Challenges and Experiences in Managing Large-Scale Proofs

T. Bourke[1] *M. Daum*[1,2] G. Klein[1,2] R. Kolanski[1,2]

[1]NICTA, Sydney, Australia

[2]The University of NSW, Sydney, Australia

9 July 2012

Australian Government
Department of Broadband, Communications and the Digital Economy
Australian Research Council

NICTA Funding and Supporting Members and Partners

A Sense of Scale

Basic Claim

Scale changes everything—be it code or mathematical proof.

A Sense of Scale

Basic Claim

Scale changes everything—be it code or mathematical proof.

Some examples of **mechanised** proofs:

- four-colour theorem: around **60,000** lines in Coq
- higher-order logic (HOL) library in Isabelle: around **66,000** lines
- Archive of Formal Proofs (AFP):
  entries range between **145** and **80,917** lines in Isabelle
- CompCert verified compiler: about **100,000** lines in Coq

A Sense of Scale

Basic Claim

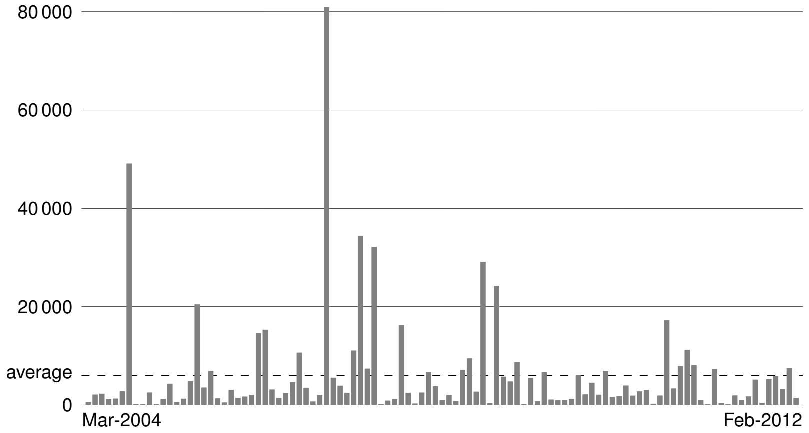
Scale changes everything—be it code or mathematical proof.

Some examples of **mechanised** proofs:

- four-colour theorem: around **60,000** lines in Coq
- higher-order logic (HOL) library in Isabelle: around **66,000** lines
- Archive of Formal Proofs (AFP):
  entries range between **145** and **80,917** lines in Isabelle
- CompCert verified compiler: about **100,000** lines in Coq
- L4.verified project repository: around **390,000** lines in Isabelle
- Verisoft project published over **500,000** lines in Isabelle

A Sense of Scale

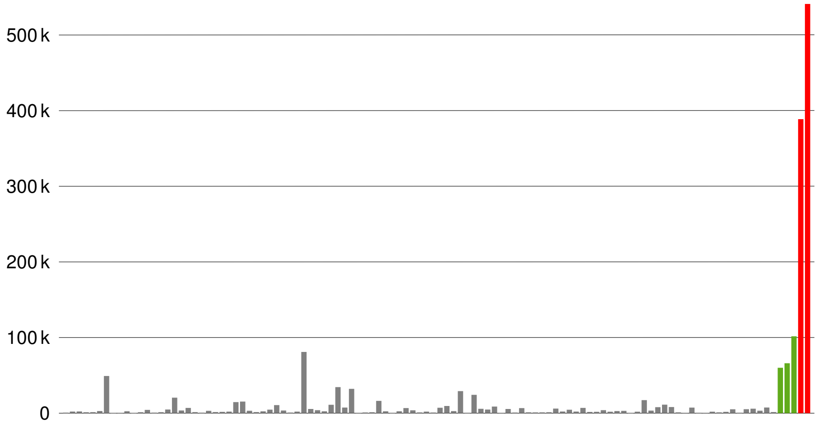
Size Distribution of AFP Entries



Size distribution of AFP entries in lines of proof, sorted by submission date

A Sense of Scale

Lines of Proof in Comparison



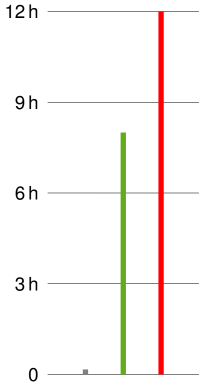
🟫 AFP entries by submission date
🟩 four-color theorem, Isabelle/HOL, CompCert
🟥 L4.verified, Verisoft

A Sense of Scale

Estimated Runtime for a Full Proof Check

- ⬛ Isabelle/HOL: 10 minutes
- 🟩 L4.verified: 8 hours
- 🟥 Verisoft: 12 hours

Note: checking times vary significantly with the utilization of the processor.



