Verified Enumeration of Plane Graphs Modulo Isomorphism

Tobias Nipkow

Fakultät für Informatik
TU München

1. Background

2. Generic enumeration

3. Application

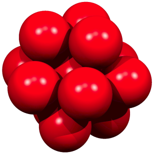
1. Background
2. Generic enumeration
3. Application

Kepler Conjecture (1611)

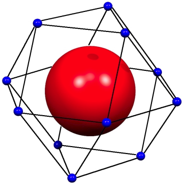
Theorem (Hales 1998). No packing of 3-dimensional balls of the same radius has density greater than the *face-centered cubic packing*.

Proof ideas

- Reduce infinite problem to (small!) finite one:



- Represent cluster as graph:



Sketch of Hales's proof

Proof by contradiction.

Assume there is a counterexample D.

Associate a plane graph (*contravening graph*) with D.

Theorem 0. Every contravening graph is *tame*.

Theorem 1. Every tame plane graph is isomorphic to a graph in the *Archive*.

Theorem 2. No graph in the Archive is contravening.

QED

Hales's proof of Theorem 1

- Java program to enumerate all tame plane graphs.
- Run program and check that each enumerated graph is isomorphic to one in the Archive.

But is the program correct?

The Flyspeck project

Check *all* of the proof with *interactive theorem provers*

Tom Hales & Co	Pitt & Vietnam	HOL light
John Harrison	Intel	HOL light
Steven Obua	TUM	Isabelle/HOL
Gertrud Bauer, T.N.	TUM	Isabelle/HOL

A first contribution

N., Bauer, Schultz verified Theorem 1 (IJCAR 2006):

- HOL is a functional programming language.
- Express executable enumeration of tame plane graphs in HOL (instead of Java).
- Verify that enumeration is complete.
- Execute enumeration and check against Archive.

Hales was right

Executing HOL

Execution by equational logic:

$$\mathit{last}[1,2,3] = \mathit{last}[2,3] = \mathit{last}[3] = 3$$

Too inefficient for Flyspeck.

Execution by compilation (to ML):

$$\mathit{last}[1,2,3] \overset{\mathrm{ML}}{\leadsto} 3$$

$100 \times$ less time and space.

Statistics for 2006 proof

Size of proof:	17 000 lines
Execution time:	1 hour
Number of graphs generated:	23 000 000
Number of tame graphs found:	35 000
Number of tame graphs mod iso:	3 000
Average size of graphs in Archive:	13 nodes, 18 faces

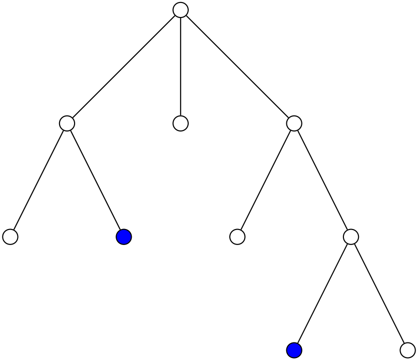
An improved proof

Christian Marchal. *Study of the Kepler's conjecture: The problem of the closest packing.*
Mathematische Zeitschrift. Published online 2009.

$\rightsquigarrow$

- simplifies geometric consideration
- simpler notion of tameness
- new archive of 19 000 tame graphs (mod iso)
- adapted Isabelle/HOL enumeration of tame graphs runs out of space

An enumeration tree



tame

The formalization

Given:

$succs : graph \to (graph)list$
$tame : graph \to bool$

A naive depth-first search:

$enum : (graph)list \to (graph)list \to (graph)list$

$enum\ []\ tgs \ = \ tgs$
$enum\ (g \cdot gs)\ tgs \ =$
$enum\ (succs\ g\ @\ gs)\ (\text{if } tame\ g \text{ then } g \cdot tgs \text{ else } tgs)$

Problems and solutions

Problems:

- Termination
- Removal of isomorphic tame graphs

Generic solutions:

- While combinator for partial functions
- Collections over a preorder (subsumption relation)

Termination

HOL:

- A logic of total functions
- Can also define partial functions by totalizing them

Function *enum*:

- Do not want to prove its termination — difficult
- It should suffice that its actual execution terminates
- Currently not directly definable in Isabelle
  (or elsewhere)

A while combinator

With a few tricks definable

$\mathit{while} : (\alpha \to \mathit{bool}) \to (\alpha \to \alpha) \to \alpha \to (\alpha)\mathit{option}$

where datatype $(\alpha)\mathit{option} = \mathit{None} \mid \mathit{Some}\ \alpha$

Lemmas:

$\mathit{while}\ b\ c\ s\ =\ (\text{if } b\ s \text{ then } \mathit{while}\ b\ c\ (c\ s) \text{ else } \mathit{Some}\ s)$

$$\frac{\mathit{while}\ b\ c\ s = \mathit{Some}\ t \quad P\ s \quad \forall s.\ P\ s \wedge b\ s \longrightarrow P(c\ s)}{P\ t}$$

A worklist function

$worklist\ succs\ f\ []\ s\ =\ Some\ s$

$worklist\ succs\ f\ (x \cdot ws)\ s\ =$
$worklist\ succs\ f\ (succs\ x\ @\ ws)\ (f\ x\ s)$

Easily definable from *while*.

