Do mathematicians dream of data?

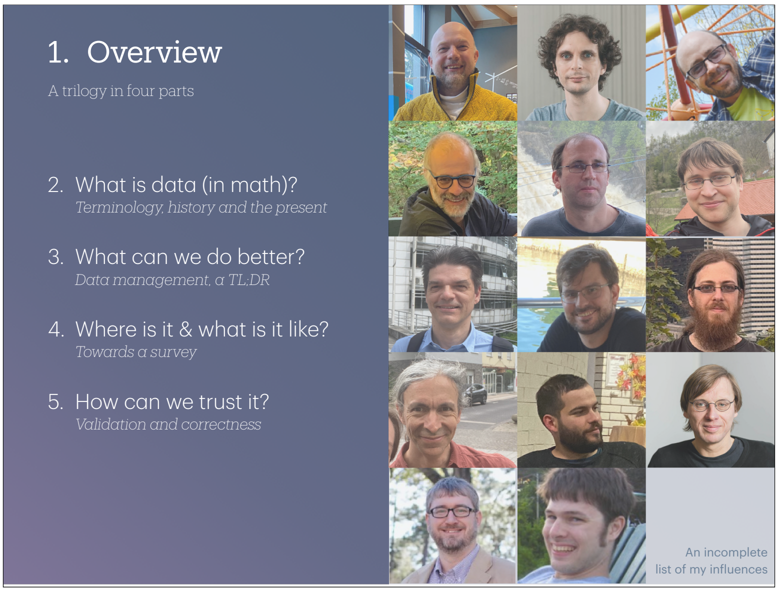
Katja Berčič
University of Ljubljana and IMFM

LMFDB, Computation, and Number Theory (LuCaNT) 2025
ICERM, July 7-11

No.*

* The rest of this talk is about the asterisk

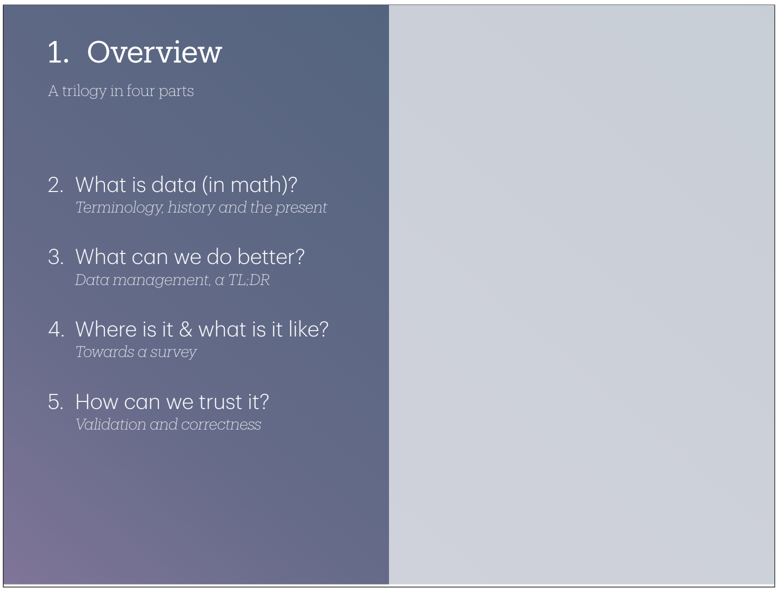
No: most (but not all) mathematicians (perhaps those that don't call themselves applied) would say so. There are some subtleties to consider, and this talk is about those.



On the last slide, you'll get a QR code to a page with a list of references (papers and otherwise), and if I had a productive evening, the slides, too.

Disclaimer

Before we get to the good part, I have a disclaimer to make. The conversation around data in mathematics is still in its early days, and so are the words we use to talk about it. My perspective is inevitably shaped by my background in combinatorics and graph theory — though it seems to me that everyone else is grappling with similar predicaments.



A long time ago, when I was a young and naive undergraduate student, I built a database of tournaments for the Slovenian Go Association. Soon after that I started my PhD and one of the first things I encountered was my advisor's dataset of cubic vertex-transitive graphs. To me, it seemed rather urgent that something like that belonged in a database, not just as a plain text file.

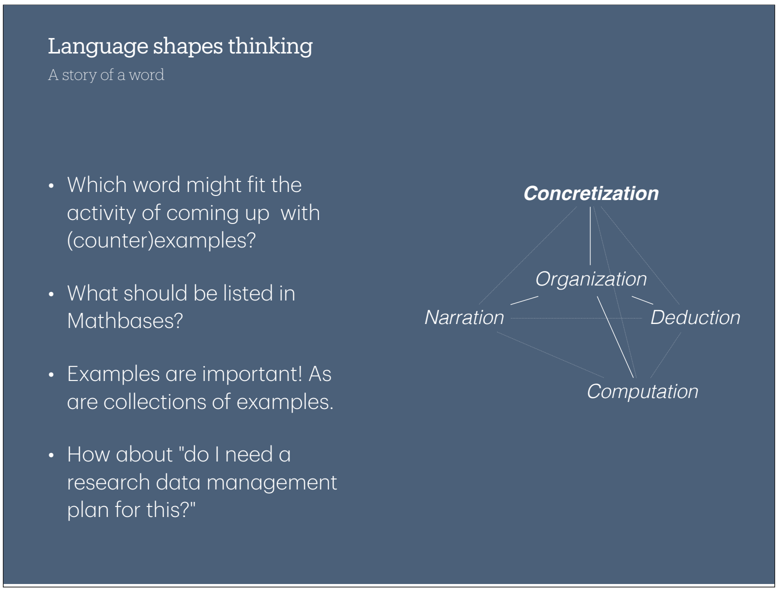
Here's how this talk will unfold:
- As a warm-up (not one of the main sections), we'll spend a bit of time on terminology — with some neat pictures of historical data.
- Then we'll move on to the three main parts: first, FAIR&RDM (the boring bit); second, where some of the datasets are and what they look like; and finally, how much we can trust the data.
- If we're counting generously, this slide could be the "fifth part," making this talk a trilogy in five parts. Don't panic; there won't be a sixth.