A Sense of Scale

Our Notion of “Large”

- different possible measures: lines, theorems, theories, . . .
- numbers vary with person, language, tool, problem, . . .
- hence: less stress on precise figures

A Sense of Scale

- different possible measures: lines, theorems, theories, . . .
- numbers vary with person, language, tool, problem, . . .
- hence: less stress on precise figures

Classification

Large-scale developments

- concern multiple people over multiple years and
- no single person understands the whole proof at any given time.

A Sense of Scale

Our Notion of “Large”

- different possible measures: lines, theorems, theories, ...
- numbers vary with person, language, tool, problem, ...
- hence: less stress on precise figures

Classification

Large-scale developments

- concern multiple people over multiple years and
- no single person understands the whole proof at any given time.

L4.verified: 12^+ people over 7^+ years

Overview: Challenges

Four perspectives:

- proof introspection – finding existing definitions and theorems
- proof development – proving new statements
- proof maintenance – keeping the proof base alive
- social and management aspects – from many brains to one proof

Overview: Challenges

Four perspectives:

- proof introspection – finding existing definitions and theorems
- proof development – proving new statements
- proof maintenance – keeping the proof base alive
- social and management aspects – from many brains to one proof

Proof Introspection

Rafal's Observation

I still maintain that the introspection of proof and theories is an essential part of working on a large-scale verification development.

Proof Introspection

Rafal's Observation

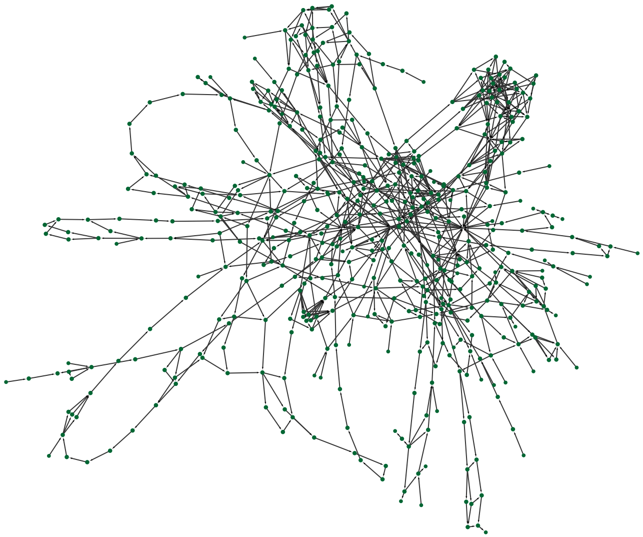
I still maintain that the introspection of proof and theories is an essential part of working on a large-scale verification development.

Our experience in training new team members:

- Learning Isabelle? – Easy.
- Understanding the verification subject? – No big deal.
- Understanding the proofs is the hard part!

Proof Introspection

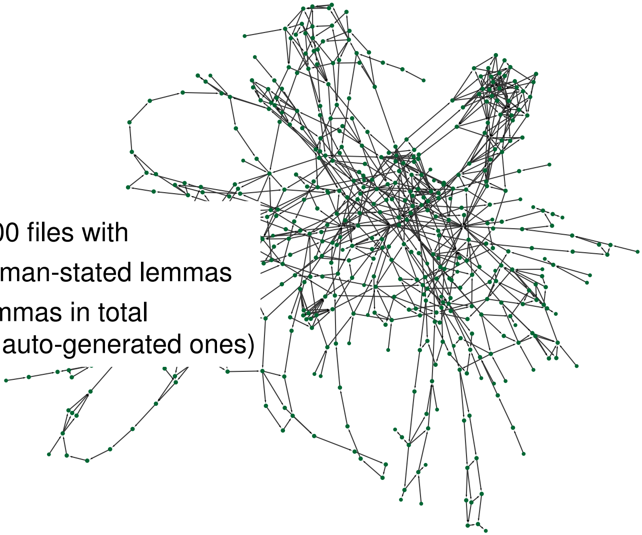
How do you find your way through this jungle?



