Interactive Proofs:
Automation, Computer Algebra, Presentation

Cezary Kaliszyk                Josef Urban
University of Innsbruck        Radboud University

August 25, 2014
Outline

**Automation for Interactive Proof (HOL(y)Hammer)**
- Machine Learning
- Translation
- Reconstruction

**Computer Algebra**
- Certified Computer Algebra
- SMTs and special functions

**Readable Proofs**
- Declarative Proofs

**Conclusion**
Interactive proofs

- Formal proof \textit{skeleton} + filling in the gaps
  - Searching for needed theorems
  - \textit{Tedious} properties
- Proof structure is lost
  - Uninteresting parts \textit{overshadow} interesting ones
Interactive proofs

- Formal proof **skeleton** + filling in the gaps
  - Searching for needed theorems
  - **Tedious** properties
- Proof structure is lost
  - Uninteresting parts **overshadow** interesting ones
- **Automation** for Interactive Proof
  - Tableaux: Itaut, Tauto, Blast
  - Rewriting: Simp, Subst, HORewrite
  - Decision Procedures: Congruence Closure, Ring, Omega, Cooper
- Large-theory **ATP and translation** techniques
  - Mizar: MaLARea
  - Isabelle/HOL: Sledgehammer
  - HOL(y)Hammer
HOL(y) Hammer

Learning-assisted automated reasoning for HOL Light

Request Advice:

Input the HOL Light formula to prove and select HOL Light session:

- polyhedron p ==> convex (relative_interior p)
- mv193.

Result (3.81s): CONVEX_RELATIVE_INTERIOR POLYHEDRON_IMP_CONVEX
Replaying: SUCCESS
(0.29s):SIMP_TAC[POLYHEDRON_IMP_CONVEX;CONVEX_RELATIVE_INTERIOR]

Examples:
AI-ATP systems (✩-Hammers)

Proof Assistant \rightarrow ITP Proof \rightarrow ✩Hammer \rightarrow ATP Proof \rightarrow First Order Problem \rightarrow Current Goal \rightarrow Proof Assistant

Flyspeck (including core HOL Light and Multivariate)
Mizar / MML
Isabelle (Auth, Jinja)

45%
AI-ATP systems (⋆-Hammers)

Current Goal \rightarrow ITP Proof \rightarrow ⋆Hammer \rightarrow ATP Proof \rightarrow First Order Problem

Proof Assistant

How much can it do?
AI-ATP systems (★-Hammers)

How much can it do?

- Flyspeck (including core HOL Light and Multivariate)
- Mizar / MML
- Isabelle (Auth, Jinja)
AI-ATP systems (⭐-Hammers)

Current Goal \rightarrow ITP Proof \rightarrow ⭐Hammer \rightarrow ATP Proof \rightarrow First Order Problem

Proof Assistant \rightarrow ITP Proof \rightarrow ⭐Hammer

How much can it do?

- Flyspeck (including core HOL Light and Multivariate)
- Mizar / MML
- Isabelle (Auth, Jinja)

\approx 45\%
Machine learning techniques

Algorithms

- Syntactic methods
  - Neighbours using various metrics, Recursive (MePo)
- Sparse Naive Bayes
  - Variable prior, Confidence
- k-Nearest Neighbours
  - TF-IDF, Dependency weighting
- Neural Networks
  - Winnow, Perceptron
- Linear Regression
  - Needs feature and theorem space reduction

Combining original and ATP dependencies

- Added value depends on the precision of human deps
Translation overview (FOL)

1. Heuristic type instantiation
   - Similar for induction

2. Eliminate $\epsilon$

3. Remove $\lambda$-abstractions
   - lifting, combinators, ...

4. Optimizations
   - if..then..else, $\exists!$

5. Separate predicates and terms
   - Consider cases, introduce bool variables

6. NNF, Skolemize

7. Use apply functor to make all applications first-order

8. Encode remaining types
   - monomorphisation, tags, guards

9. Various optimizations (incomplete)

Similar for TFF1, THF0, ...

## Re-proving (Flyspeck, 30sec)

<table>
<thead>
<tr>
<th>Prover</th>
<th>Theorem%</th>
<th>CounterSat%</th>
<th>Sotac–Σ</th>
</tr>
</thead>
<tbody>
<tr>
<td>E-par</td>
<td>38.4</td>
<td>0.0</td>
<td>69.12</td>
</tr>
<tr>
<td>Z3-4</td>
<td>36.1</td>
<td>0.0</td>
<td>61.51</td>
</tr>
<tr>
<td>E</td>
<td>32.6</td>
<td>0.0</td>
<td>45.44</td>
</tr>
<tr>
<td>Leo II</td>
<td>31.0</td>
<td>0.0</td>
<td>45.77</td>
</tr>
<tr>
<td>Vampire</td>
<td>30.5</td>
<td>0.0</td>
<td>45.75</td>
</tr>
<tr>
<td>CVC3</td>
<td>28.9</td>
<td>0.0</td>
<td>43.36</td>
</tr>
<tr>
<td>Satallax</td>
<td>26.9</td>
<td>0.0</td>
<td>48.75</td>
</tr>
<tr>
<td>Yices1</td>
<td>25.3</td>
<td>0.0</td>
<td>33.32</td>
</tr>
<tr>
<td>IProver</td>
<td>24.5</td>
<td>0.6</td>
<td>29.50</td>
</tr>
<tr>
<td>Prover9</td>
<td>24.3</td>
<td>0.0</td>
<td>29.98</td>
</tr>
<tr>
<td>Spass</td>
<td>22.9</td>
<td>0.0</td>
<td>26.22</td>
</tr>
<tr>
<td>LeanCop</td>
<td>21.4</td>
<td>0.0</td>
<td>26.98</td>
</tr>
<tr>
<td>AltErgo</td>
<td>19.8</td>
<td>0.0</td>
<td>26.82</td>
</tr>
<tr>
<td>Paradox 4</td>
<td>0.0</td>
<td>18.2</td>
<td>0.06</td>
</tr>
<tr>
<td>any</td>
<td>50.2</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
Proof Reconstruction

