Disclaimers

Since it is an itch of mine, will mostly talk about how we might use alignments to make data more useful.

Hopefully, the current (small) momentum behind data might give some energy to the alignments project.

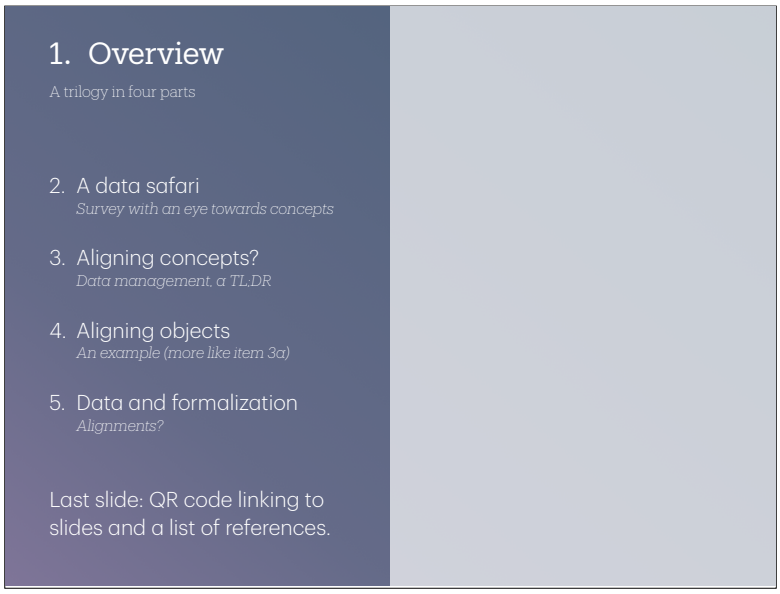
1. Overview

A trilogy in four parts

2. A data safari
   Survey with an eye towards concepts
3. Aligning concepts?
   Data management, a TL;DR
4. Aligning objects
   An example (more like item 3a)
5. Data and formalization
   Alignments?

Last slide: QR code linking to slides and a list of references.

An incomplete list of my influences



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.

- To start, we'll spend some time on a safari of mathematical data.
- We will look at a use case for alignments: annotating data, making it more useful.
- We will (perhaps cheekily) stretch the notion of alignments to objects.
- Finally, we will consider an example of incorporating data with formalization.

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.

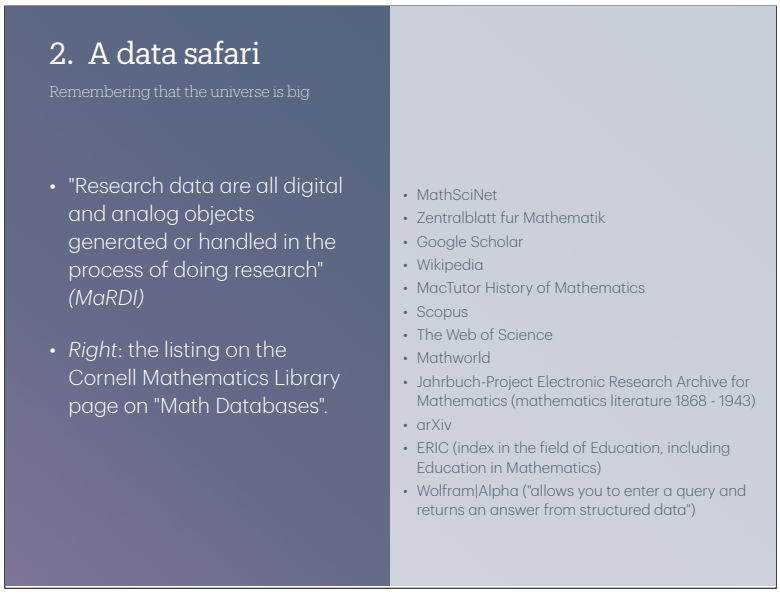
Warming up with a poll

Thank you for being patient with me

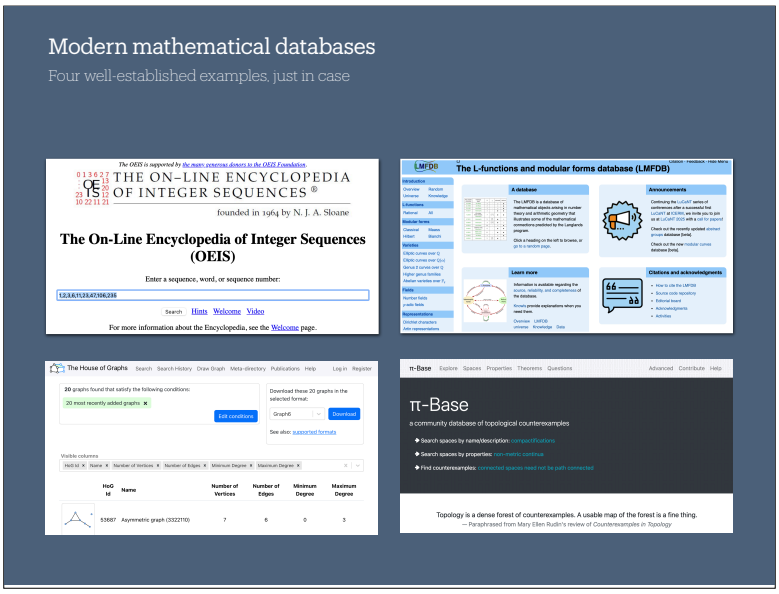
Scan the QR code on the right or go to `menti.com` and enter the code `3882 2786`

When answering, consider

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



"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.



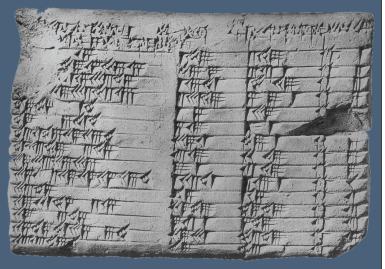
I mostly think about data that broadly looks like the databases above (possibly with less infrastructure): collections of (counter)examples

Some in audience have probably seen OEIS, LMFDB. Most datasets are not so complex (in the infrastructure sense).

Cuneiform and the Greeks

Data probably goes as far back as math

- Earliest datable table containing mathematical computations: length measurements, corresponding areas (Sumer, c. 2600 BCE).
- *picture*
  A list of Pythagorean triples (Babylon, c. 1800 BCE).



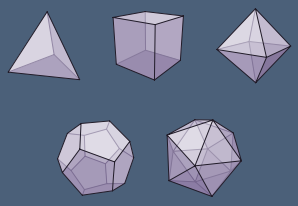
Plimpton 322 clay tablet; Bill Casselman

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]

Cuneiform and the Greeks

Data probably goes as far back as math

- Earliest datable table containing mathematical computations: length measurements, corresponding areas (Sumer, c. 2600 BCE).
- *picture*
  The list of the Platonic solids (a.k.a. convex regular polyhedra; Theaetetus of Athens, c. 417 - c. 369 BCE).



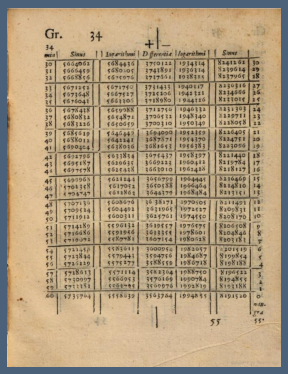
Platonic solids; Wikipedia user Drummyfish

Platonic solids: constructed by regular polygons as faces, with the same number of them at each vertex, convex.

