?

- Lean sounds wonderful: open source, a small trusted kernel, a powerful elaboration engine including a Prolog-like algorithm for type-class resolution, multi-core support, incremental compilation, support for both constructive and classical mathematics, successful projects in homotopy type theory, excellent documentation, and a web browser interface.
- In more detail, a "minimalist and high performance kernel" was an explicit goal of the Lean. Independent implementations of the kernel can have have been given (Selsam 2000 lines, etc.) alleviating any concerns about a bug in the C++ implementation of Lean.
- The semantics of Lean are now completely spelled out (thanks to Mario Carneiro, building on [Werner]). In particular, Carneiro has built a model of Lean's logic (CiC with non-cumulative universes) in ZFC set theory (augmented by a countable number of inaccessible cardinals).
- Lean has a clean syntax. For example, to add two elements in an abelian group, one can simply write `x+y` and Lean correctly infers the group in which the addition is to be performed. I have more to say about Lean's syntax later.
- Lean makes it easy to switch from constructive to classical logic (you just open the classical logic module). Lean makes quotient types easy (unlike Coq, when tends to work with awkward setoids).
- Lean is its own meta language. I find this very appealing. Contrast this with HOL-Light, which has OCaml as meta-language or Coq which has a domain-specific language Ltac for tactics.
- Finally, there was a personal reason. CMU is the center of Lean library development. I live in Pittsburgh and am a regular participant in CMU's Lean group meetings.

Needed improvements in LEAN?

- The kernel is bloated. Specifically, from what I hear, for performance reasons, mutually inductive types will soon be moved into the kernel. This bloats the kernel and kills the former claims of a minimalistic kernel.
- Lean is not backwards compatible. Lean 3 broke the Lean 2 libraries, and old libraries still haven't been ported to Lean 3. After nearly 2 years, it doesn't look like that will ever happen. Instead new libraries are being built (at great cost). Lean 4 is guaranteed to break the Lean 3 libraries (at what cost?). In short, Lean is experimental, evolving, and unstable.
- The learning curve is steep. It is **very hard** to learn to use Lean proficiently. Are you a graduate student at Stanford or CMU writing a thesis on Lean? Are you a student at Imperial being guided by Kevin Buzzard? If not, Lean might not be for you.
- Lean is its own metalanguage. Lean is new, and the language libraries are almost non-existant. 10 million programmers know Java. Hardly any major programs have been written in Lean (Lewis's thesis is a notable exception). It is impossible to do any serious programming in Lean.
- Typing is nominal rather than structural.
- There are performance issues. It is not clear (to me or perhaps even to anyone) why performance is such a big problem, because Lean was implemented in C++ for the sake of performance. However in fact, the compilation of the math libraries is currently very slow. Something is wrong here.
- Ugly projection chains are required.
- Structure depends on notation. Lean has a library of results about multiplicative groups and a separate library of results about additive groups. The only difference is that one uses the symbol `*` for the group operation and the other uses the symbol + for the group operation. Mathematician will find it absurd that the theorems in group theory depend on the symbol used for composition.
- No diamonds are allowed. (For a review of diamonds in OOP, see https://en.wikipedia.org/wiki/Multiple_inheritance.)
- Structures are meaninglessly parameterized from a mathematical perspective. To briefly introduce the topic of parameters and bundling, users choose whether data appears as an external parameter.
- Lean discards valuable information that is later reconstructed (at a cost) by its type class resolution engine.

Even proof assistants based on set theory need to make decisions about subsets. In ZFC, we do not naturally have

$$\mathbb{N} \subset \mathbb{Z} \subset \mathbb{Q} \subset \mathbb{R} \subset \mathbb{C}.$$

The Mizar proof assistant achieves these inclusions by an act of butchery. The image of $\mathbb{N}$ in $\mathbb{Z}$ is excised from $\mathbb{Z}$ and replaced by $\mathbb{N}$, and so forth. But these decisions are quite arbitrary. Why not $\mathbb{Q} \subset \mathbb{Q}_p$?

The HOL Light proof assistant maintains the explicit embeddings:

$$\mathbb{N} \to \mathbb{Z}, \quad \mathbb{Z} \to \mathbb{R}, \quad \text{etc.},$$

(but $\mathbb{Q} \subset \mathbb{R}$).

Proof assistants also need to deal with identifications.

For example, we identify $\mathbb{Q}_p$ (the completion of the field $\mathbb{Q}$ with respect to the p-adic norm) with the field of fractions of $\mathbb{Z}_p$ (defined as an inverse limit of $\mathbb{Z}/p^n\mathbb{Z}$).

We identify

$$GL(2, \mathbb{A}) \quad \text{and} \quad \Pi'_v GL(2, \mathbb{Q}_v),$$

where $\mathbb{A} = \Pi'_v \mathbb{Q}_v$. However, the elements of one are matrices with coefficients in a restricted product of fields, but the right hand side is are restricted product of groups.

We identify $X \times (X \times X)$ with $(X \times X) \times X$, except when we don't.

Abuses of Language

- Structured math objects. Is a group a set G or a tuple (G,*,1,inv)?
- Structured math objects. A topological group is neither a group nor a topological space. A metric space is not a topological space.
- A polynomial is both a function and an element of R[x]. (This distinction must be preserved.)
- The ring of integers is not really a subset of the field of rational numbers.   A complex vector space is not really a real vector space.
- Complete ordered fields (such as the field of real numbers) are only unique up to unique isomorphism.
- A measurable function is an equivalence class of functions. “f is continuous.”
- X x (Y x Z) = (X x Y) x Z means canonical isomorphism between the two.

The definitions of mathematics

The Oxford English dictionary (2nd edition) has 273,000 headwords and over 600,000 word forms. (The longest entry is for the word set, which continues for 25 pages).

Medicine has a specialized terminology of approximately 250,000 items [Kucharz].

The Math Subject Classification (MSC) lists over 6000 subfields of mathematics.

