Boosting Automated Reasoning by Mining Existing Proofs

Thomas Gransden

Department of Computer Science
University of Leicester
tg75@student.le.ac.uk
Interactive Theorem Proving is Difficult

• User Driven
• Expert Required
• Large amounts of knowledge
• Time Consuming

A Large Scale Verification:
25-30 years combined effort
200,000 lines of Isabelle code

Problem:
Finding a suitable sequence of proof steps is hard!
Interactive Theorem Proving is Difficult

- User Driven
Interactive Theorem Proving is Difficult

- User Driven
- Expert Required
Interactive Theorem Proving is Difficult

- User Driven
- Expert Required
- Large amounts of knowledge
Interactive Theorem Proving is Difficult

- User Driven
- Expert Required
- Large amounts of knowledge
- Time Consuming
Interactive Theorem Proving is Difficult

- User Driven
- Expert Required
- Large amounts of knowledge
- Time Consuming

A Large Scale Verification:

25-30 years combined effort
200,000 lines of Isabelle code
Interactive Theorem Proving is Difficult

- User Driven
- Expert Required
- Large amounts of knowledge
- Time Consuming

A Large Scale Verification:

25-30 years combined effort
200,000 lines of Isabelle code

Problem:
Finding a suitable sequence of proof steps is hard!
Proof Automation
Proof Automation

- Much sought after property
  - Reduces Human Intervention
  - Benefits in many fields
Proof Automation

- Much sought after property
  - Reduces Human Intervention
  - Benefits in many fields

- Very active research area
Proof Automation

- Much sought after property
  - Reduces Human Intervention
  - Benefits in many fields

- Very active research area
  - International Tournaments!
Proof Automation

• Much sought after property
  • Reduces Human Intervention
  • Benefits in many fields

• Very active research area
  • International Tournaments!

• Restricted by underlying logic
**Proof Automation**

- Much sought after property
  - Reduces Human Intervention
  - Benefits in many fields

- Very active research area
  - International Tournaments!

- Restricted by underlying logic
  - Expressivity vs Automation Tradeoff
Proof Libraries

lemma "(3x. Vy. P x y) → (Vy. 3x. P x y)"
apply (rule impI)
apply (erule exE)
apply (rule allI)
apply (erule allE)
apply (rule exI)
apply assumption
done
Proof Libraries

- Examples of successful proofs

```plaintext
lemma "(∃x. ∀y. P x y) → (∀y. ∃x. P x y)"
  apply (rule impI)
  apply (erule exE)
  apply (rule allI)
  apply (rule allE)
  apply (rule exI)
  apply assumption
  done
```
Proof Libraries

- Examples of successful proofs
- Provided by an expert

```
lemma "(∃x. ∀y. P x y) → (∀y. ∃x. P x y)"
apply (rule impI)
apply (erule exE)
apply (rule allI)
apply (erule allE)
apply (rule exI)
apply assumption
done
```
Proof Libraries

• Examples of successful proofs
• Provided by an expert
• Variety of complexities/domains
Proof Libraries

- Examples of successful proofs
- Provided by an expert
- Variety of complexities/domains
- Specified as **proof steps**
Proof Libraries

- Examples of successful proofs
- Provided by an expert
- Variety of complexities/domains
- Specified as **proof steps**

**Idea:**
Can we use this information to automate new proofs?
<table>
<thead>
<tr>
<th>Available Knowledge</th>
<th>New Goal</th>
<th>ATP Systems</th>
<th>Isabelle/Metis</th>
<th>Relevance</th>
<th>Filter</th>
<th>Sledgehammer</th>
<th>Proof Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>2.</td>
<td>2.</td>
<td>2.</td>
<td>3.</td>
<td>4.</td>
<td>5.</td>
<td></td>
</tr>
</tbody>
</table>
Available Knowledge ➔ Relevance Filter ➔ ATP Systems ➔ Isabelle/Metis ➔ Proof Text

1. New Goal ➔ Sledgehammer ➔ 2.

Relevance Filter ➔ 2.

ATP Systems ➔ 3.

Isabelle/Metis ➔ 4.

Proof Text ➔ 5.
Increasing Automation in ITP’s - Proof Hints
Increasing Automation in ITP’s - Proof Hints

1. Proof Library
2. ML4PG
3. Lemma Clusters
4. New Goal
5. Theorem Prover
6.
**Useful Sequences** - Sequences of proof steps that could prove useful in proving some new goal
Useful Sequences - Sequences of proof steps that could prove useful in proving some new goal

Tactic - A function that is applied to a proof state
**Useful Sequences** - Sequences of proof steps that could prove useful in proving some new goal

**Tactic** - A function that is applied to a proof state

**Tactic Mining** - Automatically forming tactics from large libraries of existing proofs
Useful Sequences - Sequences of proof steps that could prove useful in proving some new goal

Tactic - A function that is applied to a proof state

Tactic Mining - Automatically forming tactics from large libraries of existing proofs

Sequence 1: rule impl assumption  Sequence 2: rule conjl assumption  Tactic: (rule impl OR rule conjl) THEN assumption
Previous Tactic Mining Work

Carried out by Hazel Duncan at Edinburgh.
Previous Tactic Mining Work

Carried out by Hazel Duncan at Edinburgh.
Critique of Duncan’s approach

There are some limitations of Duncan’s work:
Critique of Duncan’s approach

There are some limitations of Duncan’s work:

- Moderately effective on test set
Critique of Duncan’s approach

There are some limitations of Duncan’s work:

- Moderately effective on test set
- No subgoal information
Critique of Duncan’s approach

There are some limitations of Duncan’s work:

- Moderately effective on test set
- No subgoal information
- Inefficient tactic application
My Tactic Mining Approach

1. New Goal
2. Proof Library
3. Useful Sequences
4. Tactics
5. Theorem Prover
6. Tactic Miner
1. How can we deal with complex Higher Order Languages?

Variable instantiations and proof directives
1. How can we deal with complex Higher Order Languages?

Variable instantiations and proof directives

One sequence of steps solves many proofs and vice versa
1. How can we deal with complex Higher Order Languages?

Variable instantiations and proof directives

One sequence of steps solves many proofs and vice versa

Different proof styles
2. Which Data Mining Techniques can help?

An open research question
2. Which Data Mining Techniques can help?

An open research question

Two tasks: Finding the patterns and generalising into tactics
3. How will the theorem prover and tactic miner communicate?

We require two methods of communication to be defined:

- Theorem Prover to Tactic Miner
- Tactic Miner to Theorem Prover
4. How can we make use of negative information?

Leverage negative information from:

- User inputs
- Failed traces from existing automated tools

Would enable a supervised learning approach.

Diagram:

- **Tactic Miner**
  - **Proof Library** (1.)
  - **New Goal** (5.)
  - **Theorem Prover** (6.)
  - **Useful Sequences** (3.)
  - **Tactics** (4.)
4. How can we make use of negative information?

Leverage negative information from:

- User inputs
4. How can we make use of negative information?

Leverage negative information from:

- User inputs
- Failed traces from existing automated tools
4. How can we make use of negative information?

Leverage negative information from:

- User inputs
- Failed traces from existing automated tools

Would enable a supervised learning approach.
Current Work

I am currently at the following stage with my work:

- Data Extraction from Isabelle
- Considering learning techniques
I am currently at the following stage with my work:

- Data Extraction from Isabelle
I am currently at the following stage with my work:

- Data Extraction from Isabelle
- Considering learning techniques
Any Questions?

Please feel free to ask me any questions, either now or at any point during the workshop!