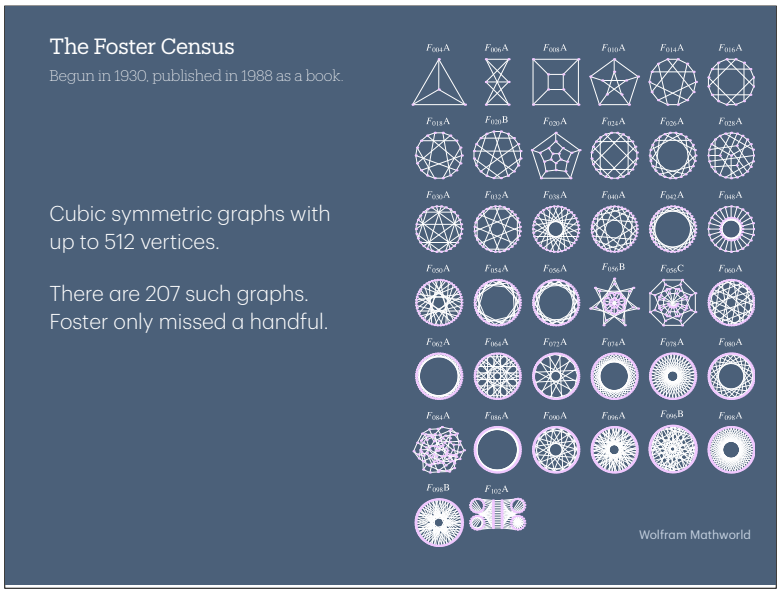
Results of computations

Persistence: data outlasting the computation before computers

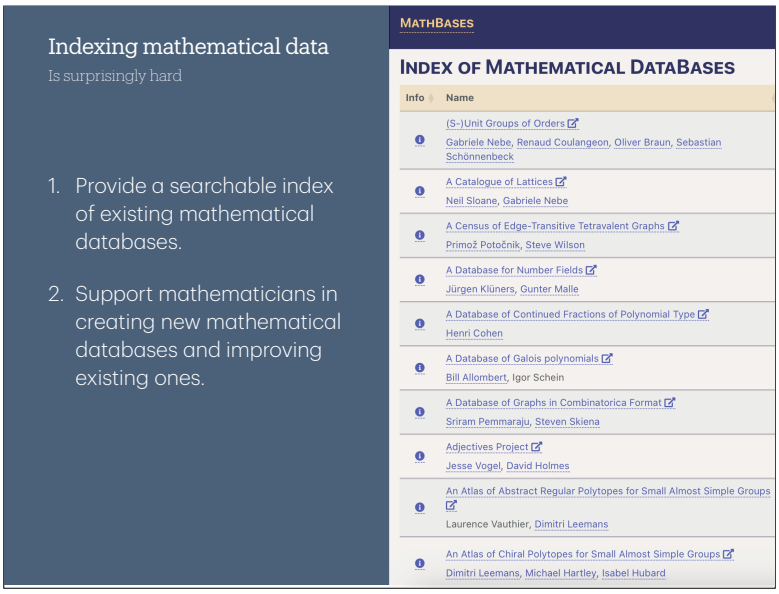
- *picture*
  Tables of trigonometric, logarithmic, and exponential functions *(Napier's* Mirifici logarithmorum, *trig and log trig data for 34 degrees)*
- Math Tables Project: human computers constructed tables of mathematical functions (1938 - 1946; for hand computation).



John Napier, 1614

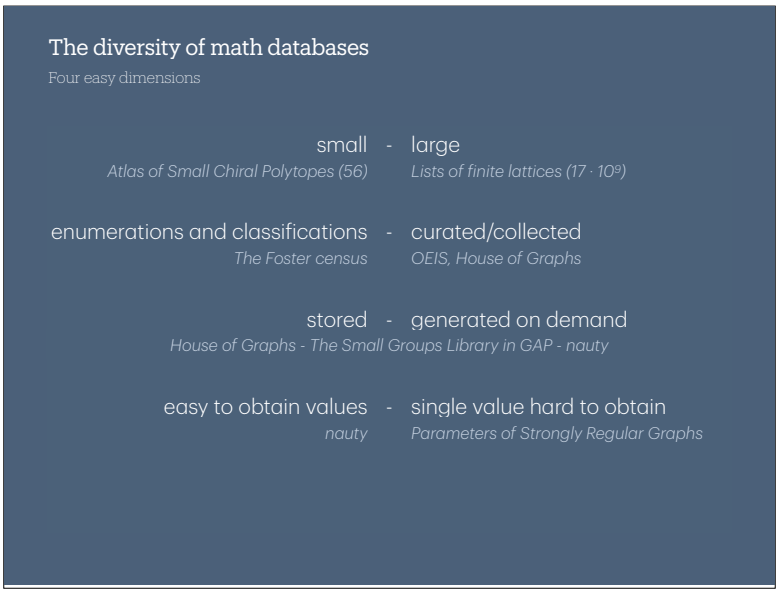


Symmetric graph: every ordered pair of adjacent vertices (an arc) can be mapped to any other such pair.

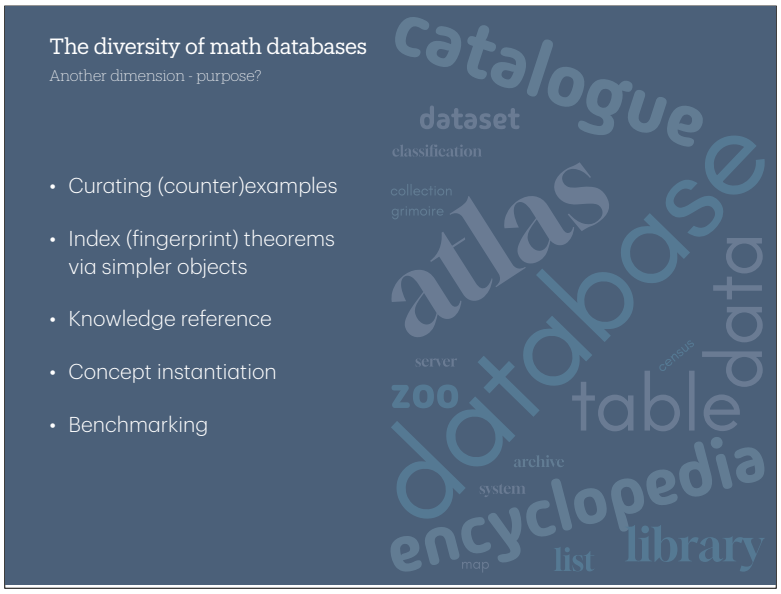


MathBases began as an effort to map what is out there as well as to index and showcase mathematical databases. There is a bias towards number theory and combinatorics, which reflects the editors' familiarity with these areas.

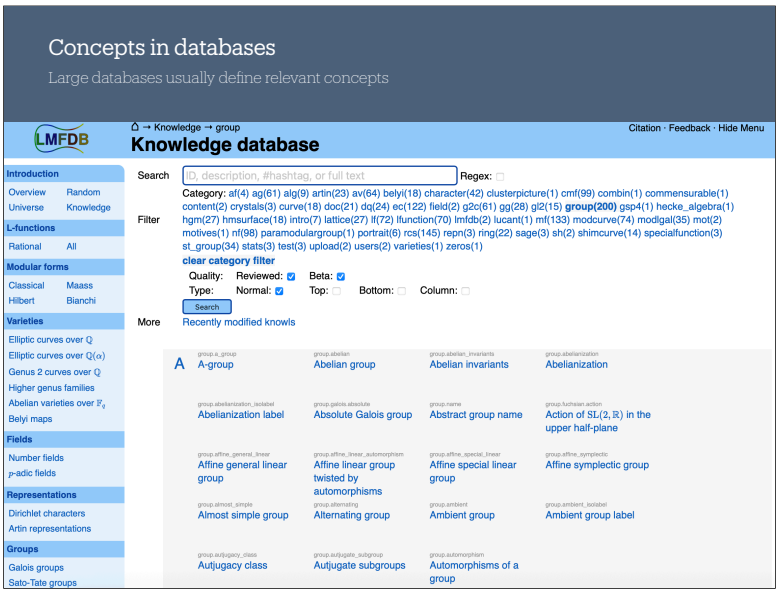
MathBases has gained surprising visibility, we could promote use of standardized vocabulary.