Simple instance: $f\ x\ s = \text{if } tame\ x \text{ then } x \cdot s \text{ else } s$

Must avoid collecting isomorphic graphs!

Ignore x if $x \preceq y$ for some y already encountered

for some preorder $\preceq$

Collections over a preorder

An abstract data type:

$$
\begin{array}{ll}
\preceq: & e \rightarrow e \rightarrow \mathit{bool} \\[1ex]
\mathit{empty}: & s \\
\mathit{insert\text{-}mod}: & e \rightarrow s \rightarrow s \\
\mathit{set\text{-}of}: & s \rightarrow (e)\mathit{set}
\end{array}
$$

$$
\mathit{set\text{-}of}(\mathit{insert\text{-}mod}\ x\ s) = \{x\} \cup (\mathit{set\text{-}of}\ \ s) \vee \\
(\exists y \in \mathit{set\text{-}of}\ \ s.\ x \preceq y) \wedge \mathit{insert\text{-}mod}\ x\ s = s
$$

Enumeration modulo $\preceq$

$enum\ succs\ P\ =$
$worklist\ succs\ (\lambda x\ s.\ \text{if}\ P\ x\ \text{then}\ insert\text{-}mod\ x\ s\ \text{else}\ s)$

Implementing collections over $\preceq$

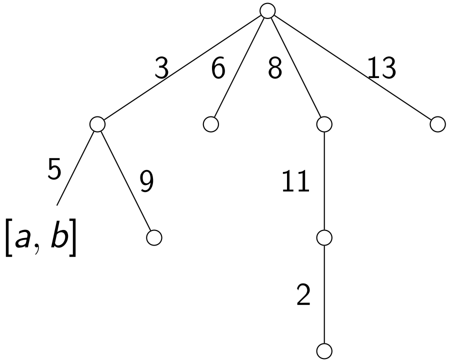
By hash-maps to lists of elements:

$$\begin{array}{ll} \mathit{key} : & e \rightarrow k \\ \mathit{lookup} : & m \rightarrow k \rightarrow (e)\mathit{list} \\ \mathit{update} : & m \rightarrow k \rightarrow (e)\mathit{list} \rightarrow m \end{array}$$

$$\begin{array}{l} \mathit{insert\text{-}mod}\ x\ m\ = \\ \text{let}\ \ k = \mathit{key}\ x; \\ \quad\ \ \mathit{ys} = \mathit{lookup}\ m\ k \\ \text{in}\ \ \text{if}\ \ \exists y \in \mathit{set}\ \mathit{ys}.\ x \preceq y\ \ \text{then}\ \ m \\ \quad\ \ \text{else}\ \ \mathit{update}\ m\ k\ (x \cdot \mathit{ys}) \end{array}$$

Implementing hash-maps

By tries ($\Longrightarrow$ key must be a list)



$$[3, 5] \mapsto [a, b], \ldots$$

Realisation in Isabelle

- Specify ADT as “locale” (= parameterised theory)
- Implement ADT by theory interpretation

1. Background
2. Generic enumeration
3. Application

To apply the generic enumeration theory to tame plane graphs we need

- a graph isomorphism test ($\preceq$)
- a hash function for graphs

Plane graph isomorphism test

Three alternatives:

- Implement and verify efficient linear-time algorithm
  — hard
- Implement unverified test and verify result-checker
  — the clever cop out
- Implement and verify reasonable algorithm
  — not too hard, and lets you sleep better

Hash function

$$key : graph \rightarrow (nat)list$$

$$key\ g \;=\; sort(map\ degree\ (nodes\ g))$$

Results

Execution time:	10 hours
Number of graphs generated:	2×10^9
Number of tame graphs found:	350 000
Number of tame graphs mod iso:	19 000
Avg number of graphs per trie node:	3

Found 2 graphs that were missing from Hales's Archive!

Two days later Hales emailed me:

> *I found the bug in my code! It was in the code that uses symmetry to reduce the search space. This is a bug that goes all the way back to the 1998 proof. It is just a happy coincidence that there were no missed cases in the 1998 proof. This is a good example of the importance of formal proof in computer-assisted proofs.*