Supreme Court Justices, law professor play with words

Tuesday, January 12, 2010

Supreme Court justices deal in words, and they are always on the lookout for new ones.

University of Michigan law professor Richard D. Friedman discovered that Monday when he answered a question from Justice Anthony M. Kennedy, but added that it was "entirely orthogonal" to the argument he was making in *Briscoe v. Virginia*.

Friedman attempted to move on, but Chief Justice John G. Roberts Jr. stopped him.

"I'm sorry," Roberts said. "Entirely *what*?"

"Orthogonal," Friedman repeated, and then defined the word: "Right angle. Unrelated. Irrelevant."

"Oh," Roberts replied.

Friedman again tried to continue, but he had caught the interest of Justice Antonin Scalia, who considers himself the court's wordsmith. Scalia recently criticized a lawyer for using "choate" to mean the opposite of "inchoate," a word that has created a debate in the dictionary world.

"What was that adjective?" Scalia asked Monday. "I liked that."

"Orthogonal," Friedman said.

"Orthogonal," Roberts said.

"Orthogonal," Scalia said. "Ooh."

Friedman seemed to start to regret the whole thing, saying the use of the word was "a bit of professorship creeping in, I suppose," but Scalia was happy.

"I think we should use that in the opinion," he said.

"Or the dissent," added Roberts, who in this case was in rare disagreement with Scalia.

-- Robert Barnes

Sylvester, "On a theory of Syzygetic Relations"

allotrious, apocapated, Bezoutic, Bezoutoid, co-bezoutiant, cogredient, contragredient, combinant, concomitant, conjunctive, contravariant, covariant, cumulant, determinant, dialytic, discriminant, disjunctive, effluent, emanant, endoscopic, exoscopic, Hessian, hyperdeterminant, inertia, intercalation, invariance, invariant, Jacobian, kenotheme, matrix, minor determinant, monotheme, persymmetrical, quadrinvariant, resultant, rhizoristic, signaletic, semaphoretic, substitution, syrrhizoristic, syzygetic, transform, umbral.

Math Words

- rng = ring without i
- lluf subcategory = full backwards
- clopen = closed and open, bananaman = Banach analytic manifold,
- bra and ket (from bracket), parahori = parabola + Iwahori,
- icthyomorphisms = transformations between Poisson manifolds
- pointless topology, killing fields, abstract nonsense
- alfalfa (derived from alpha by the Iowa school of representation theory)
- the unknot (a circle) was coined during 7-ups uncola advertising campaign.
- Conwayisms: nimber, moonshine, baby monster
- buildings (apartment, chamber, wall, etc.), tree (forest, leaf, root, etc.), quivers (arrows).
- cepstrum (spectrum) in quefrency analysis
- Pin is to O, what Spin is to SO.
- iff, xor, wlog, nth,
- snark, quark, fluxion, gerbe, totient, heteroscedasticity, anabelian, zenzizenzizenzic, Nullstellensatz, Entscheidungsproblem

VOCABULARY OF THE KEPLER CONJECTURE

- quoin, negligible, fcc-compatible, decomposition star, score, score adjustment, quasi-regular tetrahedron, contravening, tame graph, pentahedral prism, crown, quarter, upright, flat, quartered octahedron, strict quarter, enclosed vertex, central vertex, corners, isolated quarter, isolated pair, conflicting diagonals, Q-system, S-system, V-cells, barrier, obstructed, face with negative orientation, Delaunay star, colored spaces, compression, quad cluster, mixed quad cluster, standard cluster, standard region, vertex type, quad cluster, Rogers simplex, anchor, anchored simplex, erasing, loops, subcluster, corner cell, truncated corner cell , tame graph, weight assignment, contravening circuit, crowded diagonal, n-crowded, masked, confined, penalties, penalty-free score, exceptional region, special simplex, distinguished edge, nonexternal edge, concave corner, concave vertex, t-cone, partial plane graph, patch, aggregated face,

VOCABUARY OF IUT1/ABC (MOCHIZUKI)

- inter-universal Teichmuller theory, semi-graphs of anabelioids, Frobenioids, etale theta function, log-shells, log-theta-lattices, log-link, log-volume, initial Theta-data, Hodge theaters, absolute anabelian geometry, absolute anabelian reconstruction, tempered fundamental group, prime-strips, local arithmetic holomorphic structure, mono-analyticizations, mono-analytic core, global realified Frobenioid, labels, label crushing, conjugate synchronization, Frobenioid-theoretic theta function, full poly-isomorphisms, multiradiality, alien ring structures, alien arithmetic holomorphic structure, cyclotomic rigidity isomorphism, real analytic container, mono-analytic container, Theta-link, Theta-dilation, Belyi cuspidalization, topological pseudo-monoid, capsule of objects, capsule indices, connected temperoid, commensurably terminal, co-holomorphicization, base-NF-bridges, poly-action, cyclotomes, coric structure, Kummer black-out, Kummer-blind, solvable factorization, dismantling, functorial dynamics, holomorphic procession, entangled structures, indigenous bundle