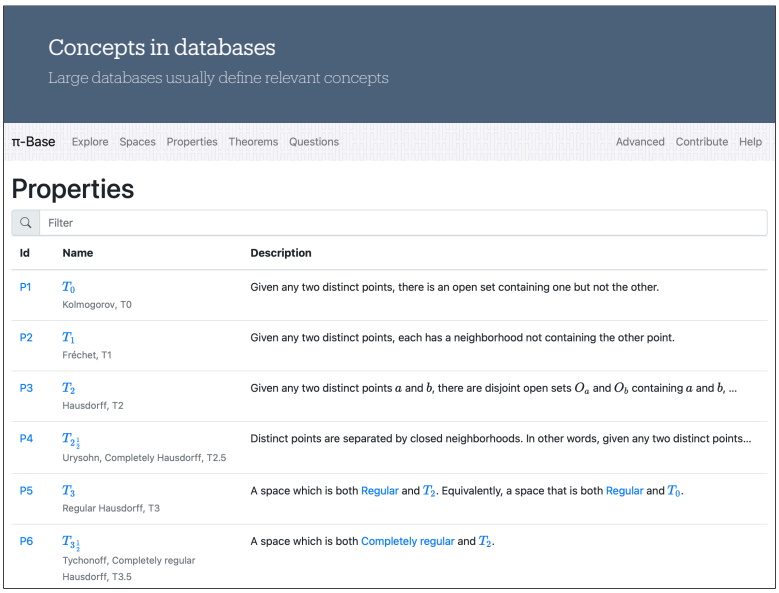
Despite the restriction we have imposed, there is great diversity.



- 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

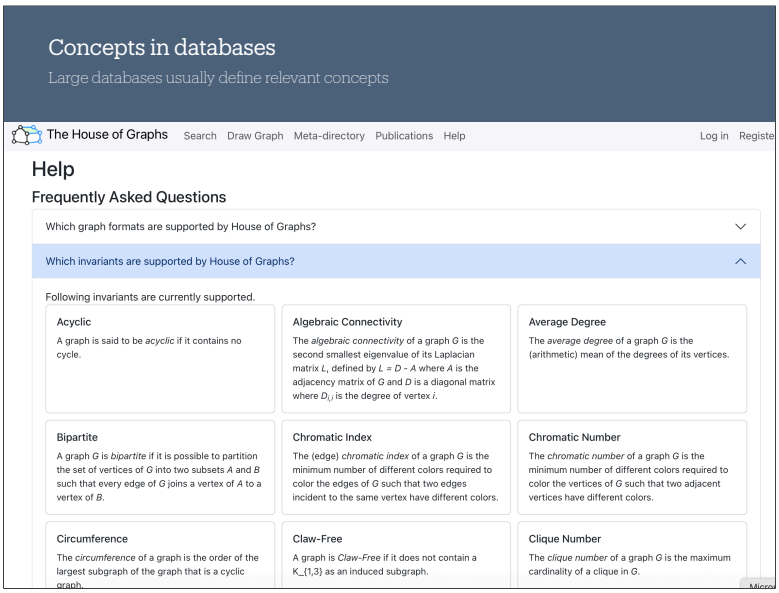


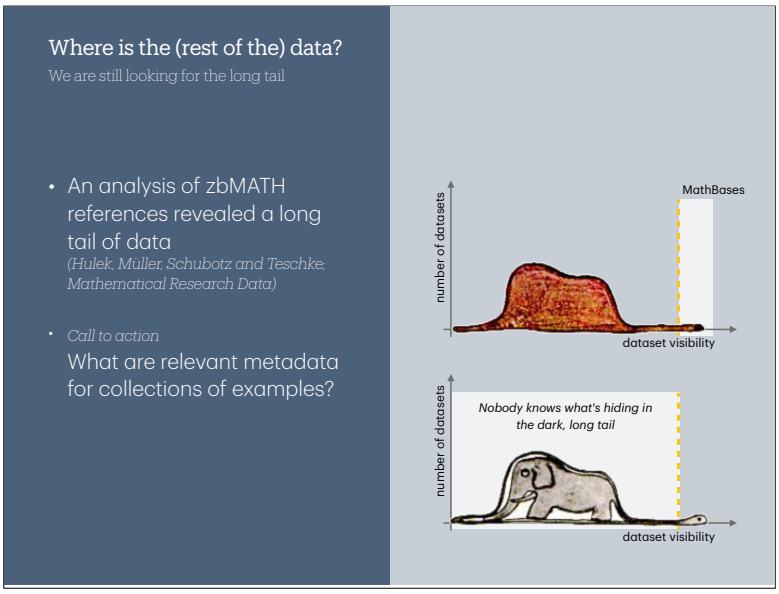
The larger databases are pretty good at defining their concepts.


Concepts in databases
Large databases usually define relevant concepts
π-Base
Explore
Spaces
Properties
Theorems
Questions
Advanced
Contribute
Help
Properties
Filter
Id
Name
Description
P1
T_0
Kolmogorov, T0
Given any two distinct points, there is an open set containing one but not the other.
P2
T_1
Fréchet, T1
Given any two distinct points, each has a neighborhood not containing the other point.
P3
T_2
Hausdorff, T2
Given any two distinct points a and b, there are disjoint open sets O_a and O_b containing a and b, ...
P4
$T_{2\frac{1}{2}}$
Urysohn, Completely Hausdorff, T2.5
Distinct points are separated by closed neighborhoods. In other words, given any two distinct points...
P5
T_3
Regular Hausdorff, T3
A space which is both Regular and $T_{\frac{1}{2}}$. Equivalently, a space that is both Regular and T_0.
P6
$T_{3\frac{1}{2}}$
Tychonoff, Completely regular Hausdorff, T3.5
A space which is both Completely regular and T_2.

Concepts in databases

Large databases usually define relevant concepts





There are currently 128 entries in MathBases, a few more waiting to be entered. Is this all?

What you see on MathBases is probably just the tip of the iceberg (even if we're trying to make this not the case).

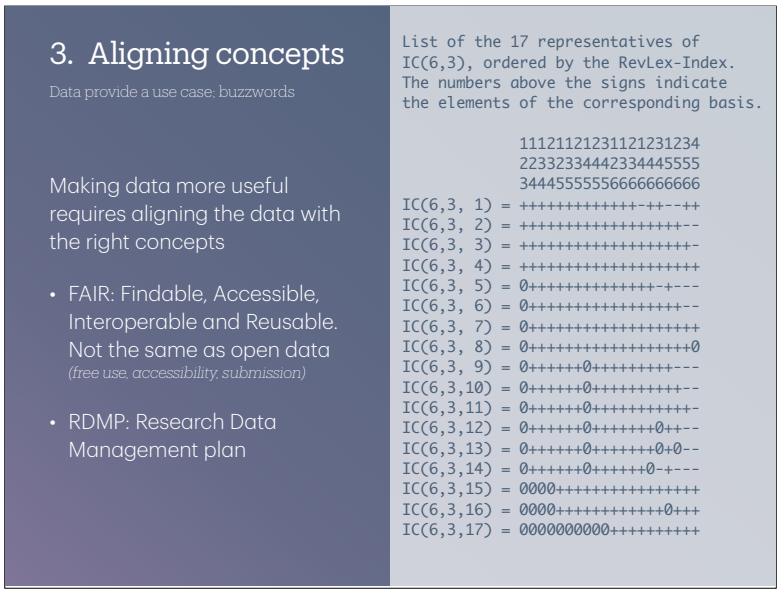
Questions?

I want to illustrate my frustration with some of the data in mathematics.

Let us consider the specimen of industrial design pictured on the right.

There are red arrows on the right and blue arrows on the left. The main handle turns counter-clockwise and seemed to correlate somewhat with water output. The smaller handle appeared to have little effect. I messed around with it for a while and could not get the water to be warmer than lukewarm.

As a last resort, I turned the handle all the way into the worryingly blue zone, which unexpectedly but fortunately resulted in hot water.

