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:

\[ \text{last}[1, 2, 3] = \text{last}[2, 3] = \text{last}[3] = 3 \]

Too inefficient for Flyspeck.

Execution by compilation (to ML):

\[ \text{last}[1, 2, 3]^{\text{ML}} \rightsquigarrow 3 \]

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


• 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
1 Background

2 Generic enumeration

3 Application
An enumeration tree

tame
The formalization

Given:

\[ \text{succs : graph} \rightarrow (\text{graph})\text{list} \]
\[ \text{tame : graph} \rightarrow \text{bool} \]

A naive depth-first search:

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

\[ \text{while} : (\alpha \to \text{bool}) \to (\alpha \to \alpha) \to \alpha \to (\alpha)\text{option} \]

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

Lemmas:

\[ \text{while } b \ c \ s = (\text{if } b \ s \text{ then while } b \ c \ (c \ s) \text{ else Some } s) \]

\[ \text{while } b \ c \ s = \text{Some } t \quad P \ s \quad \forall s. \ P \ s \land b \ s \rightarrow P(c \ s) \]

\[ P \ t \]
A worklist function

\[ \text{worklist succs } f \left( \emptyset \right) s = \text{Some } s \]

\[ \text{worklist succs } f \left( x \cdot ws \right) s = \text{worklist succs } f \left( \text{succs } x \odot ws \right) \left( f \cdot x \cdot s \right) \]

Easily definable from \textit{while}.

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

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:

\[ \preceq : \quad e \rightarrow e \rightarrow \text{bool} \]

\textit{empty} : \quad s

\textit{insert-mod} : \quad e \rightarrow s \rightarrow s

\textit{set-of} : \quad s \rightarrow (e)\text{set}

\[ \text{set-of}(\text{insert-mod} \times s) = \{ x \} \cup (\text{set-of} \ s) \uplus \]
\[ (\exists y \in \text{set-of} \ s. \ x \preceq y) \land \text{insert-mod} \times s = s \]
Enumeration modulo $\preceq$

\[
\text{enum succs } P = \\
\text{worklist succs } (\lambda x s. \text{if } P x \text{ then } \text{insert-mod } x s \text{ else } s)
\]
Implementing collections over $\leq$

By hash-maps to lists of elements:

- $key: e \rightarrow k$
- $lookup: m \rightarrow k \rightarrow (e)\text{list}$
- $update: m \rightarrow k \rightarrow (e)\text{list} \rightarrow m$

$\text{insert-mod } x \ m =$

let $k = key x;$
    $ys = lookup m k$

if $\exists y \in \text{set } ys. x \leq y$ then $m$
else $update m k (x \cdot ys)$
Implementing hash-maps

By tries (key must be a list)

\[
\begin{align*}
[3, 5] & \mapsto [a, b], \ldots \\
[5] & \mapsto [a, b] \\
3, 6, 8, & 13
\end{align*}
\]
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 ($\leq$)
- a hash function for graphs
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

\[ key : \text{graph} \rightarrow (\text{nat})\text{list} \]

\[ key \ g \ = \ \text{sort}(\text{map} \ \text{degree} \ (\text{nodes} \ g)) \]
Results

Execution time: 10 hours

Number of graphs generated: $2 \times 10^9$
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.