Trott's MathOverflow data

1 Chinese remainder theorem
2 prime number theorem
3 central limit theorem
4 Fermat's Last theorem
5 Hahn-Banach theorem
6 Atiyah-Singer index theorem
7 implicit function theorem
8 Riemann-Roch theorem
9 spectral theorem
10 Riemann mapping theorem
11 Riesz representation theorem
12 Gauss-Bonnet theorem
13 Dirichlet's theorem
14 Jordan curve theorem
15 incompleteness theorem
16 Liouville's theorem
17 Fubini's theorem
18 Brouwer fixed point theorem
19 universal coefficient theorem
20 intermediate value theorem
21 Whitehead theorem
22 mean value theorem
23 uniformization theorem
24 Ramsey's theorem
25 Peter-Weyl theorem
26 inverse function theorem
27 Baire category theorem
28 Mordell-Weil theorem
29 Frobenius theorem
30 Stokes theorem
31 Pythagorean theorem
32 Cayley-Hamilton theorem
33 Perron-Frobenius theorem
34 Birkhoff ergodic theorem
35 Main theorem
36 Lefschetz fixed point theorem
37 Bertini's theorem
38 Hodge theorem
39 Sylow theorem
40 fundamental theorem of algebra
41 Stone-Weierstrass theorem
42 Roth's theorem
43 Second Incompleteness theorem
44 Riemann Existence theorem
45 Cauchy's theorem
46 residue theorem
47 Torelli theorem
48 dominated convergence theorem
49 Chevalley's theorem
50 open mapping theorem
51 Sobolev embedding theorem
52 fundamental theorem of calculus
53 Tychonoff's theorem
54 Taylor's theorem
55 Tarski's theorem
56 comparison theorem
57 Recursion theo
58 Radon-Nikodym
59 Value theorem
60 theorem
61 Whitney embedding theorem
62 Lowenheim-Skolem theorem
63 Minkowski's theorem
64 Vanishing theorem
65 van Kampen theorem
66 Cayley's theorem
67 Noether's theorem
68 Rolle's theorem
69 Lebesgue density theorem
70 Kodaira vanishing theorem
71 Weierstrass approximation theorem
72 Hall's marriage theorem
73 MRDP theorem
74 Krull-Schmidt theorem
75 Wilson's theorem
76 Whitney extension theorem
77 Whitney's theorem
78 Tauberian theorem
79 Weyl's theorem
80 Schwartz kernel theorem
81 Rice's theorem
82 Weil's theorem
83 Thue-Siegel-Roth theorem
84 Hodge decomposition theorem
85 Their theorem
86 Wedderburn's theorem
87 Stone representation theorem
88 Unit theorem
89 Turan's theorem
90 Yau's theorem
91 Tate's theorem
92 Mean Value theorem
93 Chinese Remainder theorem
94 binomial theorem
95 intermediate value theorem
96 Pythagorean theorem
97 Value theorem
98 residue theorem
99 squeeze theorem
100 dominated convergence theorem
101 Fermat's little theorem
102 fundamental theorem of calculus
103 Central Limit theorem
104 Lagrange's theorem
105 Fubini's theorem
106 implicit function theorem
107 first isomorphism theorem
108 Cauchy's theorem
109 Sylow theorem
110 inverse function theorem
111 rank-nullity theorem
112 spectral theorem
116 Rolle's theorem
117 Cayley-Hamilton theorem
118 prime number theorem
119 Liouville's theorem
120 Fermat's Last theorem
121 Green's theorem
122 open mapping theorem
123 Monotone Convergence theorem
124 Heine-Borel theorem
125 Cauchy's integral theorem
126 fundamental theorem of algebra
127 rational root theorem
128 Bolzano-Weierstrass theorem
129 Stokes theorem
130 Master theorem
131 identity theorem
132 Bayes theorem
133 Banach fixed point theorem
134 fundamental theorem of arithmetic
135 Baire category theorem
136 isomorphism theorem
137 Dirichlet's theorem
138 Stone-Weierstrass theorem
139 Riemann mapping theorem
140 Pythagoras theorem
141 Factor theorem
142 Wilson's theorem
143 Jordan curve theorem
144 Fermat's theorem
145 Weierstrass theorem
146 Weierstrass approximation theorem
147 closed graph theorem
148 Cantor's theorem
149 orbit-stabilizer theorem
150 Radon-Nikodym theorem
151 Tonelli's theorem
152 convolution theorem
153 incompleteness theorem
154 fundamental theorem of calculus.
155 universal coefficient theorem
156 Arzela-Ascoli theorem
157 uniqueness theorem
158 Picard's theorem
159 Sandwich theorem
160 Tychonoff's theorem
161 correspondence theorem
162 Bezout's theorem
163 Remainder theorem
164 Rouche's theorem
165 Cantor-Bernstein theorem
166 Tietze extension theorem
167 multinomial theorem
168 Kampen theorem

https://en.wikipedia.org/wiki/Normal

Search

What is normal in math?

There are many unrelated notions of "normality" in mathematics.

Algebra and number theory [edit source]

- Normal basis (of a Galois extension), used heavily in cryptography
- Normal degree, a rational curve on a surface that meets certain conditions
- Normal domain (integrally closed domain), a ring integrally closed in its fraction field
  - Normal ring, a reduced ring whose localizations at prime ideals are integrally closed domains
  - Normal scheme, an algebraic variety or scheme that meets certain conditions
- Normal extensions (or quasi-Galois) field extensions, splitting fields for a set of polynomials over the base field
- Normal variety, a projective variety embedded by a complete linear system, as in a rational normal scroll (unrelated to the concept of normal scheme above)
- Normal order of an arithmetic function, a type of asymptotic behavior useful in number theory
- Normal subgroup, a subgroup invariant under conjugation

Analysis [edit source]

- Normal family, a pre-compact family of continuous functions
- Normal number, a real number with a "uniform" distribution of digits
- Normal number (computing), a floating-point number within the balanced range supported by a given format (unrelated to the previous notion)
- Normal operator, an operator that commutes with its Hermitian adjoint
  - Normal matrix, a complex square matrix that meets certain conditions
- Normal modes of vibration in an oscillating system

Geometry [edit source]

- Normal (geometry), a vector perpendicular to a surface (normal vector)
- Normal bundle, a term related to the preceding concept
- Normal cone, of a subscheme in algebraic geometry
- Normal coordinates, in differential geometry, local coordinates obtained from the exponential map (Riemannian geometry)
- Normal invariants, in geometric topology
- Normal polytopes, in polyhedral geometry and computational commutative algebra
- Normal space (or T_4) spaces, topological spaces characterized by separation of closed sets

Logic and foundations [edit source]

- Normal function, in set theory
- Normal measure, in set theory

Mathematical physics [edit source]

- Normal order or Wick order in Quantum Field Theory

Probability and statistics [edit source]

- Normal, the middle 95% of a bell curve (see 1.96)
- Normal distribution, the Gaussian continuous probability distribution

Other mathematics [edit source]

- Normal form (disambiguation)
- Normalization (disambiguation)

What is a group?