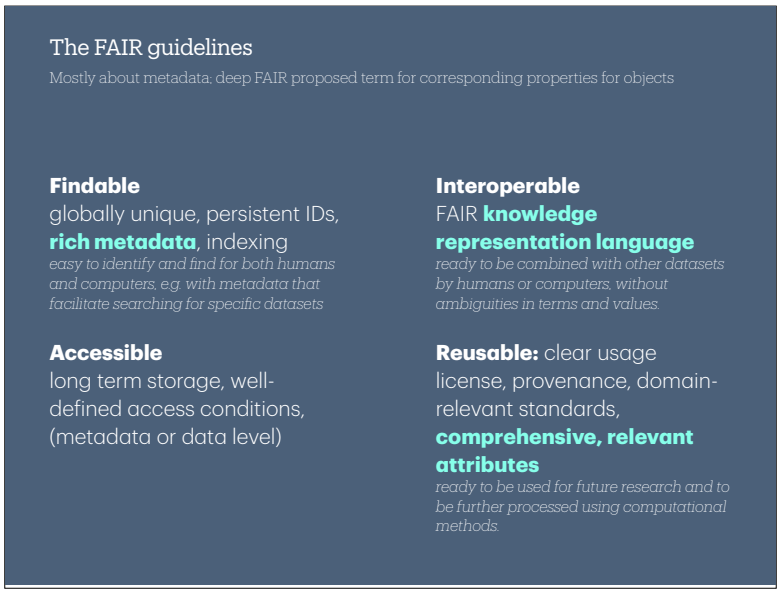


The first time I saw the data on the right side of the slide, I had no idea what I was looking at. As with the tap, it is possible to figure it out (I presume), given enough time.

The FAIR guiding principles were published in 2016 and are an attempt to describe, in very general terms, how usable data look like. Importantly some of what they recommend is related to alignments.

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 RDMP is another buzzword that appears in relation to research data. We won't spend too much time on it, but I thought it would be helpful to at least mention it.

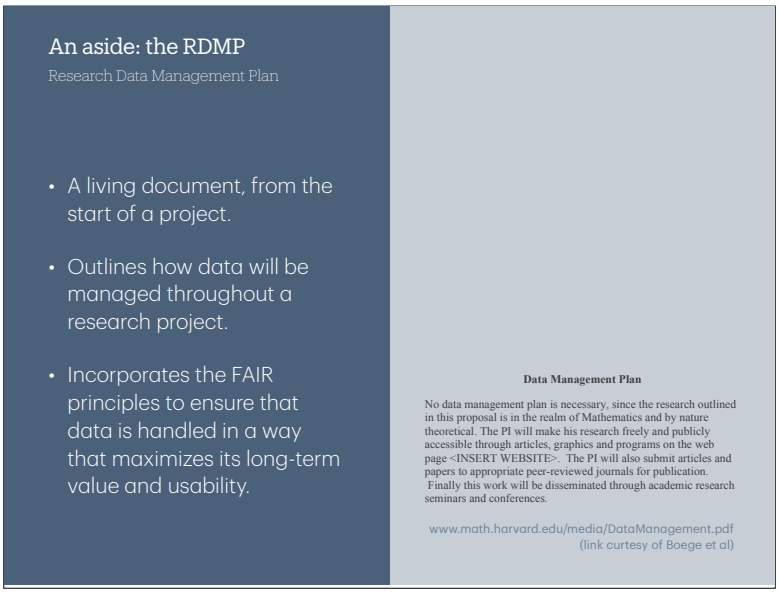


The FAIR guidelines are intentionally broad and somewhat vague; designed to provide communities with a flexible framework to be further developed and adapted. They focus primarily on metadata — covering aspects such as authorship, provenance, licensing, and descriptions of the dataset's contents.

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

In particular, the principles stress the importance of accurately and understandably describing the data so that it is findable, interoperable, and reusable.

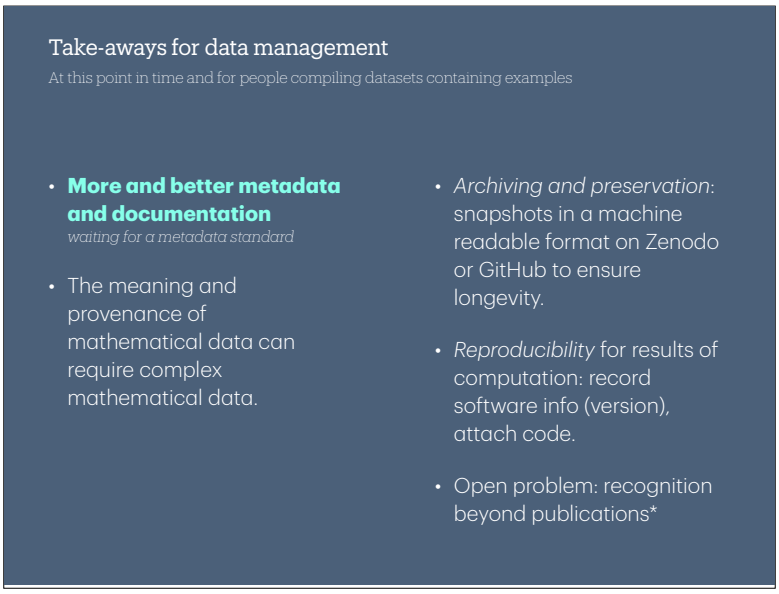
A brief note on interoperability: to the best of my knowledge, there is currently no standard knowledge representation language for mathematical data. While this means we don't yet have to worry about strict interoperability requirements, it is an area where the community should invest effort in the future.



The other buzzword you might encounter is the RDMP, a document outlining the plan for managing the data. It can incorporate the FAIR principles and can be required by funding agencies and/or institutions.

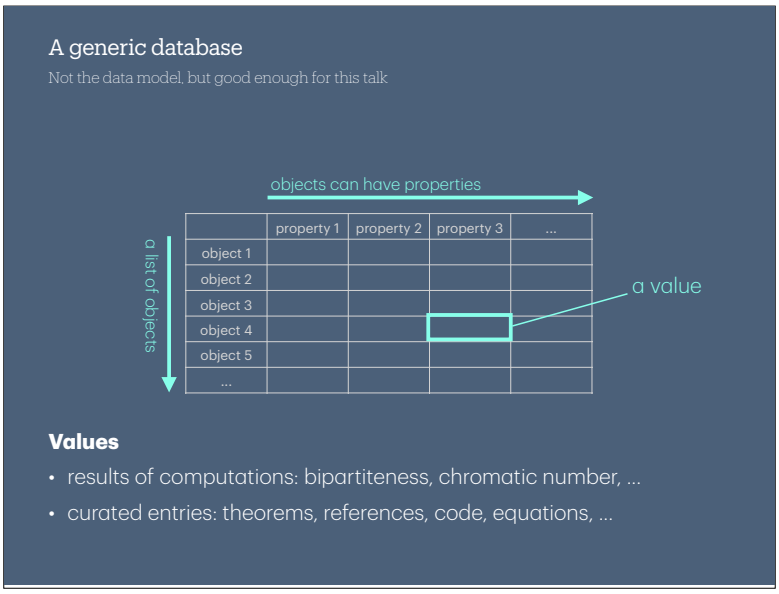
I'm primarily bringing it up because of the example on the right, which echoes a claim I've often heard in and about the field of mathematics. It is that mathematicians rarely produce data, and that the data they do produce requires little to no management. I've also frequently come across statements 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.

This attitude is an obstacle (of a social nature) that stands in the way of better data in math.

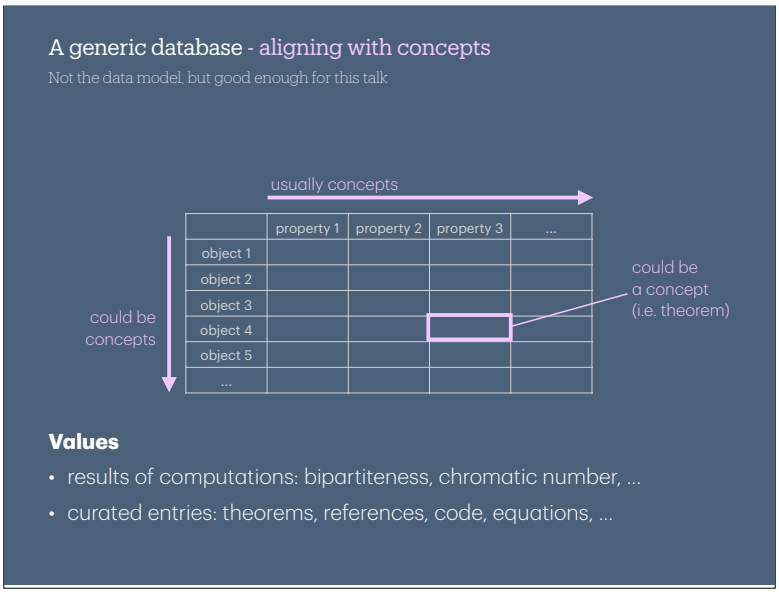


There are a few things dataset authors can do to improve the usability, and documenting the contents and structure is an important part of that.