2. What is data?

A poll

Scan the QR code on the right or go to `menti.com` and enter the code `2106 7957`

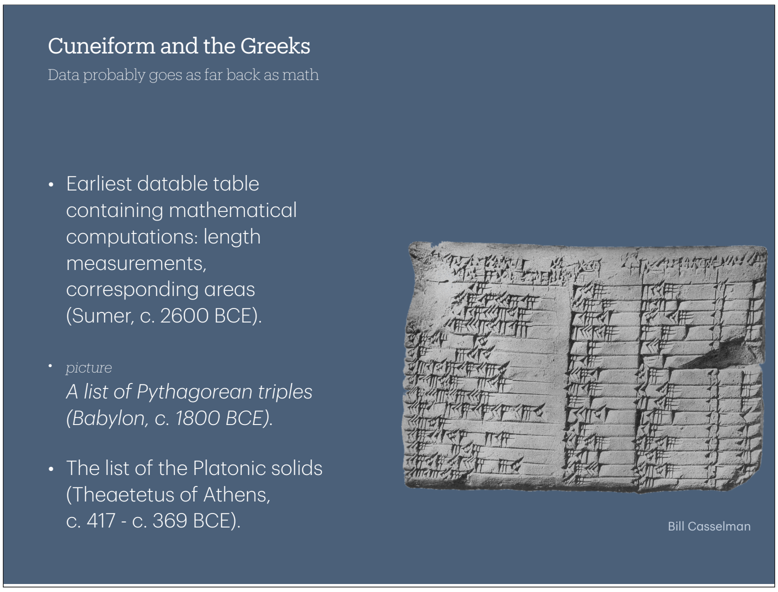
When answering, onsider
- who produces the data,
- who are the users,
- what is the content.



Jacques Carette, Bill Farmer, Michael Kohlhase, and Florian Rabe were searching for a word to describe the core mathematical activity involved in constructing databases. When I joined the conversation, it was still being called tabulation, but even before that, they thought that there might be a more fundamental process beneath that. We eventually settled on concretization (making abstract ideas concrete), though I'm not sure we were ever all completely satisfied with it.

At the time, I was thinking about — and worse, trying to implement — a generic math database (I like to think I came out of that a little wiser). Wrestling with that elusive fifth activity helped me organize my thoughts and I now see instances of concretization in more places than before.

"All digital and analog objects" includes: paper publications, proofs, computational results (and more). Does this mean that all mathematicians should have a research data management plan when they start writing a paper? Probably not, but perhaps they should.

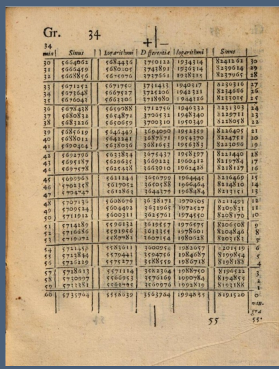


Eleanor Robson, "Tables and tabular formatting in Sumer, Babylonia, and Assyria, 2500 BCE-50," Campbell-Kelly et al [eds]. The History of Mathematical Tables from Sumer to Spreadsheets [2003]

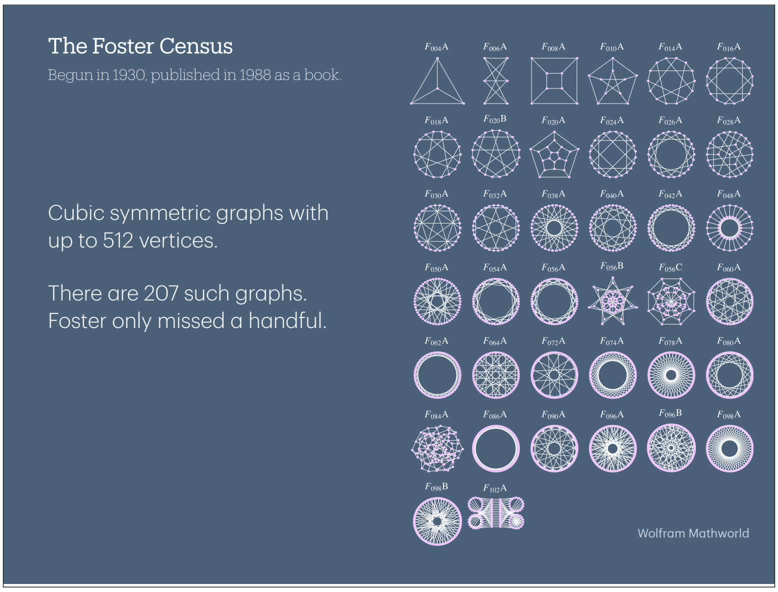
Results of computations

Persistence: data outlasting the computation before computers

- *picture*
  Trigonometric, logarithmic, and exponential functions become subjects of tables (Napier's Mirifici logarithmorum, *trigonometric and log trig data for 34 degrees)*
- Math Tables Project (1938 - 1946): human computers constructed tables of mathematical functions (needed for hand computation).


Gr. 34

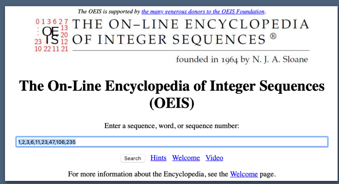
John Napier, 1614

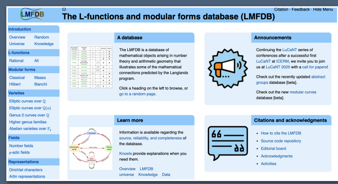


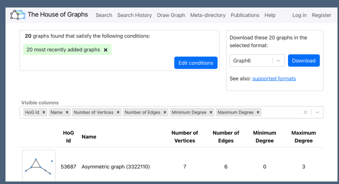
Symmetric graph: every ordered pair of adjacent vertices (an arc) can be mapped to any other such pair.