Definitions of group (algebra)

- A group is a set with a binary operation, identity element, and inverse operation, satisfying axioms of associativity, inverse, and identity.
- A group object in a category. A group in the first sense is a group object in the category of sets. A Lie group is a group object in the category of smooth manifolds. A topological group is a group object in the category of topological spaces. An affine group scheme is a group object in the category of affine schemes. (Caution: the Zariski product topology is not the product topology.)
- A Poisson-Lie group a group object in the category of Poisson manifolds, except that the inverse operation is not required to be a morphism of Poisson manifolds. (In

What is a group?

general, the inverse is an anti-Poisson morphism.)

- A quantum group is an object in the opposite category to the category of Hopf algebras.
- A compact matrix quantum group is a C^*-algebra with additional structure (Woronowicz).
- A strict 2-group is a group object in the category of categories (or a category object in the category of groups).
- A 2-group $\ldots$
- An n-group $\ldots$
- A formal group

Mathematics Subject Classification – MSC2010

- **00** General mathematics
- **01** History and biography
- **03** Mathematical logic and foundations
- **05** Combinatorics
- **06** Order, lattices, ordered algebraic structures
- **08** General algebraic systems
- **11** Number theory
- **12** Field theory and polynomials
- **13** Commutative algebra
- **14** Algebraic geometry
- **15** Linear and multilinear algebra; matrix theory
- **16** Associative rings and algebras
- **17** Nonassociative rings and algebras
- **18** Category theory, homological algebra
- **19** K-theory
- **20** Group theory and generalizations
- **22** Topological groups, Lie groups
- **26** Real functions
- **28** Measure and integration
- **30** Functions of a complex variable
- **31** Potential theory
- **32** Several complex variables and analytic spaces
- **33** Special functions
- **34** Ordinary differential equations
- **35** Partial differential equations
- **37** Dynamical systems and ergodic theory
- **39** Difference and functional equations
- **40** Sequences, series, summability
- **41** Approximation and expansions
- **42** Harmonic analysis on Euclidean spaces
- **43** Abstract harmonic analysis
- **44** Integral transforms, operational calculus
- **45** Integral equations
- **46** Functional analysis
- **47** Operator theory
- **49** Calculus of variations and optimal control; optimization
- **51** Geometry
- **52** Convex and discrete geometry
- **53** Differential geometry
- **54** General topology
- **55** Algebraic topology
- **57** Manifolds and cell complexes
- **58** Global analysis, analysis on manifolds
- **60** Probability theory and stochastic processes
- **62** Statistics
- **65** Numerical analysis
- **68** Computer science
- **70** Mechanics of particles and systems
- **74** Mechanics of deformable solids
- **76** Fluid mechanics
- **78** Optics, electromagnetic theory
- **80** Classical thermodynamics, heat transfer
- **81** Quantum Theory
- **82** Statistical mechanics, structure of matter
- **83** Relativity and gravitational theory
- **85** Astronomy and astrophysics
- **86** Geophysics
- **90** Operations research, mathematical programming
- **91** Game theory, economics, social and behavioral sciences
- **92** Biology and other natural sciences
- **93** Systems theory; control
- **94** Information and communication, circuits
- **97** Mathematics education

14B99 None of the above, but in this section

14Cxx Cycles and subschemes

14C05 Parametrization (Chow and Hilbert schemes)
14C15 (Equivariant) Chow groups and rings; motives
14C17 Intersection theory, characteristic classes, intersection multiplicities [See also 13H15]
14C20 Divisors, linear systems, invertible sheaves
14C21 Pencils, nets, webs [See also 53A60]
14C22 Picard groups
14C25 Algebraic cycles
14C30 Transcendental methods, Hodge theory [See also 14D07, 32G20, 32J25, 32S35], Hodge conjecture
14C34 Torelli problem [See also 32G20]
14C35 Applications of methods of algebraic K-theory [See also 19Exx]
14C40 Riemann-Roch theorems [See also 19E20, 19L10]
14C99 None of the above, but in this section

14Dxx Families, fibrations

14D05 Structure of families (Picard-Lefschetz, monodromy, etc.)
14D06 Fibrations, degenerations
14D07 Variation of Hodge structures [See also 32G20]
14D10 Arithmetic ground fields (finite, local, global)
14D15 Formal methods; deformations [See also 13D10, 14B07, 32Gxx]
14D20 Algebraic moduli problems, moduli of vector bundles {For analytic moduli problems, see 32G13}
14D21 Applications of vector bundles and moduli spaces in mathematical physics (twistor theory, instantons, quantum field theory) [See also 32L25, 81Txx]
14D22 Fine and coarse moduli spaces
14D23 Stacks and moduli problems
14D24 Geometric Langlands program: algebro-geometric aspects [See also 22E57]
14D99 None of the above, but in this section

14Exx Birational geometry

14E05 Rational and birational maps
14E07 Birational automorphisms, Cremona group and generalizations
14E08 Rationality questions [See also 14M20]
14E15 Global theory and resolution of singularities [See also 14B05, 32S20, 32S45]
14E16 McKay correspondence
14E18 Arcs and motivic integration
14E20 Coverings [See also 14H30]
14E22 Ramification problems [See also 11S15]
14E25 Embeddings
14E30 Minimal model program (Mori theory, extremal rays)
14E99 None of the above, but in this section

14Fxx (Co)homology theory [See also 13Dxx]