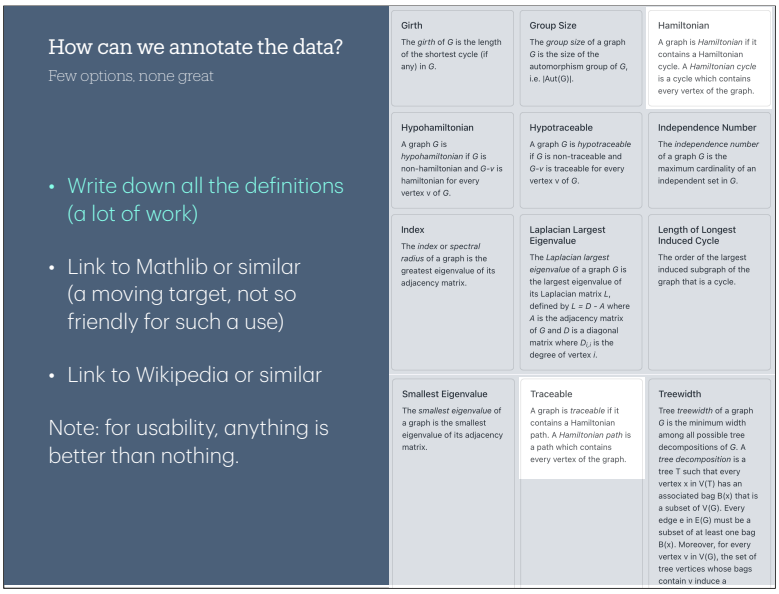
You might remember that one of the stated goals of MathBases is to help mathematicians compile new databases and improve existing ones. Because it has gained some visibility (which came as a little bit of a surprise to me), we are in a position to recommend good practices. What should we recommend?



For now, let's take a simple table as a model for datasets - it is going to be good enough.



Ideally, all of these should be nicely annotated: properties, but also some of the objects and possibly some of the object-property values.

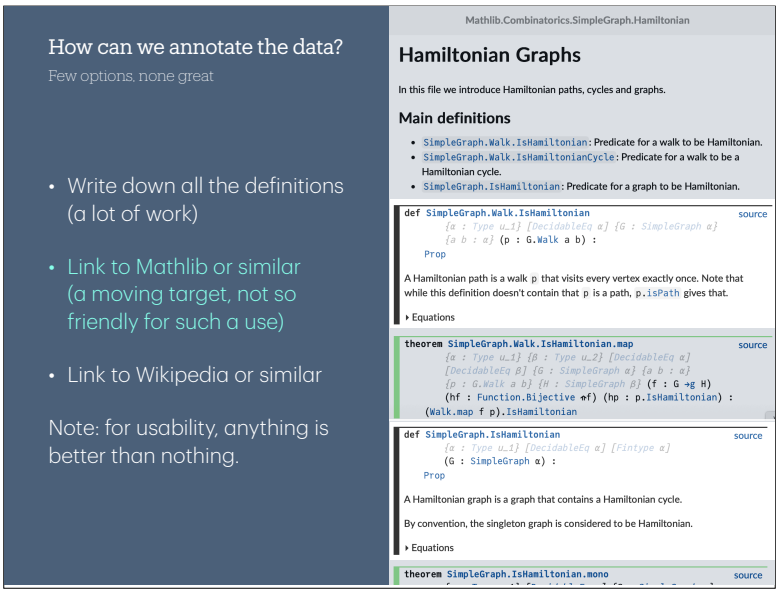


Let's look at defining one of the properties.

Suppose we take a dataset containing graphs; one of the properties could be whether a graph is Hamiltonian.

Big databases usually have the contributors to do the work of writing down all the definitions.

For smaller projects, property definitions are usually not defined (except where there is potential for ambiguity) - they are expected to be known by the members of the community, the main users. Unfortunately, this makes the data less useful outside of the area of origin.



We could recommend linking to one of the formal mathematics libraries; not super readable, potentially too precise.

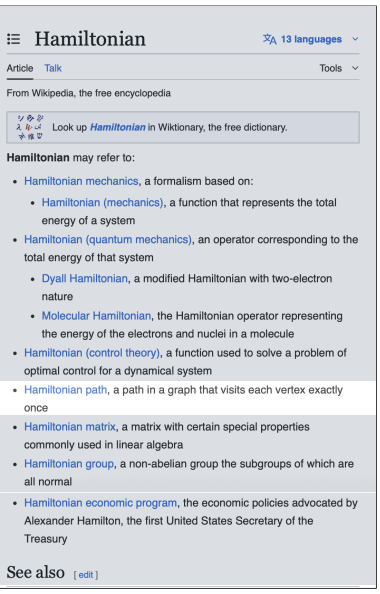
About two years ago I asked my office mate at the time to mark up definitions in the Agda Unimath library with references to other resources. It took repeated convincing that it is ok for these references to have subtle differences in the definition of the concept before the annotations took off. I suspect part of it was that linking to accessible resources made it easier for undergraduate students to join the formalization project. Linking to a formal library from a database goes in the other direction but might present a similar (or worse) problem.

How can we annotate the data?

Few options, none great

- Write down all the definitions (a lot of work)
- Link to Mathlib or similar (a moving target, not so friendly for such a use)
- Link to Wikipedia or similar

Note: for usability, anything is better than nothing.



Hamiltonian

13 languages

Article Talk Tools

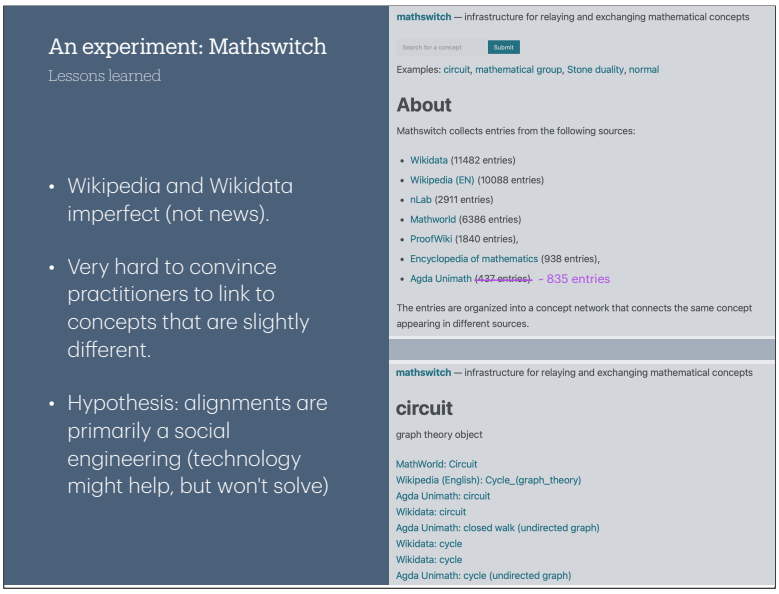
From Wikipedia, the free encyclopedia

Look up ***Hamiltonian*** in Wiktionary, the free dictionary.

Hamiltonian may refer to:

- Hamiltonian mechanics, a formalism based on:
  - Hamiltonian (mechanics), a function that represents the total energy of a system
- Hamiltonian (quantum mechanics), an operator corresponding to the total energy of that system
  - Dyall Hamiltonian, a modified Hamiltonian with two-electron nature
  - Molecular Hamiltonian, the Hamiltonian operator representing the energy of the electrons and nuclei in a molecule
- Hamiltonian (control theory), a function used to solve a problem of optimal control for a dynamical system
- Hamiltonian path, a path in a graph that visits each vertex exactly once
- Hamiltonian matrix, a matrix with certain special properties commonly used in linear algebra
- Hamiltonian group, a non-abelian group the subgroups of which are all normal
- Hamiltonian economic program, the economic policies advocated by Alexander Hamilton, the first United States Secretary of the Treasury

See also [edit]



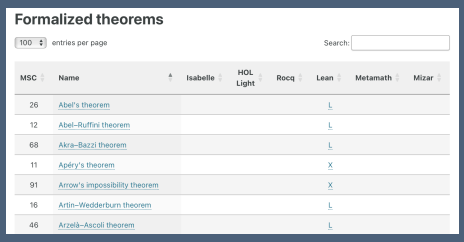
An experiment grew out of this frustration after the Dagstuhl workshop two years ago.