Theory-file dependencies in L4.verified

Proof Introspection

How do you find your way through this jungle?



- roughly 500 files with
- 22,000 human-stated lemmas
- 95,000 lemmas in total (including auto-generated ones)

Theory-file dependencies in L4.verified

Proof Introspection

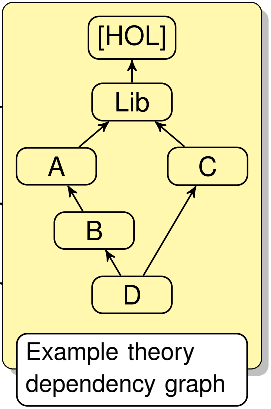
- `find_theorems` tool with
  - pattern-matching against theorem statements and names
  - filtering rules against current goal
  - ranking by most accurate match
- **auto-solve** function – warn if existing lemma is restated
- context-independent search over a **web-interface**
- **locate** tool[1] – find definitions; decode syntactic sugar

[1] inspired by Coq.

Proof Introspection

Solutions – L4.verified Tools

- `find_theorems` tool with
  - pattern-matching against theorem statem
  - filtering rules against current goal
  - ranking by most accurate match
- **auto-solve** function – warn if existing lem
- context-independent search over a **web-i**
- **locate** tool[1] – find definitions; decode syr



Example theory dependency graph

[1]inspired by Coq.

Proof Introspection

- `find_theorems` tool with
  - pattern-matching against theorem statements and names
  - filtering rules against current goal
  - ranking by most accurate match
- **auto-solve** function – warn if existing lemma is restated
- context-independent search over a **web-interface**
- **locate** tool[1] – find definitions; decode syntactic sugar

```
lemma trancl_refcl: "(r⁼)⁺ = r*"
```

[1] inspired by Coq.

Overview: Challenges

Four perspectives:

- proof introspection – finding existing definitions and theorems
- proof development – proving new statements
- proof maintenance – keeping the proof base alive
- social and management aspects – from many brains to one proof

Proof Development

Answer, Quick!

Progress

Proof checking in progress
Please wait...

7 h 56 min remaining

Proof Development

Matthias' Conjecture

Over the years, I must have waited weeks for Isabelle.
Productivity hinges on a short edit-check cycle;
for that, I am even willing to (temporarily) sacrifice soundness.

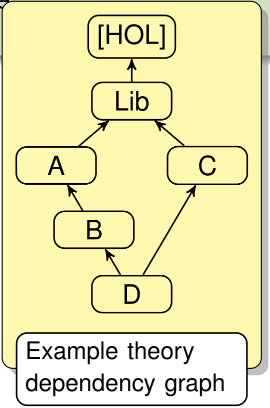
- Challenges:
  - non-local change
  - speculative change
  - distributed development
- Solutions:
  - skip-proof mode (Isabelle)
  - proof cache (L4.verified)
  - concurrency (recent improvement)

Proof Development

Answer, Quick!

Matthias' Conjecture

Over the years, I must have waited weeks for Isabelle.
Productivity hinges on a short edit-check cycle;
for that, I am even willing to (temporarily) sacr


$$\begin{tikzcd} & \text{[HOL]} & \\ & \text{Lib} \arrow[u] & \\ A \arrow[ur] & & C \arrow[ul] \\ B \arrow[u] & & \\ & D \arrow[ul] \arrow[uur] & \end{tikzcd}$$

Example theory dependency graph

- Challenges:
  - non-local change
  - speculative change
  - distributed development
- Solutions:
  - skip-proof mode (Isabelle)
  - proof cache (L4.verified)
  - concurrency (recent improvement)

Proof Development

`by auto`

Tim's Statement

Automating "donkey work" allows attention and effort to be focussed where most needed – but it must be done judiciously.

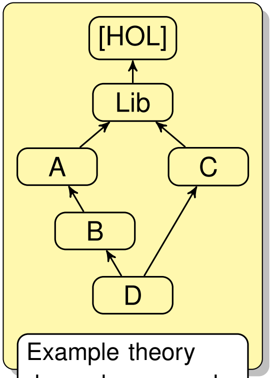
Focus: domain-specific automation

- extensions boost productivity
- unsoundness strikes back!