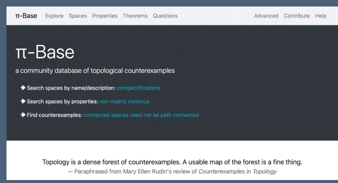
Modern mathematical databases

Four well-established examples, just in case

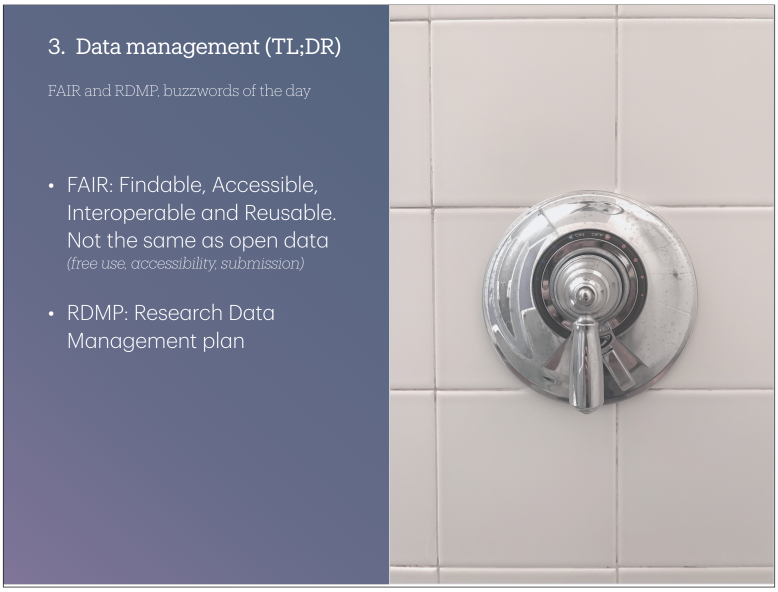




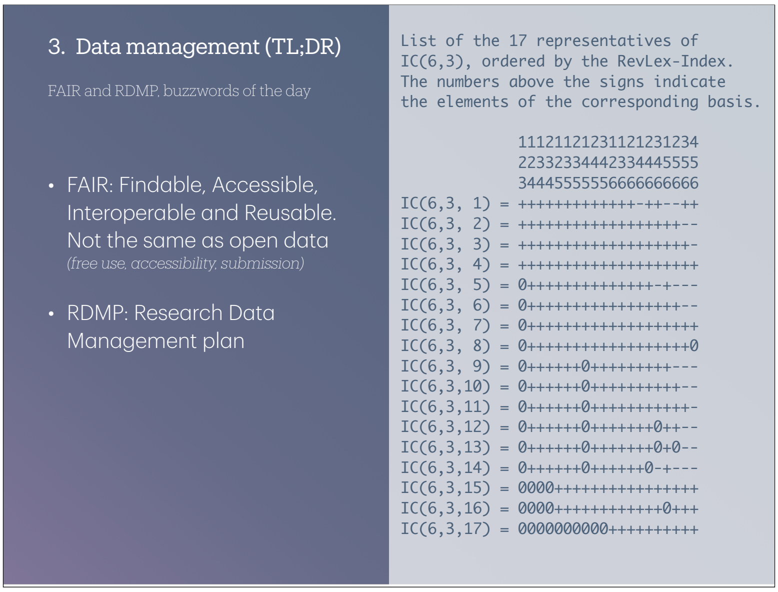




Questions?



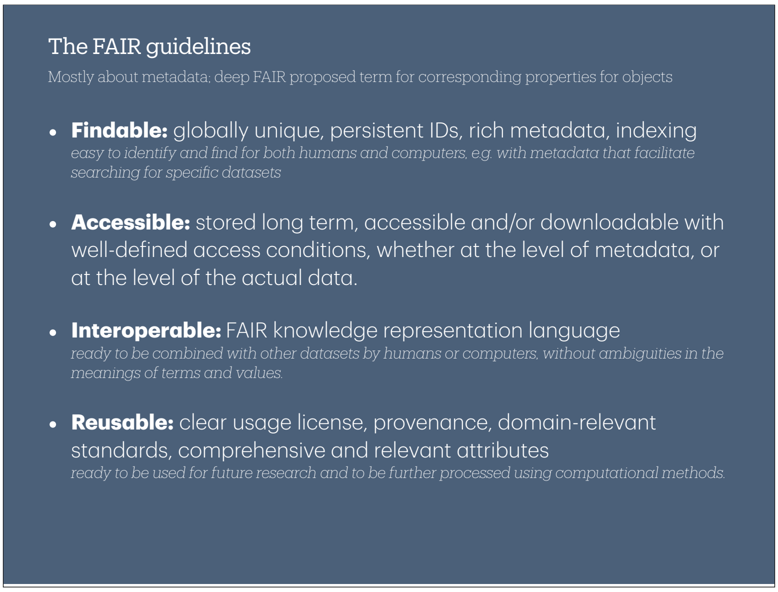
I was struggling with an image that would illustrate my frustration with some of the data in mathematics I come across (typically as some from outside of the area of mathematics), until I attempted to take a shower after checking-in on Sunday. The interface made little sense to me. There are red arrows on the right and blue arrows on the left. The main handle turns counter-clockwise. I could not get the water to be warmer than lukewarm. The smaller handle appeared to have little effect. I messed around with it for a while until, as a last resort, I turned the handle all the way into the worryingly blue zone, which unexpectedly but fortunately resulted in hot water.



The first time I saw the data on the right side of the slide, I had no idea what I was looking at. Of course, like with the tap, it is possible to figure it out (I presume), given enough time.

The FAIR guiding principles were published in 2016 and are an attempt to describe how usable data look like in very general terms.