- Ask libraries to link to *anything*; link in place, close to the concepts (for maintenance reasons).
- Collect links and organize them into a network to provide value.
- That alone seemed to be enough to talk friends into it, but not enough to convince Mathlib, for example.

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

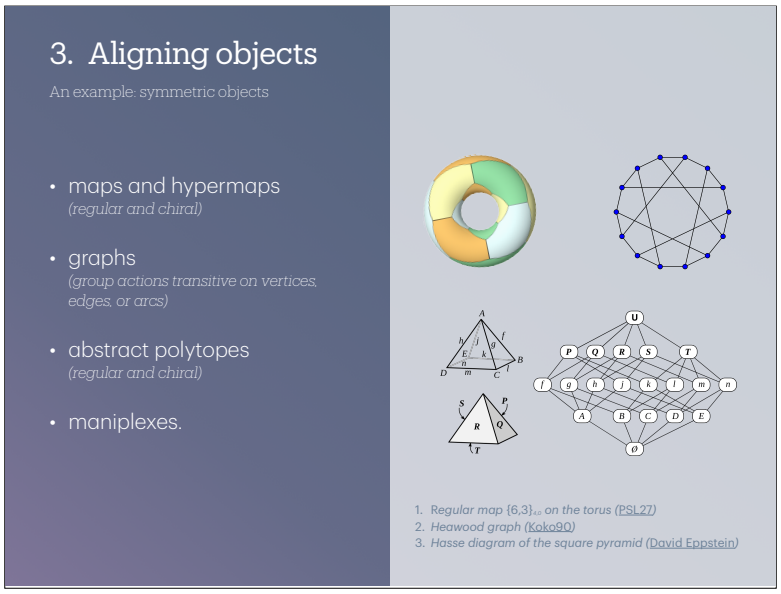


Formalized theorems

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		

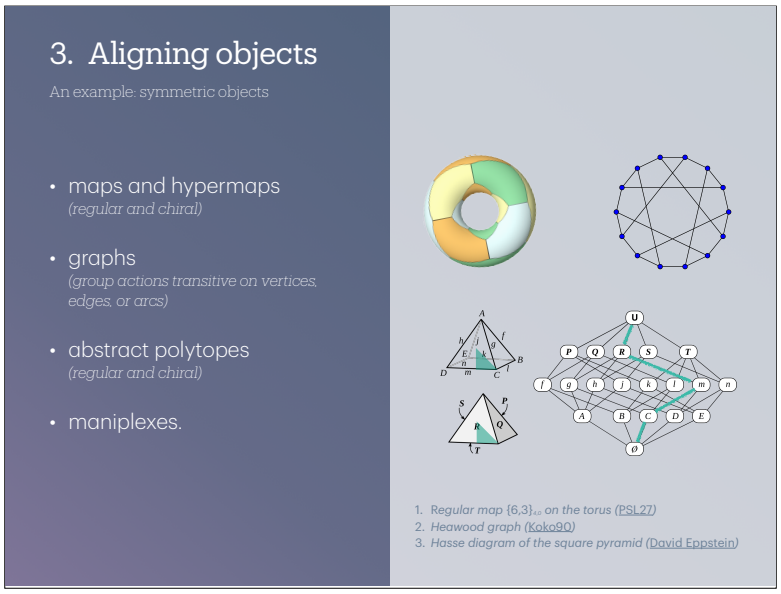
1000-plus.github.io



Context: finite (combinatorial), symmetic (~transitive automorphism group)

Regular map: a decomposition of a two-dimensional manifold (such as a sphere, torus, or real projective plane) into topological disks such that every flag (an incident vertex-edge-face triple) can be transformed into any other flag by a symmetry of the decomposition.

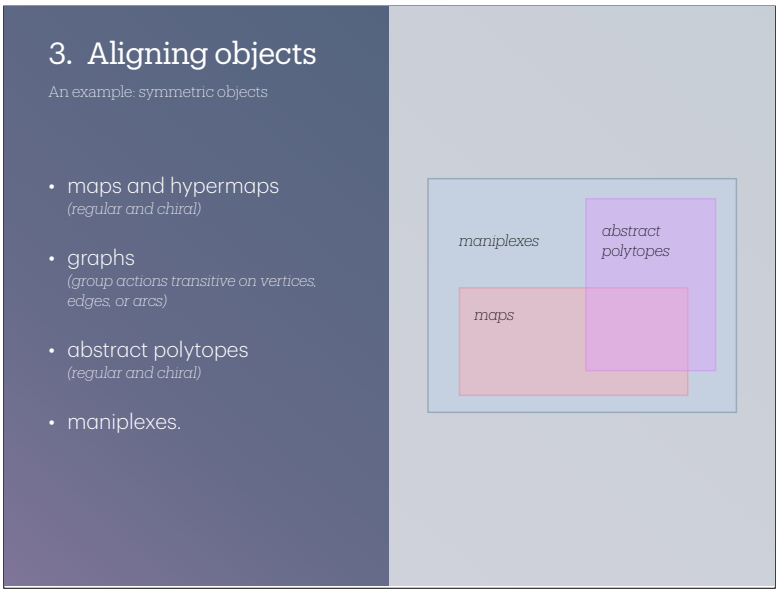
Abstract polytope: partially ordered set which captures the dyadic property of a traditional polytope without specifying purely geometric properties such as points and lines. (1) It has just one least face and one greatest face. (2) All flags contain the same number of faces. (3) It is strongly connected. (4) If the ranks of two faces a > b differ by 2, then there are exactly 2 faces that lie strictly between a and b.
- Diamond condition: for i = 0, . . . , n – 1 there is exactly one flag that differs from a given flag in the i-face
- Rank: {–1, . . . , n}; 0, 1 and n – 1 are vertices, edges and facets



Schläfli symbol: a notation of the form { p, q, r, ... } that defines regular polytopes and tessellations.

Examples:
{ p } - p-sided regular convex polygon
{ p, q } - q regular p-sided polygon faces around each vertex (cube: { 4, 3 })
{ p, q, r } - r { p, q } regular polyhedral cells around each edge (tesseract: { 4, 3, 3 } has 3 cubes around an edge)

Top left: 16 faces, 32 vertices and 48 edges, with a symmetry group of order 192



Maps and abstract polytopes: group representation (Coxeter generators), Schläfli symbol;

Chiral: rotational, but no reflectional symmetry

Goal: link objects in the intersection, but also record other relations, such as maps and their skeleton graphs.