14F05 Sheaves, derived categories of sheaves and related constructions [See also 14H60, 14J60, 18F20, 32Lxx, 46M20]
14F10 Differentials and other special sheaves; D-modules; Bernstein-Sato ideals and polynomials [See also 13Nxx, 32C38]
14F17 Vanishing theorems [See also 32L20]
14F18 Multiplier ideals
14F20 Étale and other Grothendieck topologies and (co)homologies
14F22 Brauer groups of schemes [See also 12G05, 16K50]
14F25 Classical real and complex (co)homology
14F30 p-adic cohomology, crystalline cohomology
14F35 Homotopy theory; fundamental groups [See also 14H30]
14F40 de Rham cohomology [See also 14C30, 32C35, 32L10]
14F42 Motivic cohomology; motivic homotopy theory [See also 19E15]
14F43 Other algebro-geometric (co)homologies (e.g., intersection, equivariant, Lawson, Deligne (co)homologies)

14G99 None of the above, but in this section

14Hxx Curves

14H05 Algebraic functions; function fields [See also 11R58]
14H10 Families, moduli (algebraic)
14H15 Families, moduli (analytic) [See also 30F10, 32G15]
14H20 Singularities, local rings [See also 13Hxx, 14B05]
14H25 Arithmetic ground fields [See also 11Dxx, 11G05, 14Gxx]
14H30 Coverings, fundamental group [See also 14E20, 14F35]
14H37 Automorphisms
14H40 Jacobians, Prym varieties [See also 32G20]
14H42 Theta functions; Schottky problem [See also 14K25, 32G20]
14H45 Special curves and curves of low genus
14H50 Plane and space curves
14H51 Special divisors (gonality, Brill-Noether theory)
14H52 Elliptic curves [See also 11G05, 11G07, 14Kxx]
14H55 Riemann surfaces; Weierstrass points; gap sequences [See also 30Fxx]
14H57 Dessins d'enfants theory {For arithmetic aspects, see 11G32}
14H60 Vector bundles on curves and their moduli [See also 14D20, 14F05]
14H70 Relationships with integrable systems
14H81 Relationships with physics
14H99 None of the above, but in this section

14Jxx Surfaces and higher-dimensional varieties {For analytic theory, see 32Jxx}

14J10 Families, moduli, classification: algebraic theory
14J15 Moduli, classification: analytic theory; relations with modular forms [See also 32G13]
14J17 Singularities [See also 14B05, 14E15]
14J20 Arithmetic ground fields [See also 11Dxx, 11G25, 11G35, 14Gxx]
14J25 Special surfaces {For Hilbert modular surfaces, see 14G35}
14J26 Rational and ruled surfaces
14J27 Elliptic surfaces
14J28 $K3$ surfaces and Enriques surfaces
14J29 Surfaces of general type
14J30 3-folds [See also 32Q25]
14J32 Calabi-Yau manifolds
14J33 Mirror symmetry [See also 11G42, 53D37]
14J35 4-folds
14J40 n-folds ($n > 4$)
14J45 Fano varieties
14J50 Automorphisms of surfaces and higher-dimensional varieties
14J60 Vector bundles on surfaces and higher-dimensional varieties, and their moduli [See also 14D20, 14F05, 32Lxx]
14J70 Hypersurfaces
14J80 Topology of surfaces (Donaldson polynomials, Seiberg-Witten invariants)
14J81 Relationships with physics
14J99 None of the above, but in this section

14Kxx Abelian varieties and schemes

14K02 Isogeny
14K05 Algebraic theory
14K10 Algebraic moduli, classification [See also 11G15]
14K12 Subvarieties
14K15 Arithmetic ground fields [See also 11Dxx, 11Fxx, 11G10, 14Gxx]
14K20 Analytic theory; abelian integrals and differentials
14K22 Complex multiplication [See also 11G15]
14K25 Theta functions [See also 14H42]
14K30 Picard schemes, higher Jacobians [See also 14H40, 32G20]
14K99 None of the above, but in this section

14Lxx Algebraic groups {For linear algebraic groups, see 20Gxx; for Lie algebras, see 17B45}

14L05 Formal groups, p-divisible groups [See also 55N22]

Sign Manifesto

Pierre Deligne and Daniel S. Freed

§1. Standard mathematical conventions

- *We apply the sign rule relentlessly.*

§2. Choices

- *A hermitian inner product on a complex vector space V is conjugate linear in the first variable:*

(3) $$\langle \lambda_1 v_1, \lambda_2 v_2 \rangle = \overline{\lambda_1} \lambda_2 \langle v_1, v_2 \rangle, \qquad \lambda_i \in \mathbb{C}, \quad v_i \in V.$$

- *If $V = V^0 \oplus V^1$ is a super Hilbert space, then*

(4) $$-i\langle v, v \rangle \geq 0, \qquad v \in V^1.$$

§7. Miscellaneous signs

- *Let X be a smooth manifold, ξ a vector field on X, φ_t the one-parameter group of diffeomorphisms generated, and T a tensor field. Then*

(39) $$\mathrm{Lie}(\xi)T = \frac{d}{dt}\Big|_{t=0} \varphi_t^* T = \frac{d}{dt}\Big|_{t=0} (\varphi_{-t})_* T.$$

Hartshorne (Residues and Duality): "And since the chore of inventing these diagrams and checking their commutativity is almost mechanical, the reader would not want to read them, nor I write them."

"the reader [of Hartshorne] is left with checking lots and lots of commutative diagrams, some of them depending on very subtle sign conventions in homological algebra!"

Recent and Current Projects

Recent and Current Projects

A formalization of forcing and the unprovability of the continuum hypothesis

Jesse Michael Han[1]
Department of Mathematics, University of Pittsburgh
https://www.pitt.edu/~jmh288
jessemichaelhan@gmail.com

Floris van Doorn
Department of Mathematics, University of Pittsburgh
http://florisvandoorn.com/
fpvdoorn@gmail.com

Abstract

We describe a formalization of forcing using Boolean-valued models in the Lean 3 theorem prover, including the fundamental theorem of forcing and a deep embedding of first-order logic with a Boolean-valued soundness theorem. As an application of our framework, we specialize our construction to the Boolean algebra of regular opens of the Cantor space $2^{\omega_2 \times \omega}$ and formally verify the failure of the continuum hypothesis in the resulting model.