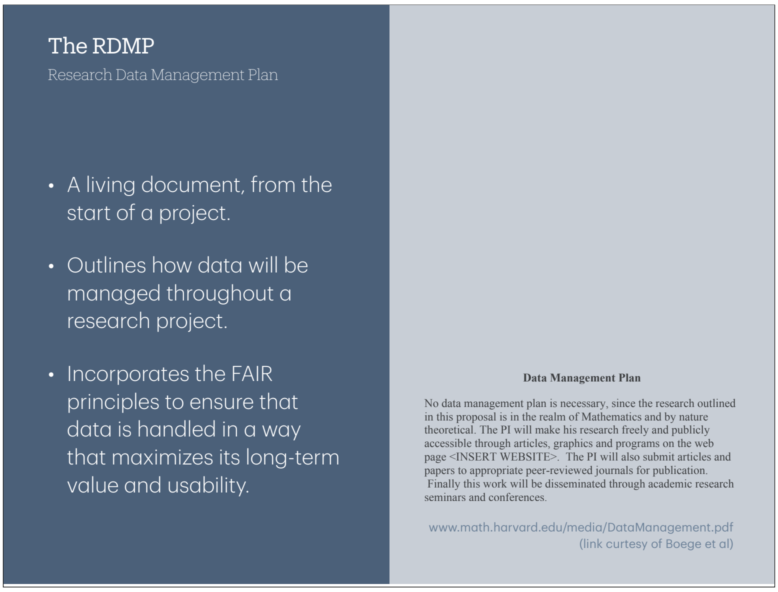
Open in "open access" refers to the removal of financial, legal and technical barriers to data, while accessibility in FAIR refers to the data being retrievable by humans and machines.



The FAIR guidelines are intentionally broad and somewhat vague; they are designed to provide communities with a flexible framework that can be further developed and adapted to specific needs. In practice, they focus primarily on metadata — covering aspects such as authorship, provenance, licensing, and descriptions of the dataset's contents.

Adopting FAIR principles can significantly improve the citability, visibility, and confirmability of datasets, making computational results more easily reproducible.

A brief note on interoperability: to the best of my knowledge, there is currently no standard knowledge representation language for mathematical data. While this means we don't yet have to worry about strict interoperability requirements, it is an area where the community should invest effort in the future.
If you think back to "everything is data" from earlier, a tricky question arises: can we state in general terms what metadata are sufficient to ensure reusability of data in mathematics?



A claim I've often heard in and about the field of mathematics is that mathematicians rarely produce data, and that the data they do produce requires little to no management. I've also frequently come across statements like "you can't license mathematical objects" and the belief that if data is posted on someone's website, it is automatically in the public domain and freely usable.

A taste of an RDMP questionnaire

What we may end up dealing it if the current trends continue

- *Project metadata*: title, ID, grant reference, PI names, institution, …
- *Expected questions*: RDMP author, data types and formats, how the data will be organized, secure storage and backup, documentation, volume of data, storage type, …
- *Often disregarded*: license.
- *Possibly does not apply* to mathematics:
  - Ethics approval, legal issues, IP, culturally sensitive issues.
  - Data confidentiality and sensitivity, access restrictions (incl. cost)
  - Non-digital data questions
  - Data destruction

Documenting changes in the approach to collecting data can be informative
clear documentation of techniques is instrumental to reproducibility, also minimizes the impact of onboarding new collaborators

Take-aways for data management

At this point in time and for people compiling datasets containing examples

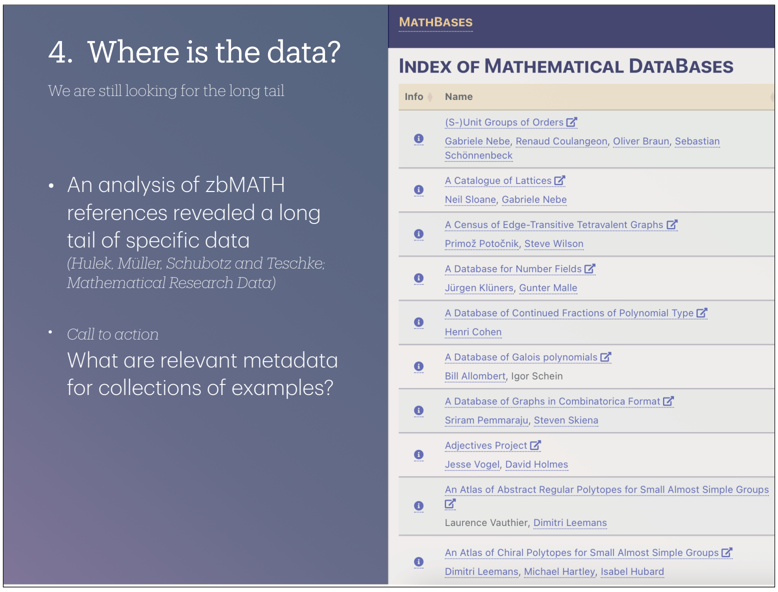
- More and better metadata and documentation
  though we are still waiting for a metadata standard

- *Archiving and preservation*: snapshots in a machine readable format on Zenodo or GitHub to ensure longevity

- *Reproducibility* for results of computation: record software info (version), attach code.

- An interesting problem up for community consideration:
  the meaning and provenance of mathematical data can require more complex mathematical data.

- A solution for recognition for research data beyond a journal publication is hard.



MathBases began as an effort to index and showcase mathematical databases. David has already given you a tour of MathBases, so I won't dwell on that here — except to note that the strong focus on combinatorics is partly due to the relative approachability of combinatorial data, and partly a reflection of my own bias.

MathBases indexes datasets that contain examples of objects of interest to research mathematicians. When I was compiling its precursor MathDB, I applied a similar criterion, but I often struggled to decide whether or not a dataset should be included — even when I understood its contents.