- Existing reconstruction mechanisms
  - Metis, SMT
  - Mizar by
  - MESON, Prover9

- Parse TSTP/SMT proofs
  - Create subgoals that match ATP intermediate steps
  - Automatically solve all simple ones

- High reconstruction rates give confidence in our techniques
  - Naive reconstruction: 90% (of Flyspeck solved)
    - MESON, SIMP, _ARITH_TAC
  - With TSTP parsing: 96%
Initial experiment

- Emf and Ray are 476 + 125 proved theorems
- 3 k-NN strategies and ATP deps only: 36%
Outline

Automation for Interactive Proof (HOL(y)Hammer)
  Machine Learning
  Translation
  Reconstruction

Computer Algebra
  Certified Computer Algebra
  SMTs and special functions

Readable Proofs
  Declarative Proofs

Conclusion
Computer Algebra Systems

- Computer programs for processing mathematical expressions
  - simplifications, substitutions, equations, numerical approximations, graphs
- Easy to use for a beginner
- Efficient

Problems with Computer Algebra Systems:
- Uncertified Algorithms,
- Tracking Assumptions,
- Domain of variables and type of expressions.

Solution: Build a CAS inside a Proof Assistant
- CAS-like User Interface
- Abstract Computer Algebra System conversion
- Knowledge specific for CAS systems
Computer Algebra Features

- Numerical approximation
  - Is not a computable function
Computer Algebra Features

- Numerical approximation
  - Is not a computable function

- Evaluation (vs Verification)

\[
\forall x. (f \text{ diff } g) \ x \rightarrow \text{diff } f = g \\
(f \text{ diff } g) \ x \rightarrow (\text{diff } f) \ x = g \ x
\]
Computer Algebra Features

- Numerical approximation
  - Is not a computable function

- Evaluation (vs Verification)

\[ \forall x. (f \ \text{diff} \ g) \ x \rightarrow \text{diff} \ f = g \]
\[ (f \ \text{diff} \ g) \ x \rightarrow (\text{diff} \ f) \ x = g \ x \]

- Partiality
  - On paper: \( \ldots \frac{1}{x} \ldots \)
  - In a PA: \( \forall x \in \mathbb{C}. x \neq 0 \Rightarrow \ldots \frac{1}{x} \ldots \)
Computer Algebra Features

- Numerical approximation
  - Is not a computable function

- Evaluation (vs Verification)

\[ \forall x. (f \text{ diff } g) \ x \rightarrow \text{diff } f = g \]
\[ (f \text{ diff } g) \ x \rightarrow (\text{diff } f) \ x = g \ x \]

- Partiality
  - On paper: \( \ldots \frac{1}{x} \ldots \)
  - In a PA: \( \forall x \in \mathbb{C}. x \neq 0 \Rightarrow \ldots \frac{1}{x} \ldots \)

- Real numbers
  - Either constructive and computable or classical
Satisfiability Modulo Theories

- SMT and TFA format
  - Translation of `int`, `rat`, `real` straightforward
  - Translation of `nat`

\[
\forall x : num. \ F[x] \implies \ \forall x' : int. (x' \geq 0) \implies F[x']
\]

\[
F[x : num] \implies F[x'] \wedge (x' : int \geq 0)
\]

\[
F[f] \implies F[f'] \wedge \forall (x' : int) \ y. (x' \geq 0) \implies f'(x', y) \geq 0
\]

- Initial evaluation of Beagle for the types above
  - It proves 81% of conjectures solved by Metis without arithmetic lemmas
  - Reconstruction requires proper traces

- Metitarski
  - Reasoning modulo special functions
ATP proof transformation
miz3
Conclusion and Future work

- Hammer-systems
  - For Physics
- Computer Algebra
  - Translation and Reconstruction
- Readable Proofs
  - Good Automation
  - Declarative?
  - Rendering
HOL(y) Hammer
Machine learning based premise selection for HOL Light

http://cl-informatik.uibk.ac.at/software/hh/
References

C. Kaliszyk and J. Urban.
Mizar 40 for Mizar 40.

C. Kaliszyk and J. Urban.
PRocH: Proof reconstruction for HOL Light.

C. Kaliszyk and J. Urban.
HOL(y)Hammer: Online ATP service for HOL Light.
http://dx.doi.org/10.1007/s11786-014-0182-0.

C. Kaliszyk and J. Urban.
Learning-assisted automated reasoning with Flyspeck.
http://dx.doi.org/10.1007/s10817-014-9303-3.

D. Kühlwein, J. C. Blanchette, C. Kaliszyk, and J. Urban.
MaSh: Machine learning for Sledgehammer.

C. Tankink, C. Kaliszyk, J. Urban, and H. Geuvers.
Formal mathematics on display: A wiki for Flyspeck.