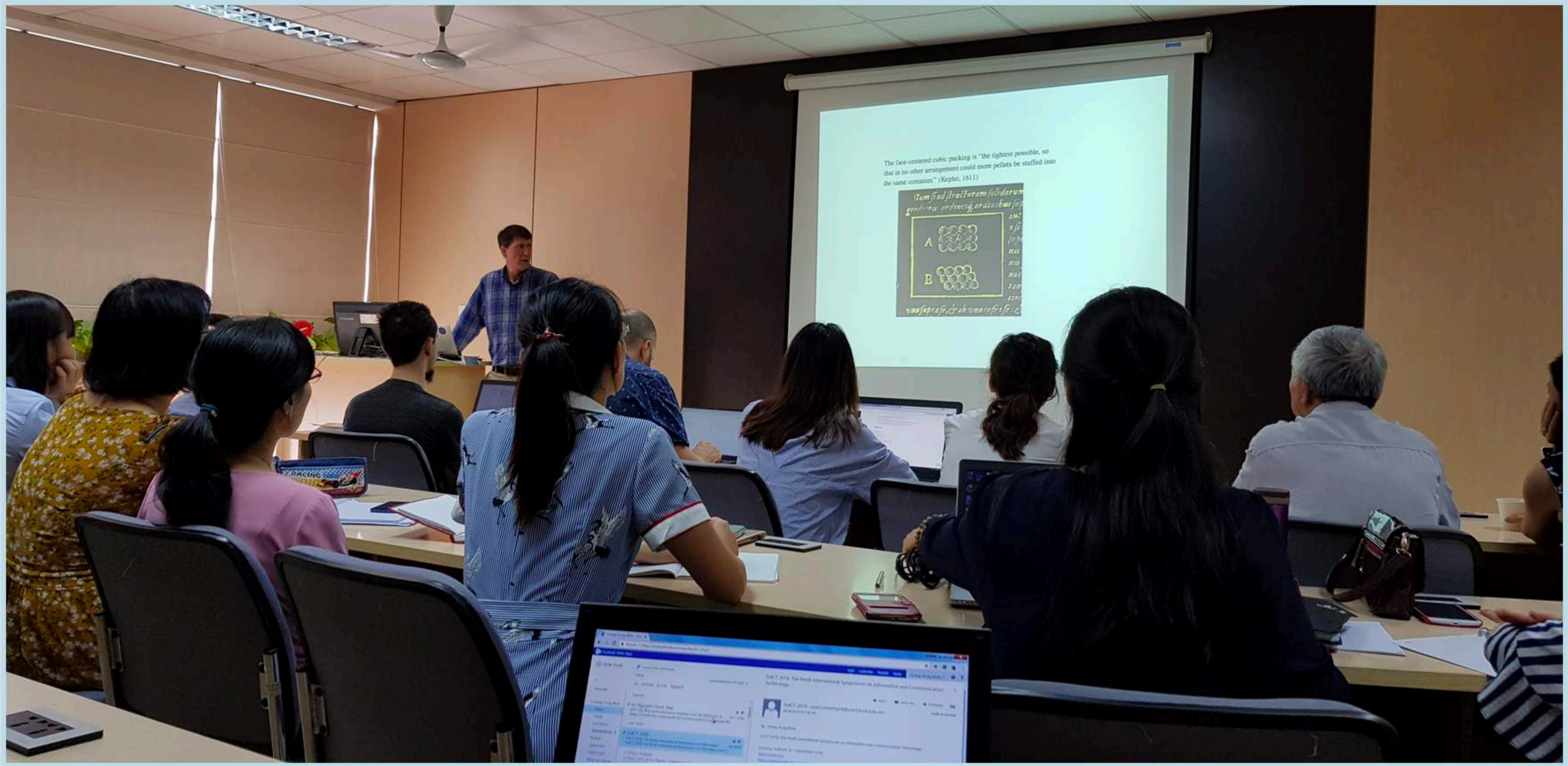
PITTSBURGH GROUP

FORMAL ABSTRACTS PROJECT

- Floris van Doorn
- Luis Berlioz (Creating a Database of Definitions From Large Mathematical Corpora)
- Jesse Han
- Koundinya Vajjha
- working with Jeremy Avigad (CMU Lean Working Group)
- working with Tran Nam Trung (Thang Long University and Mathematics Institute, VAST, Hanoi)

HANOI SUMMER SCHOOL ON FORMAL ABSTRACTS

2018



Location: Thang Long University

Dates: June 5-14

Lecturer: **Thomas Hales,** Tran Nam Trung, Jonhalnnes Holzl, Mario Carneiro, Jesse Han

Hanoi Lean 2019

Conference on Lean and Formal Abstracts in Hanoi, June 17-20, 2019

Continuum Hypothesis

Introduction

The continuum hypothesis states that there are no sets strictly larger than the countable natural numbers and strictly smaller than the uncountable real numbers. It was introduced by Cantor [7] in 1878 and was the very first problem on Hilbert's list of twenty-three outstanding problems in mathematics. Gödel [14] proved in 1938 that the continuum hypothesis was consistent with ZFC, and later conjectured that the continuum hypothesis is independent of ZFC, i.e. neither provable nor disprovable from the ZFC axioms. In 1963, Paul Cohen developed *forcing* [10, 11], which allowed him to prove the consistency of the negation of the continuum hypothesis, and therefore complete the independence proof. For this work, which marked the beginning of modern set theory, he was awarded a Fields medal—the only one to ever be awarded for a work in mathematical logic.