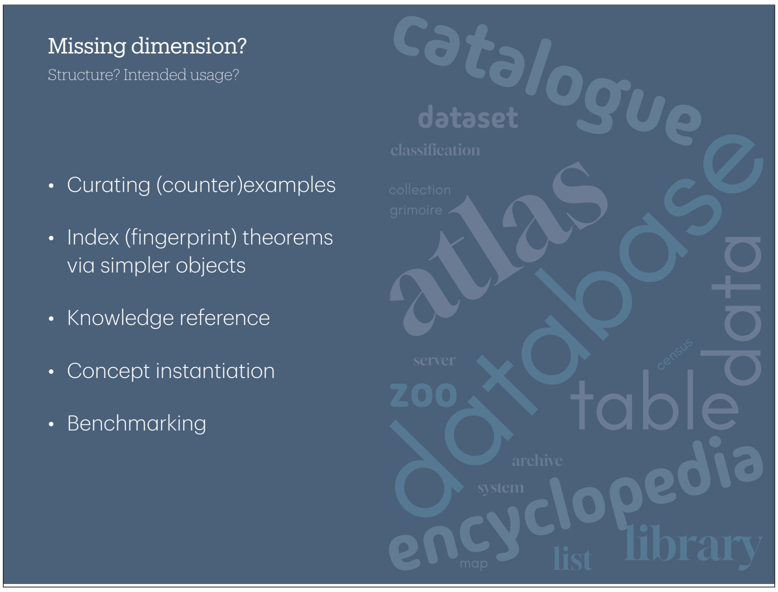
As a call to action (building on David's list of ways you can contribute), I encourage you to think about what metadata are most relevant for datasets containing examples of mathematical objects.

The diversity of math databases

Four easy dimensions

small *Atlas of Small Chiral Polytopes (56)*	-	large *Lists of finite lattices ($17 \cdot 10^9$)*
stored	-	generated on demand
House of Graphs - The Small Groups Library in GAP - nauty		
enumerated *The Foster census*	-	curated/collected *House of Graphs*
easy to obtain values *nauty*	-	a value corresponds to a paper *Parameters of Strongly Regular Graphs*

Room for more!

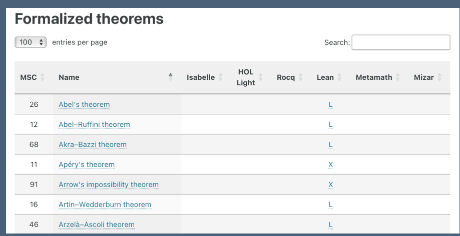


- Curation of examples: topological spaces, properties and theorems in π-base, graphs and invariants in the House of Graphs.
- **Index theorems: integer sequences (OEIS), Parameters of Strongly Regular Graphs.**
- Knowledge reference: definitions and properties of special functions in DLMF
- Instantiation: datasets in algebraic geometry (only one object, variety)
- Benchmarking: SuitSparse matrix dataset

A detour into the world of formalized mathematics: 1000+ theorems

Successor to Freek Wiedijk's 100 theorems

- 100 theorems (precursor): showcasing formalizations by keeping track of formalizations of the hundred greatest theorems (a *fixed* list)
- Indexing formalizations of a much longer (changing) list of theorems



Formalized theorems

100 entries per page   Search:

MSC	Name	Isabelle	HOL Light	Rocq	Lean	Metamath	Mizar
26	Abel's theorem				L		
12	Abel–Ruffini theorem				L		
68	Akra–Bazzi theorem				L		
11	Apéry's theorem				X		
91	Arrow's impossibility theorem				X		
16	Artin–Wedderburn theorem				L		
46	Arzelà–Ascoli theorem				L		

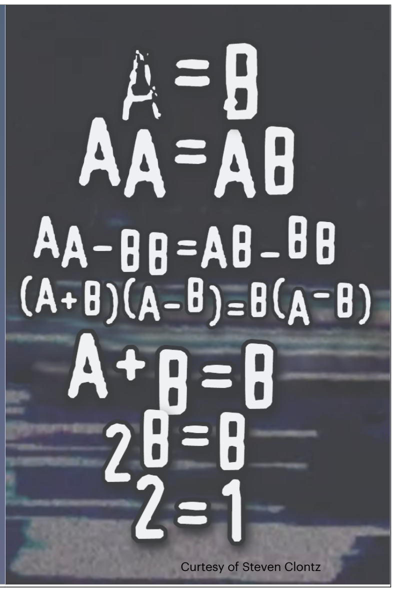
1000-plus.github.io

5. Trusting data

Beyond the standard checks

- How can we trust that a list of examples is complete (if applicable) and correct?
- Is the connection between theory and code sound?
- Are the results of computations correct?

No answers to the questions above, just an example: Lean-HoG

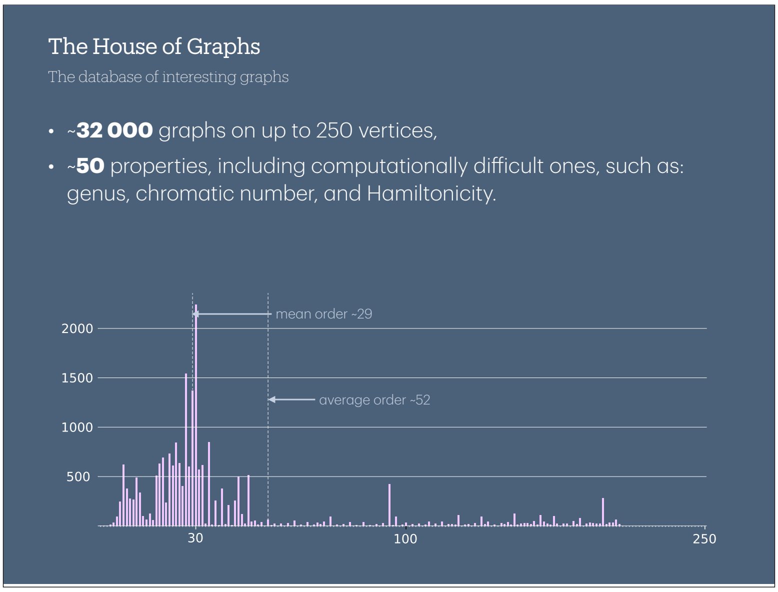


Curtesy of Steven Clontz

Some standard options to increase the level of trust

Techniques from engineering and mathematics

- Standard checks: format, type, consistency, uniqueness, ...
- *Testing*: software is run on a collection of test cases, the results are compared to reference results known to be true.
- *Redundancy*: several versions of software performing the same task are developed and executed independently, their results compared.
- Correctness of code or data is established by formal proofs.