Maniplexes

A generalization of maps and polytopes

- Can be represented as edge-colored graphs
- An i-colored edge represents moving to an adjacent flag by changing the i-face.
- s_i - the transposition corresponding to swapping the vertices along i-edges
- i-faces are cycles of alternating colors

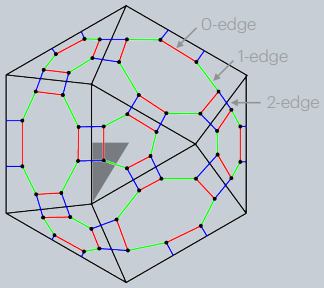


Diagram from Cunningham, Del Rio-Francos, Hubard, Toledo, Symmetry Type Graphs of Abstract Polytopes and Maniplexes

Representations

Graph canonical labellings to the rescue

- Permutation representation
- Coxeter generator relations
- Checking isomorphism hard
- Graphs: canonical labelling (label vertices so that checking isomorfism becomes checking equality)
- Use maniplexes to relate objects

```
s0 := (4,5);;
s1 := (3,4);;
s2 := (2,3);;
s3 := (1,2)(6,7);;
poly := Group([s0,s1,s2,s3]);;
```

```
Regular 3-polytope with group of
order 24   SD  Type [ 3, 3 ]
[ A.1^2, A.2^2, A.3^2, (A.1 * A.3)^2,
  (A.2 * A.1)^3, (A.3 * A.2)^3 ]
```

SD = self dual

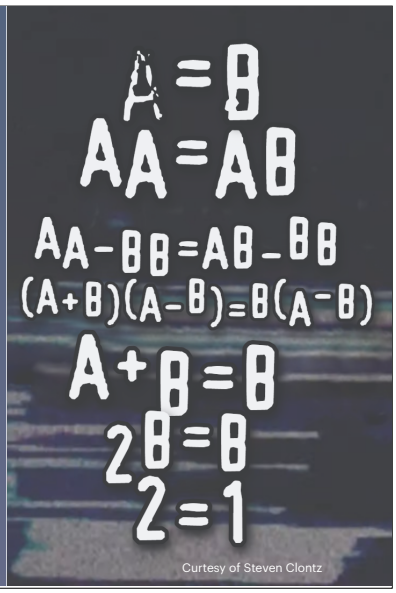
Questions?

5. Formalizing data

No answers, just an example: Lean-HoG

Formalizing a property to verify the data aligns the formalization with a column

- Source of examples for formalized mathematics
- Increased trust in the data



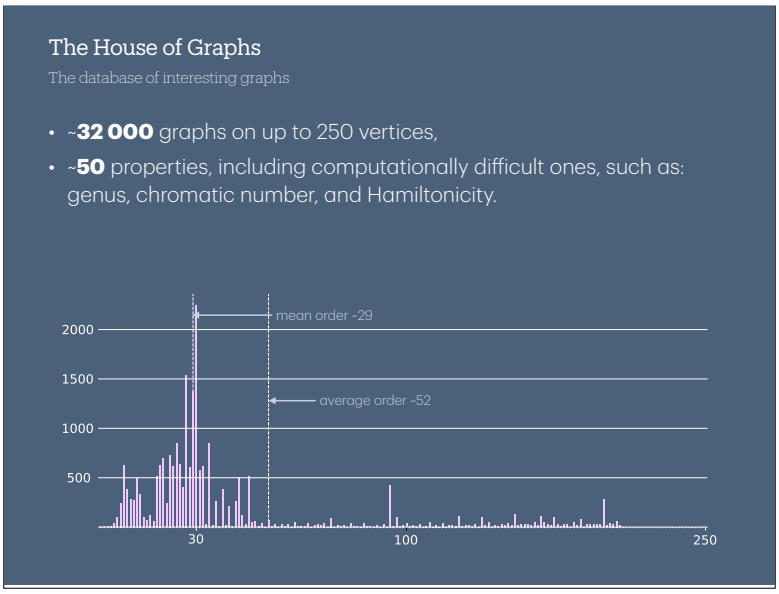
Is a list of examples complete and correct?
Is the connection between theory and code sound?
Are the computational results correct

In the LMFDB, they run redundant computations, integrity checks, and have found bugs in computer algebra systems.

Some standard options to increase the level of trust

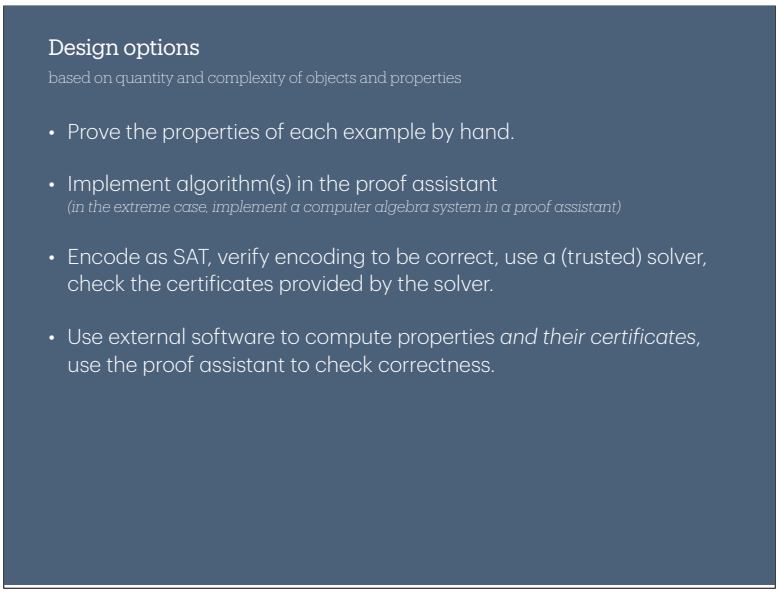
Techniques from engineering and mathematics

- 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.

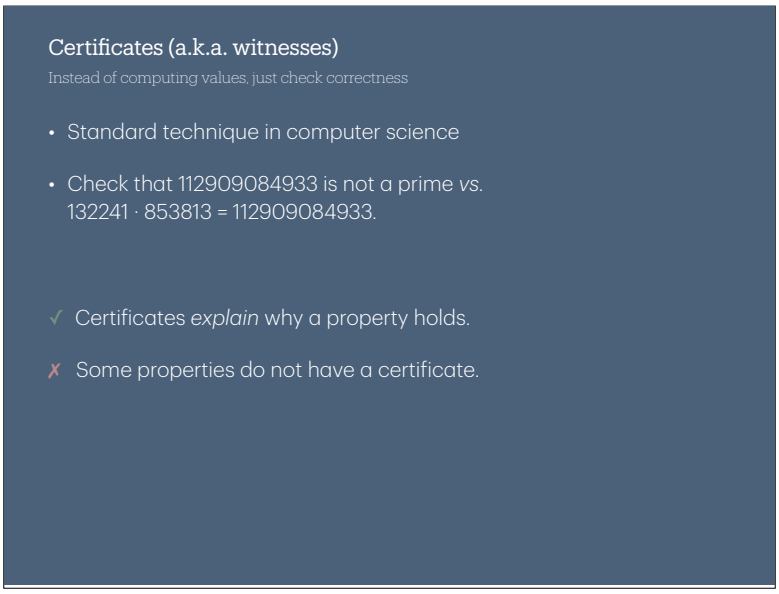


The combination of graph sizes and properties means that we can't just compute whichever way we want.

Even though we formalized only a few properties, it was harder than expected.

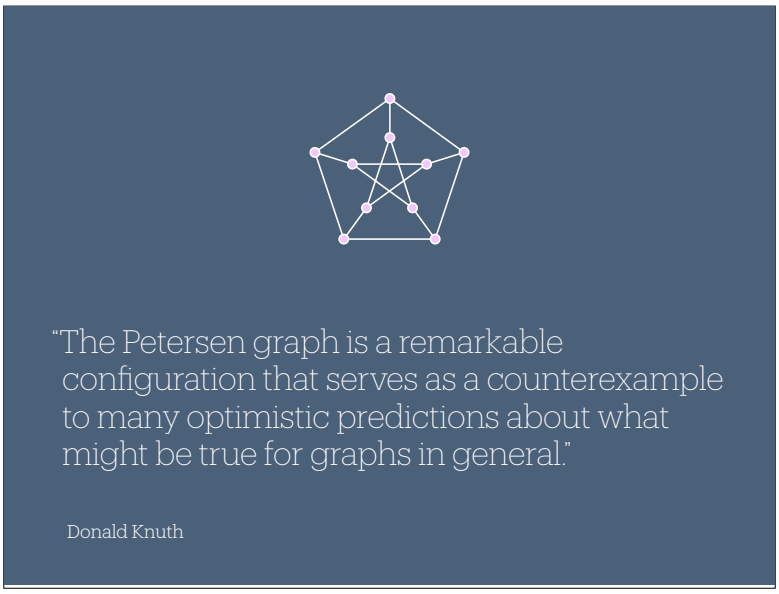


1. For few objects and properties, simple.
2. Few properties, many objects, efficiently computable: can be difficult.
3. We used a combination of the last two.



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.

