ML4PG: Machine Learning for Proof General
Ekaterina Komendantskaya and Jónathan Heras
http://www.computing.dundee.ac.uk/staff/katya/ML4PG/

Interactive Theorem Provers (ITPs)

...are programming languages applied for:

★ software verification in industry:
  • ARM microprocessor: 20,000 lines of code;
  • verified C compiler: 50,000 lines of code;
★ and formalisation of mathematics:
  • Four Colour theorem: 60,000 lines of code;
  • Feit-Thompson theorem: 170,000 lines;
  • Flipsyce project: 325,000 lines of code.

ML4PG overview

Step 1: Feature extraction

• ML4PG works on the background of Proof General, and extracts statistical features from interactive proofs in Coq [2] and SSReflect [3]:
  • The features reflect shapes of lemmas, structure of proofs, and patterns of user interaction with the ITP.
  • Proof trace method captures statistical relation between several proof steps.

Step 2: Machine learning tools

• As higher-order proofs in general can take an infinite variety of shapes and sizes, ML4PG does not use any a priori given training labels:
  • it uses unsupervised learning (clustering) algorithms implemented in MATLAB and Weka; and
  • allows the user to adjust learning parameters, e.g. the size and proximity of clusters.
  • The output shows families of related proofs.

Step 3: Interaction with ML4PG

• ML4PG automatically sends the gathered statistics to a chosen machine-learning interface and triggers execution of a clustering algorithm of the user’s choice;
• it does some gentle post-processing of the results given by the machine-learning tool, and displays families of related proofs to the user.

Discovered Proof Families

Proof hints provided by ML4PG for the proof of Lemma 1: all proof families below contain proofs of already proven ∗, ∗ ∗, and ∗ ∗ ∗.

Clustering algorithm:

<table>
<thead>
<tr>
<th>g = 1</th>
<th>g = 2</th>
<th>g = 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian</td>
<td>9</td>
<td>0</td>
</tr>
<tr>
<td>K-means (MATLAB)</td>
<td>20</td>
<td>3</td>
</tr>
<tr>
<td>K-means (Weka)</td>
<td>29</td>
<td>10</td>
</tr>
<tr>
<td>FarthestFirst</td>
<td>27</td>
<td>24</td>
</tr>
</tbody>
</table>

Size of dataset: ≈ 150 lemmas. The granularity parameter g ranges from 1 (producing big and general clusters) to 5 (producing small and precise clusters).

Our solution: ML4PG

Proof General [1] is a user interface for several existing ITPs. ML4PG [4] is a machine-learning extension to Proof General [1] that:

• finds common proof-patterns in proofs across various scripts, libraries, users and notations;
• provides proof-hints, especially in industrial cases where routine similar cases are frequent, and effort is distributed across several programmers.

ML4PG session for Coq/SSReflect

Challenges

• Manual handling of various proofs, strategies and libraries becomes difficult;
• Team-development is hard, especially since ITPs are sensitive to user notation;
• Manual handling of various proofs, strategies and libraries becomes difficult;
• Comparison of proofs and proof similarities are sensitive to user notation;

Our solution: ML4PG

Case Study: Verification of Java Virtual Machine with ML4PG

Java Virtual Machine (JVM) is a stack-based abstract machine which can execute Java bytecode.

We modelled a subset of the JVM in Coq, verifying the interpreter for JVM programs. This work is inspired by the ACL2 proofs about JVMs [5].

```coq
Lemma 1 (Factorial JVM lemma) ∀ n ∈ N, running the bytecode associated with the factorial program with n as input, the Coq JVM produces a state which contains n! on top of the stack.
```

After processing the proof statistics of 150 lemmas in the library, ML4PG correctly suggested to reuse the proof strategy from similar (already proven) lemmas concerning different operations:

• multiplication JVM lemma
  • power JVM lemma
  • exponentiation JVM lemma

Case study: ML4PG Role in Proof Pattern Discovery

As part of JVM verification process, we needed to prove in Coq the following lemma:

After processing the proof statistics of 150 lemmas in the library, ML4PG correctly suggested to reuse the proof strategy from similar (already proven) lemmas concerning different operations:

References


ML4PG: Machine Learning for Proof General
Ekaterina Komendantskaya and Jónathan Heras
http://www.computing.dundee.ac.uk/staff/katya/ML4PG/