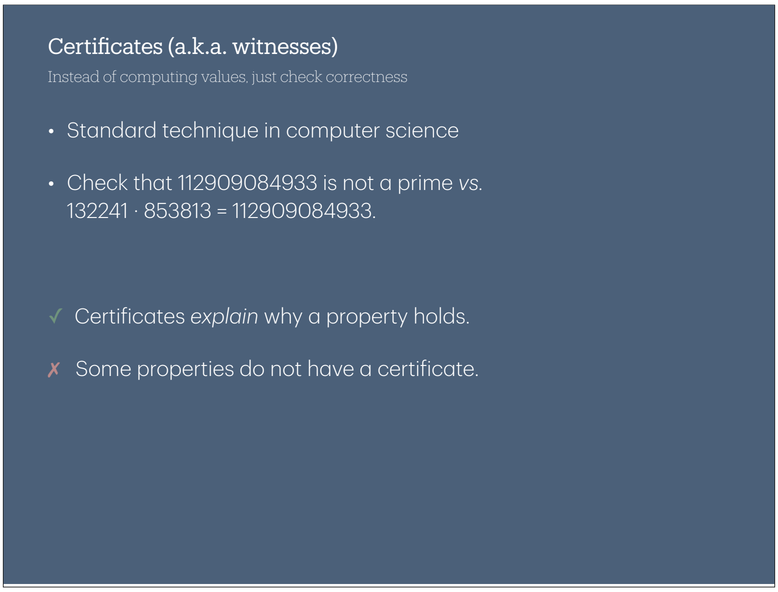
The combination of graph sizes and properties means that we can't just compute whichever way we want.

Design options

based on quantity and complexity of objects and properties

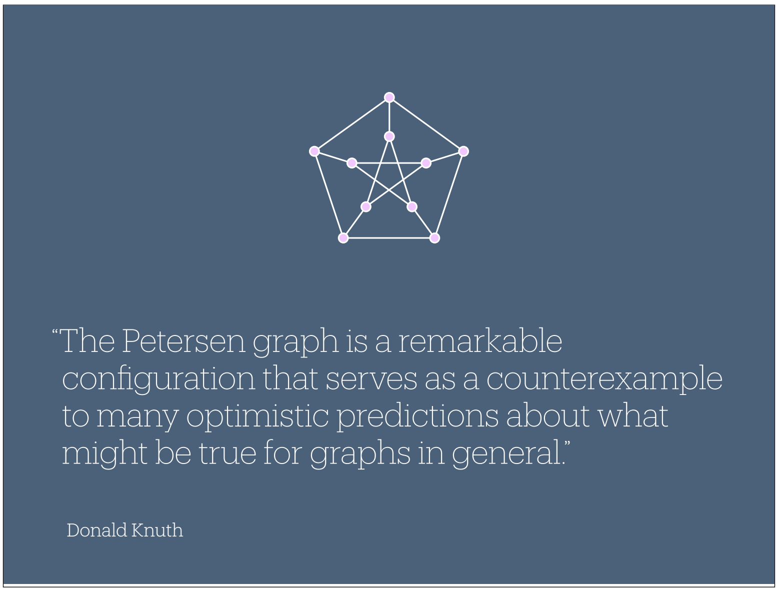
- Prove the properties of each example by hand.
- Implement algorithm(s) in the proof assistant
  (in the extreme case, implement a computer algebra system in a proof assistant)
- Encode as SAT, verify encoding to be correct, use a (trusted) solver, check the certificates provided by the solver.
- Use external software to compute properties *and their certificates*, use the proof assistant to check correctness.

1. For few objects and properties, simple.
2. Few properties, many objects, efficiently computable: can be difficult.
3. We used a combination of the last two.

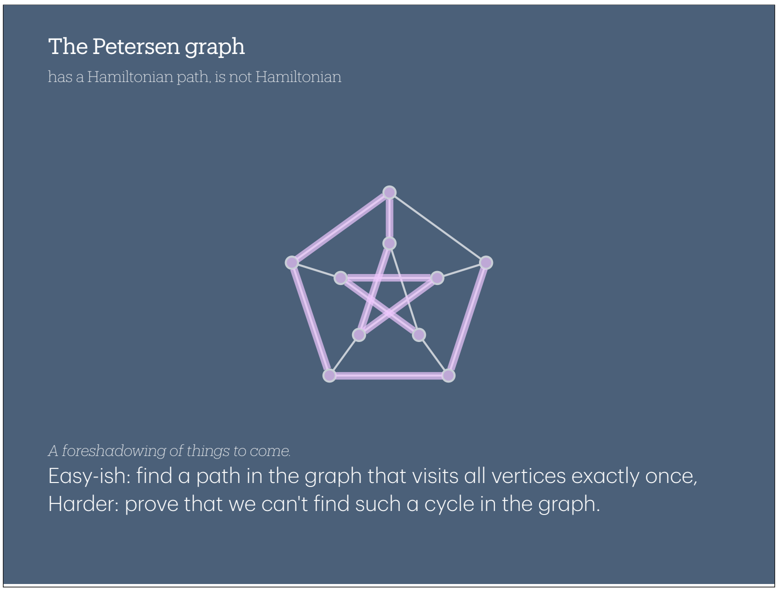


This works more broadly than you (might) think! (Used the idea for computation of election results).