Proof Development

Common Problems

- lemma placement – solutions:
  - **Gravity** tool (Verisoft)
  - **Levity** tool (L4.verified)



$$\begin{tikzcd}
 & \text{[HOL]} & \\
 & \text{Lib} \arrow[u] & \\
A \arrow[ur] & & C \arrow[ul] \\
B \arrow[u] & & \\
 & D \arrow[ul] \arrow[uur] &
\end{tikzcd}$$

Example theory dependency graph

Proof Development

- lemma placement – solutions:
  - **Gravity** tool (Verisoft)
  - **Levity** tool (L4.verified)
- dealing with duplication
  - avoid – spot & mark – remove
  - generalise – automation?

Proof Development

- lemma placement – solutions:
  - **Gravity** tool (Verisoft)
  - **Levity** tool (L4.verified)
- dealing with duplication
  - avoid – spot & mark – remove
  - generalise – automation?
- proof and specification patterns
  Fight re-invention!
  Needs community-wide awareness.

Proof Development

Common Problems

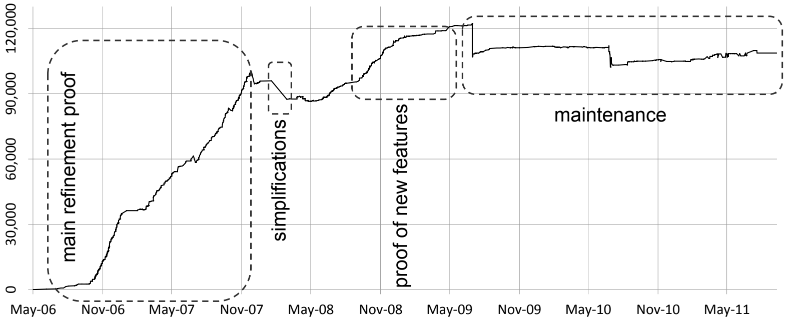
- lemma placement – solutions:
  - **Gravity** tool (Verisoft)
  - **Levity** tool (L4.verified)
- dealing with duplication
  - avoid – spot & mark – remove
  - generalise – automation?
- proof and specification patterns
  Fight re-invention!
  Needs community-wide awareness.
- scope and name spaces
  - more important but harder than in programming
  - balancing locality is tricky – definitions vs. theorems

Overview: Challenges

Four perspectives:

- proof introspection – finding existing definitions and theorems
- proof development – proving new statements
- proof maintenance – keeping the proof base alive
- social and management aspects – from many brains to one proof

Proof Maintenance



Lines of proof over time in one L4.verified module

Gerwin's Conclusion

Proof time is short but maintenance is for life.

Proof Maintenance

Need Sophisticated Refactoring Tools

WANTED

Native **proof-refactoring tools** to

- rename constants, types, and lemmas;
- reformulate definitions or properties for more consistency;
- move lemmas for better accessibility and reusablitiy;
- disentangle dependencies;
- remove duplication.

- largely unexplored, challenging research area
- even simple renaming requires semantic analysis
- non-local changes – automation paramount

Overview: Challenges

Four perspectives:

- proof introspection – finding existing definitions and theorems
- proof development – proving new statements
- proof maintenance – keeping the proof base alive
- social and management aspects – from many brains to one proof

Social and Management Aspects

From Many Brains to One Proof

- Discipline: `lemma bloody_obvious: "..."`
  - overwhelming need for meaningful names – challenging
  - self-discipline decreases over time – **tools!**
- State of Proof – regression tests
- Concurrent Development
  - compositionality under side-conditions,
    effective communication if side-conditions change
  - state the final top-level theorem first
  - continuous regression test

Social and Management Aspects

From Many Brains to One Proof

- Discipline: `lemma bloody_obvious: "..."`
  - overwhelming need for meaningful names – challenging
  - self-discipline decreases over time – **tools!**
- State of Proof –
- Concurrent Deve
  - compositiona tions,
    effective com onditions change
  - state the final rst
  - continuous re

Summary

Scale changes everything

Rafal:	**Proof introspection** is essential.
Matthias:	Productivity hinges on a **short edit-check cycle**.
Tim:	**Customisable automation** is crucial.
Gerwin:	**Maintenance** is for life.

Thank you
for your attention!