In this paper we discuss the formalization of a Boolean-valued model of set theory where the continuum hypothesis fails. The work we describe is part of the Flypitch project, which aims to formalize the independence of the continuum hypothesis. Our results mark a major milestone towards that goal.

Example. ZF set theory can be embedded into Lean. The construction is due to Aczel and Benjamin Werner and the implementation in Lean was done by Mario Carneiro. It can be done with a single constructor.

$$\mathrm{im} : \Pi(A : \mathrm{Type}), (A \to \mathrm{Set}) \to \mathrm{Set}$$

Interpret $im\ \ A\ f$ as the image of $f : A \to Set$ on A. So ZFC sets in Lean consist of all images of functions into sets.

Equality is defined recursively: $\mathrm{im}\ \ A\ f$ is equal to $\mathrm{im}\ \ B\ g$ if for every $a : A$ there exists a $b : B$ such that $f(a)$ and $g(b)$ are equal, and vice versa.

Continuum Hypothesis

```
inductive pSet : Type (u+1)
| mk (α : Type u) (A : α → pSet) : pSet
```

The Aczel-Werner encoding is closely related to the recursive definition of *names*, which is used in forcing to construct forcing extensions:

```
inductive bSet (𝔹 : Type u) [complete_boolean_algebra 𝔹] : Type (u+1)
| mk (α : Type u) (A : α → bSet) (B : α → 𝔹) : bSet
```

Recent and Current Projects

mathoverflow

Home

Questions

Tags

Users

Unanswered

Which mathematical definitions should be formalised in Lean?

Ask Question

90

53

The question.

Which mathematical objects would you like to see formally defined in the Lean Theorem Prover?

Examples.

In the current stable version of the Lean Theorem Prover, topological groups have been done, schemes have been done, Noetherian rings got done last month, Noetherian schemes have not yet been done (but are probably not going to be too difficult, if anyone is interested in trying), but complex manifolds have not yet been done. In fact I think we are nearer to perfectoid spaces than complex manifolds -- maybe because algebra is closer to the axioms than analysis. But actually we also have Lebesgue measure (it's differentiability we're not too strong at), and today we got modular forms. There is a sort of an indication of where we are.

asked 3 months ago

viewed 8,986 times

active 2 months ago

BLOG

Adios to Winter Bash 2018

Linked

36 On proof-verification using Coq

Recent and Current Projects

LIST OF FINITE SIMPLE GROUPS

1. Background

This article assumes basic facts about K-algebras (such as tensor products, ideals, radical ideals), topological spaces (connectedness), and category theory.

Building on those foundations, the article gives a complete specification of all finite simple groups. The definition of a finite simple group of Lie type appears in Definition 3. Unexplained notation from this section will be precisely defined later.

Theorem 1. *Every finite simple group is isomorphic to*

(1) *a cyclic group of prime order,*
(2) *an alternating group Alt_n on n letters for some $n \geq 5$,*
(3) *a finite simple group of Lie type, or*
(4) *one of the 26 sporadic groups.*

Every group these four families is a finite simple group.

Finite simple groups of Lie type are classified by certain data of the form (D_r, ρ, p, e) (written as ${}^{\rho}D_r(p^e)$), where D_r is a connected Dynkin diagram with r nodes, ρ is an arrow-forgetful isomorphism of the Dynkin diagram, p is a prime number, and $e \in \mathbb{Q}$ is an exponent. The explicit list of such tuples appears in Definition 1.

Recent and Current Projects

CLASSIFICATION OF FSG

GROUP OBJECTS - CATEGORIFICATION ALGORITHM

25 lines (18 sloc) | 980 Bytes

Raw | Blame | History

```
-- Copyright (c) 2019 Jesse Han. All rights reserved.
-- Released under Apache 2.0 license as described in the file LICENSE.
-- Authors: Jesse Han

import .finite_limits

open category_theory category_theory.limits category_theory.limits.binary_product
     category_theory.limits.finite_limits

universes u v

local infix ` × `:60 := binary_product

local infix ` ×.map `:60 := binary_product.map

structure group_object (C : Type u) [𝒞 : category.{v u} C] [H : has_binary_products C] [H' : has_limits_of_shape
(G : C)
(mul : G × G ⟶ G)
(mul_assoc : (by exact reassoc_hom G) ≫ (by apply (𝟙 _) ×.map mul) ≫ mul = (by apply mul ×.map (𝟙 _) ) ≫ mul)
(one : term ⟶ G)
(one_mul : (𝟙 G) = one_mul_inv _ ≫ (by apply one ×.map (𝟙 G)) ≫ mul)
(mul_one : (𝟙 G) = mul_one_inv _ ≫ (by apply (𝟙 G) ×.map one) ≫ mul)
(inv : G ⟶ G)
(mul_left_inv : (𝟙 G) = (map_to_product.mk (inv) (𝟙 G)) ≫ mul )
```

$$\text{categorify}(\lambda x\ y\ z, x * y * z = (x * y) * z)$$

Machine Learning and Mathematical Definitions

- Luis Berlioz is using machine learning to capture mathematical definitions from arXiv papers.

Objective

Create a machine learning system that can find the definitions and the terms being defined in large collections of mathematical texts.

The problem is broken down into two parts:

The Classifier: Tells if a given paragraph is a definition or not

A Named Entity Recognition system: given a definition, returns the term that is being defined (definiendum).

For each part I will describe how to:

- Get and process the relevant data.
- Train and take a look at the results.

A concrete proposal: mathematical FABSTRACTS (formal abstracts)

Given today's technology, it is not reasonable to ask for all proofs to be formalized. But with today's technology, it seems that it should be possible to create a formal abstract service that

- Gives a statement of the main theorem(s) of each published mathematical paper in a language that is both human and machine readable,
- Links each term in theorem statements to a precise definition of that term (again in human/machine readable form), and
- Grounds every statement and definition is the system in some foundational system for doing mathematics.