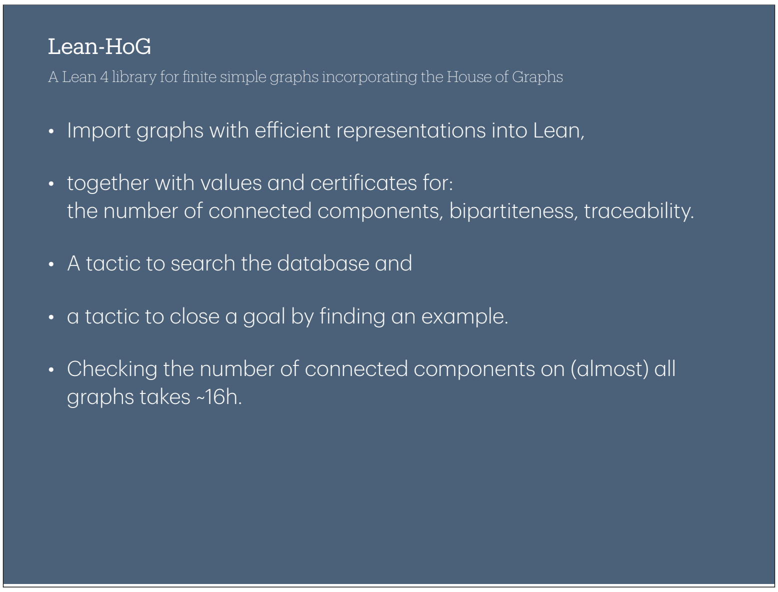
The proof assistant can check the correctness of the certificate. While the connection with the property "not prime" follows directly from the definition here, this is not the case in general; a further proof that the property follows from the certificate can be necessary.



Let's look at a random example of a graph from the House of Graphs.



The Petersen graph is also the smallest vertex-transitive graph that is not a Cayley graph.



Mathlib provides a basic, general-purpose formalization of simple graphs, but it was not suitable for our purposes. To address this, we implemented a small library for finite simple graphs, prioritizing efficiency over generality.

Early experiments showed that we could process a graph in time at most quadratic in the number of edges, and wherever possible, sub-quadratic in the number of vertices. Working naively with lists of vertices and edges — or with adjacency matrices — led almost immediately to quadratic (or worse) time complexity.

Some invariants, such as the number of edges, can be computed efficiently by the Lean kernel, provided an efficient graph representation. For other invariants — for example, testing bipartiteness via 2-coloring or detecting odd cycles — Lean can efficiently verify a certificate when supplied.

For the invariants (traceability), with certificates that only work in one direction, one strategy would be to complement them with heuristics wherever they work. For instance, detecting a disconnected graph is an easy way to rule out Hamiltonicity. Only when these simpler methods fail would we resort to SAT solving. However, we chose to take a more principled approach by using SAT for both directions.

Warning, implementation details ahead.

Getting graphs into Lean

Mathlib: graphs represented with a symmetric, irreflexive adjacency relation

- Given a coloring c, check that adjacent vertices have different colors:
  $\forall\, i\, j : \mathrm{Fin}\, n\,.\ \mathrm{Adj}\, i\, j \to (c i \neq c j)$,
  time complexity $\mathcal{O}(n^2)$, only $\mathcal{O}(|E|)$ when given a set of edges.

- Check whether a graph is regular:
  $\exists\, k : \mathbb{N}\,.\ \forall i : \mathrm{Fin}\, n\,.\ |\{j : \mathrm{Fin}\, n;\ \mathrm{Adj}\, i\, j\}| = k$
  time complexity $\mathcal{O}(n^2)$, only $\mathcal{O}(n)$ when given a neighborhood map.

Lean-HoG graph representations

A Lean 4 library for finite simple graphs incorporating the House of Graphs

All properties require efficient

- membership checking, and
- checking that something holds for every element of a set, i.e. vertices, edges.

Lean-HoG:

- RBSet and RBMap for all sets and maps,
- graphs represented via sets of edges (and an auxiliary neighborhood map, checked to be equivalent).

Certificates:
could just to regular certificates (no SAT), with paths etc; for the other side use heuristics whenever they work (disconnected graph for Hamiltonicity), only resort to SAT when all else fails; we took the more principled approach with SAT.

Connected components

```lean
/-- Verices u and v are connected if they are related by the equivalence
    relation generated by the adjacency relation. -/
def Graph.connected {G : Graph} : G.vertex → G.vertex → Prop := EqvGen G.adjacent
```

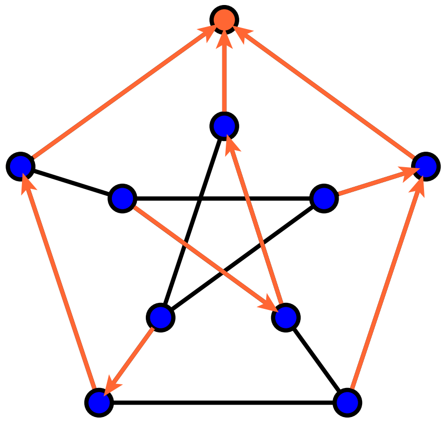
```lean
/-- Connected components of a graph, as a structure -/
class ConnectedComponents (G : Graph) : Type :=
  /-- Number of connected components -/
  val : Nat

  /-- The component of the given vertex -/
  component : G.vertex → Fin val

  /-- Components are inhabited -/
  componentInhabited : ∀ (i : Fin val), ∃ u, component u = i

  /-- The assignment of components coincides with connectedness -/
  correct : ∀ u v, component u = component v ↔ G.connected u v
```