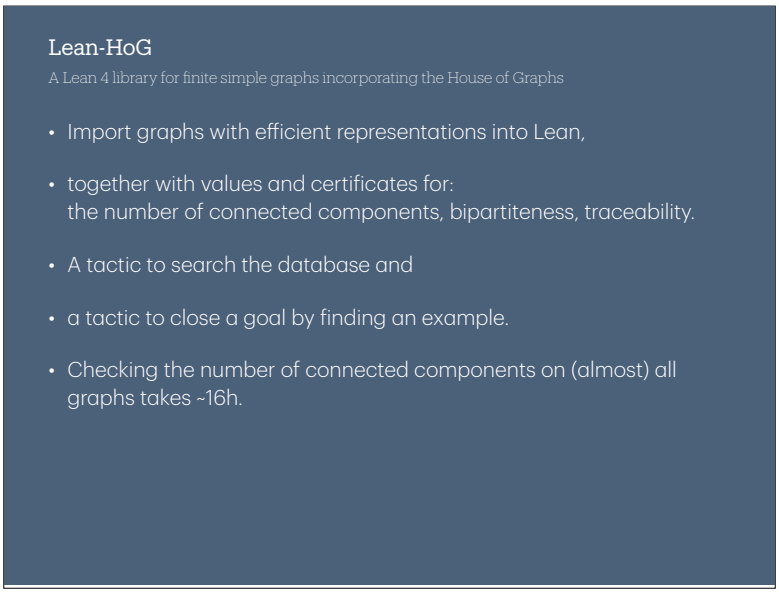
The Petersen graph

has a Hamiltonian path, is not Hamiltonian

A foreshadowing of things to come.

Easy-ish: find a path in the graph that visits all vertices exactly once,
Harder: prove that we can't find such a cycle in the graph.

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.

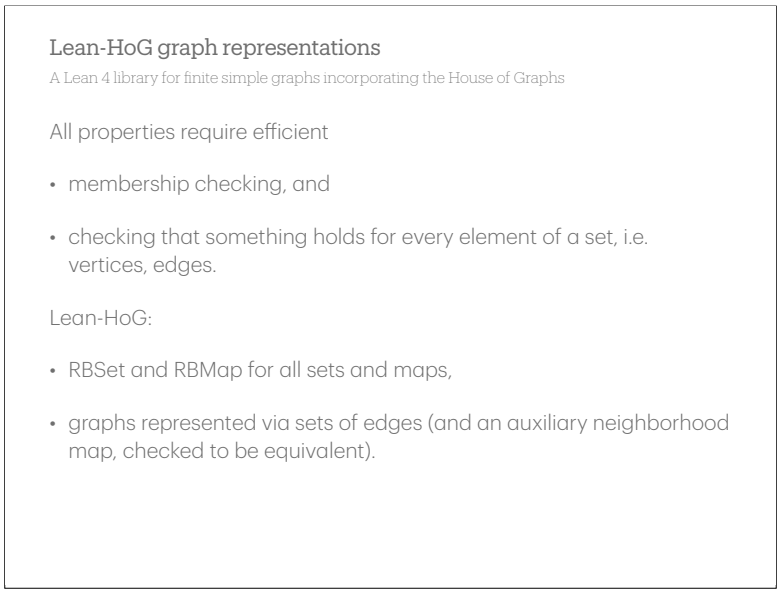
For the invariants (traceability), with certificates that only work in one direction, one strategy would be to complement them with heuristics wherever they work. For instance, detecting a disconnected graph is an easy way to rule out Hamiltonicity. Only when these simpler methods fail would we resort 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:
  $\forall\, i\, j : \mathrm{Fin}\, n\,.\, \mathrm{Adj}\, i\, j \to (ci \neq cj)$,
  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.



Certificates:

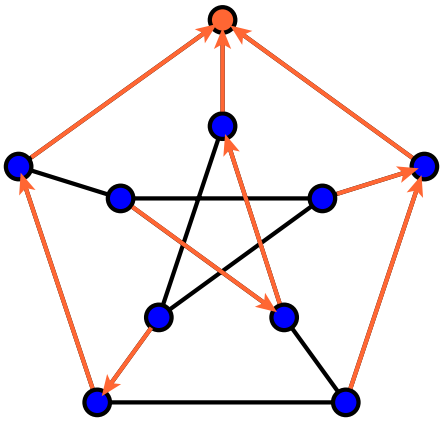
- could use regular certificates (no SAT) for the simple direction; use heuristics whenever they work for the other direction (such as a disconnected graph for Hamiltonicity), only resort to SAT when all else fails;
- we took the more principled approach with SAT.

Connected components

```
/-- 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
```

```
/-- 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

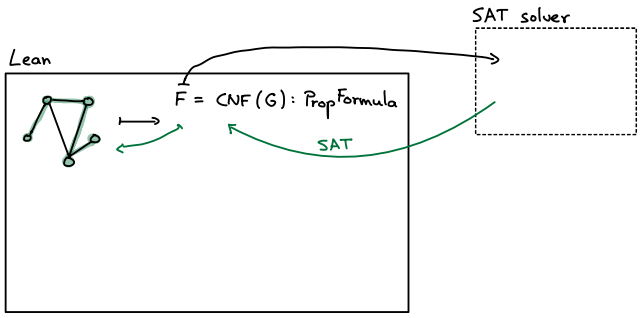
```
/-- A certificate for connected components -/
class ConnectedComponentsCertificate (G : Graph) : Type :=
  -- Data
  val : Nat
  component : G.vertex → Fin val
  root : Fin val → G.vertex
  next : G.vertex → G.vertex
  distToRoot : G.vertex → Nat

  -- 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) :=
```

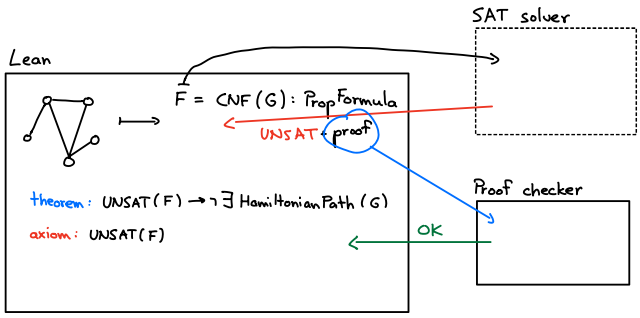
Hamiltonian Paths

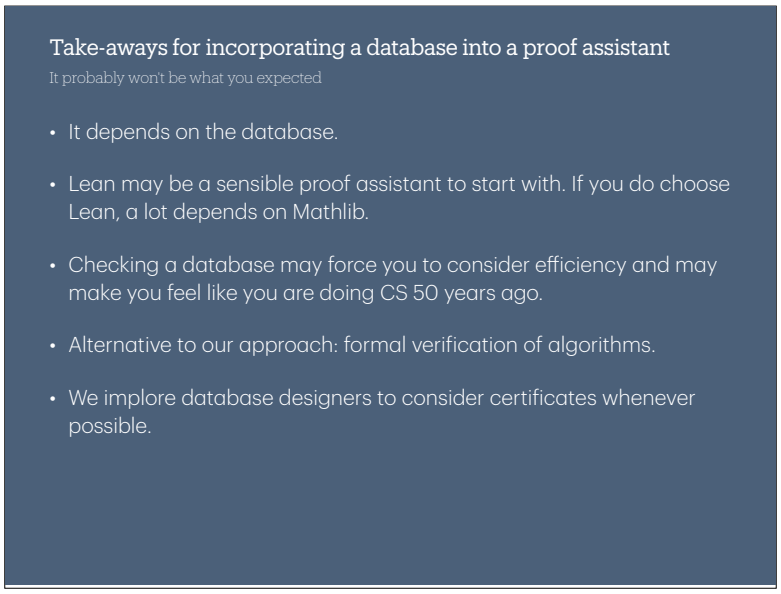
NP-hard, in general no easy certificate for non-existence.
Use a SAT solver.



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
- **1000+ theorems**
  Freek Wiedijk, Floris van Doorn, KB; editors for each system
- **Lean-HoG**
  Jure Taslak, Gauvain Devillez, KB, Andrej Bauer

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

Conference on Intelligent Computer Mathematics

CICM 2026 @Ljubljana

September 21-25

The website is online