The language of math

- Ganaselingam “The language of Math” (linguistics of mathematics)
- Wolfram Research (Wolfram Alpha)
- Controlled natural languages for mathematics: Mizar, Naproche, MathNat
- Dyngenpar (a parser that allows extendible grammars, Neumaier and students)

Controlled Natural Language (CNL)

- It is based on a single natural language (such as English).
- It has restricted syntax and semantics. Its design is deliberate and explicit.
- Speakers of the natural language can largely understand the controlled language at least intuitively. (see Tobias Kuhn)
- The definition is intended to exclude artificial languages such as Esperanto and programming languages.

Controlled Natural Languages

- Math Vernacular, (deBruijn, 1987)
- Mizar -which inspired Mizar styles in many proof assistants such as Isar in Isabelle

Examples of CNLs for Mathematics

- Naproche-SAD (and variants Forthel, Naproche, EA,…). (Paskevich, 2007) (Koepke, Cramer, Frerix, 2018) The target is first-order logic.

- MathNat (and variants CLM controlled language of mathematics). (Humayoun's thesis) The target is first-order logic.

- FMathL (formal mathematical language, CONCISE). The target is a graphical representation (sems).

Lessons from Naproche-SAD CNL

- Naproche-SAD (~ 8K lines of Haskell) gives a template for the design of Math CNLs

- Specifically, parsing with (Haskell) parser combinators, monadic, lazy, continuation style,… For example, Parsec

- Not quite a context-free grammar (CFG). It has a fixed collection of non-terminals, but certain “primitive” non-terminals can be dynamically augmented with new production rules.

- Very little linguistics is required to achieve passable English. Various tricks make this possible: canned/stock phrases, synonyms, filler words, etc.

CNL for Lean (proposal)

Definition 7 (binary relation)**.** *A* **binary relation** *is a structure with*

- *a parametric element :* `Type`
- *a relation : element* $\to$ *element* $\to$ `Prop`

In this section, let R denote a fixed binary relation.

In this section, let $(s,\ x,\ y,\ z : R\hat{\ }element)$.

In this section, let $x \leq y$ stand for $R\hat{\ }relation\ x\ y$.

Definition 8 (reflexive)**.** *We say R is* **reflexive** *iff for all x, $x \leq x$.*

Definition 9 (transitive)**.** *We say R is* **transitive** *iff for all $x\ y\ z$, $x \leq y \wedge y \leq z \to x \leq z$.*

Definition 10 (symmetric)**.** *We say R is* **symmetric** *iff for all $x\ y$, $x \leq y \to y \leq x$.*

Definition 11 (preorder)**.** *We say R is a* **preorder** *iff R is symmetric and transitive.*

Definition 12 (equivalence relation)**.** *We say R is an* **equivalence relation** *iff R is reflexive, symmetric and transitive.*

Definition 13 (antisymmetric)**.** *We say R is* **antisymmetric** *iff for all $x\ y$, $x \leq y$ and $y \leq x$ imply $x = y$.*

CNL for Lean (proposal)

Recent and Current Projects

Definition 34 (has_le)**.** *A* **has_le** *is a notational structure with*

- *a typeable* α : `Type`
- *notation_le :* $\alpha \to \alpha \to$ `Prop`

Assuming (implicit C : has_le), let $x \leq y$ denote C^notation_le x y with precedence 70 and no associativity.

Let $x < y$ stand for $x \leq y$ and $x \neq y$ with precedence 70 and no associativity.

Let $x \geq y$ stand for $y \leq x$ with precedence 70 and no associativity.

Let $x > y$ stand for $y < x$ with precedence 70 and no associativity.

Let m is **at most** n stand for $m \leq n$.

Let n is **at least** m stand for $n \geq m$.

Let m is **less than** n stand for $m < n$.

Let n is **greater** than m stand for $n > m$.

CNL for Lean (proposal)

In this section, let G denote a fixed finite group.

Definition 73 (conjugate). *Assume that* $(g : G)$. *Assume that* H *is a subgroup of* G. *The* **conjugate** *of* H *by* g *in* G *is the subgroup* H' *of* G *such that for all* x, $x \in H' \leftrightarrow g * x * g^{-1} \in H$. *This exists and is unique.*

Definition 74 (normalizer). *Assume that* H *is a subgroup of* G. *The* **normalizer of** H **in** G *is the subgroup* N *of* G *such that for all* x, $x \in N \leftrightarrow$ *for all* $h \in H$ *we have* $x^{-1} * h * x \in H$. *This exists and is unique.*

Let $|G|$ denote the order of G.

In this section, let p denote a fixed prime number.

Let m denote the multiplicity of p in $|G|$.

Definition 75 (Sylow). *A* **Sylow** p **subgroup of** G *is a subgroup* P *of* G *such that the subgroup_order of* P *is* p^m.

Definition 76. *Let* $\mathbf{Syl}(p, G) = \{P \mid (P$ *is a Sylow* p *subgroup of* $G)\}$.

Let $\mathbf{n}(p, G)$ the size of $\mathrm{Syl}(p, G)$. This is well subtyped (that is, it is finite).

CNL for Lean (proposal)

Definition 77. *let* $|Norm|$ *be equal to the size of the normalizer of each and every Sylow* p *subgroup in* G*. This exists, is unique, and is well-defined.*

Theorem 78 (Sylow1)**.** *There exists a Sylow* p *subgroup of* G*.*

Theorem 79 (Sylow 2)**.** *If* P, P' *are Sylow* p *subgroups of* G *then there exists* $(g : G)$ *such that* P' *is the conjugate of* P *by* g *in* G*.*

Theorem 80 (Sylow 3a)**.** *Assume that* $|G| = p' * p^m$*. We have* $n(p, G)$ *divides* p'*.*

Theorem 81 (Sylow 3b)**.** *We have* p *divides* $(n(p, G) - 1)$*.*

Theorem 82 (Sylow 3c)**.** *We have* $n(p, G) * |Norm| = |G|$*.*

Thank you!