Certificate



Certificate

```
/-- A certificate for connected components -/
class ConnectedComponentsCertificate (G : Graph) : Type :=
  -- Data
  val : Nat
  component : G.vertex → Fin val
  root : Fin val → G.vertex
  next : G.vertex → G.vertex
  distToRoot : G.vertex → Nat

  -- Properties
  componentEdge : G.edgeSet.all (fun e => component (G.fst e) = component (G.snd e)) =
  rootCorrect : ∀ i, component (root i) = i
  distRootZero : ∀ (i : Fin val), distToRoot (root i) = 0
  distZeroRoot : ∀ (v : G.vertex), distToRoot v = 0 → v = root (component v)
  nextRoot : ∀ i, next (root i) = root i
  nextAdjacent : ∀ v, 0 < distToRoot v → G.adjacent v (next v)
  distNext : ∀ v, 0 < distToRoot v → distToRoot (next v) < distToRoot v


/-- From a components certificate we can derive the connected components -/
instance {G : Graph} [C : ConnectedComponentsCertificate G] : ConnectedComponents G :=
```

Load the certificate

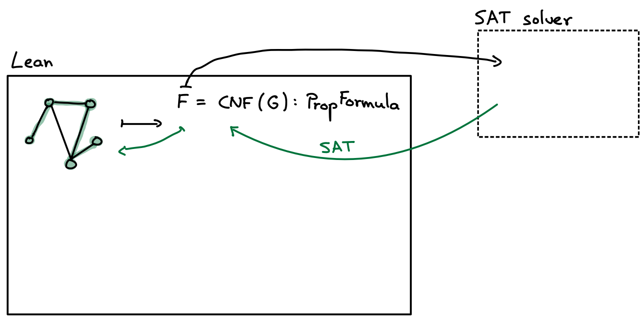
```
/--
  JSON representation of connected components certificate.
-/
structure ConnectedComponentsData : Type where
  val : Nat
  component : Array (Nat × Nat)
  root : Array (Nat × Nat)
  next : Array (Nat × Nat)
  distToRoot : Array (Nat × Nat)

deriving Lean.FromJson
```

```
/-- Build a connected components certificate expression from the data. -/
def buildCert (G : Q(Graph)) : ConnectedComponentsData → Q(ConnectedComponentsCertificate $G) :=
```

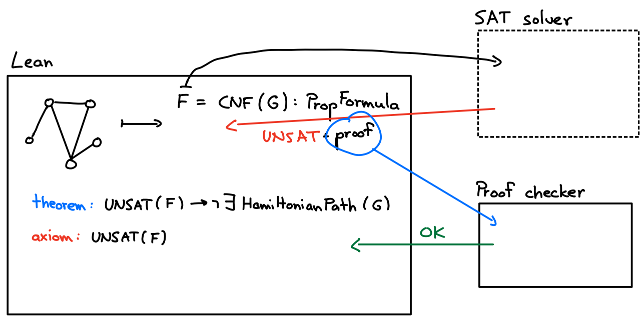
Hamiltonian Paths

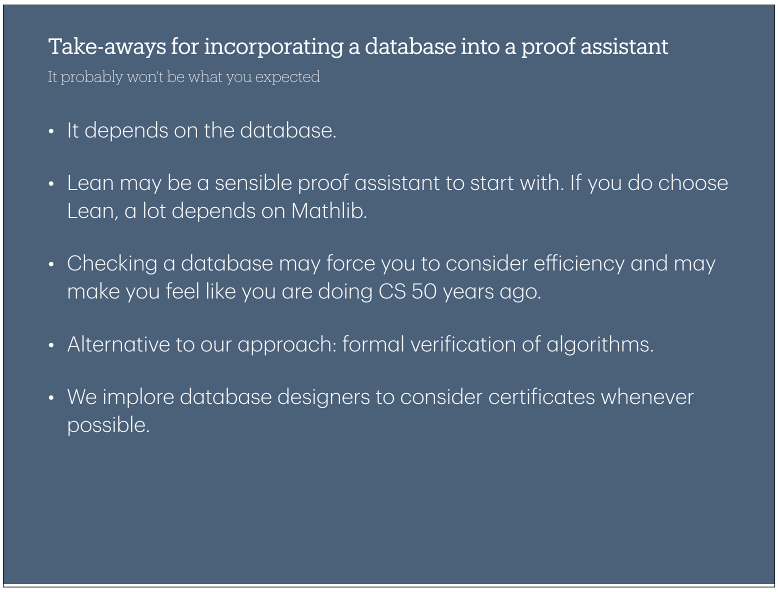
NP-hard, in general no easy certificate for non-existence.
Use a SAT solver.



Hamiltonian Paths

NP-hard, in general no easy certificate for non-existence.
Use a SAT solver.





We found it particularly advantageous to minimize the amount of computation performed directly by Lean, especially in situations involving meta-programming, where Lean metaprograms construct proofs for each value.

It would be possible to implement most of the properties of graphs in HoG. In some cases, however, we did not see a clear way out. For instance, computing the maximum or minimum eigenvalues of the adjacency matrix would require not only a standard format for algebraic numbers and a trusted, efficient computation engine for them, but also further considerations if we wanted to reason about extremality.

Thank you!

Recap of what we have been up to

- **MathBases**
  Adam Towsley, Ben Spitz, David Roe, David Lowry-Duda, Benjamin Hutz, Edgar Costa, KB
- **Lean-HoG**
  Jure Taslak, Gauvain Devillez, KB, Andrej Bauer
- **1000+ theorems**
  Freek Wiedijk, Floris van Doorn, KB; editors for each system
- **MathDataHub**
  Tom Wiesing, KB

Slides (hopefully) and references at katja.not.si, as promised

I feel like it is some kind of a rite of passage when you finally get to advertise a job. The PI on the project is Andrej Bauer (foundations of mathematics and logic, constructive and computable mathematics, homotopy type theory, mathematical foundations of programming languages, exact scientific computation, also, a very cool colleague). I will be working on the same project.
("can you lure a postdoc to Lj that would be more or less a copy of you")

Conference on Intelligent Computer Mathematics

CICM 2026 @Ljubljana

Published proceedings,
peer-reviewed contributed papers,
database descriptions welcome

summer/September (TBD)
Hope to see you there!