

Hammering towards Qed

Cezary Kaliszyk

Josef Urban

University of Innsbruck

Radboud University

July 18, 2014

Outline

Automation for Interactive Proof

- Translations

- Evaluation

- Machine Learning

- Reconstruction

Towards Qed

- Strength

- Logics

- Knowledge

Interactive proofs

- ▶ Formal proof **skeleton** + filling in the gaps
 - ▶ Searching for needed theorems
 - ▶ **Tedious** properties
- ▶ Proof structure is lost
 - ▶ Uninteresting parts **overshadow** interesting ones

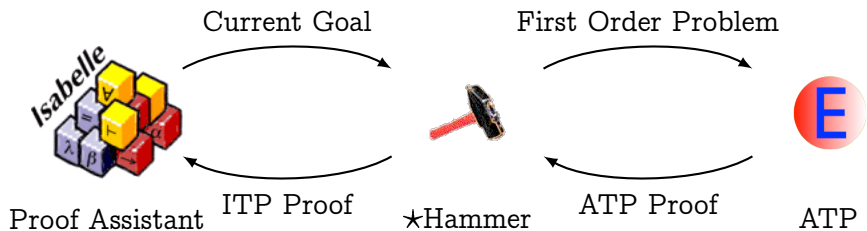
Interactive proofs

- ▶ 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

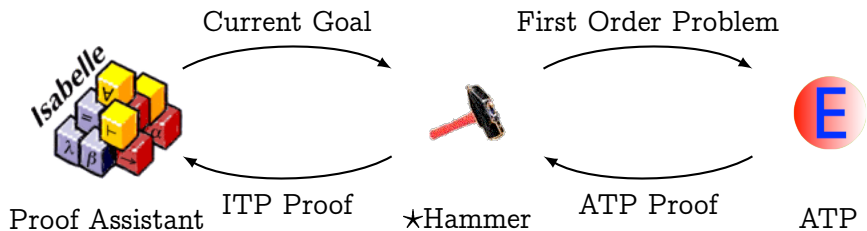
MizAR demo

<https://www.youtube.com/watch?v=4es4iJKtM3I>

AI-ATP systems (\star -Hammers)

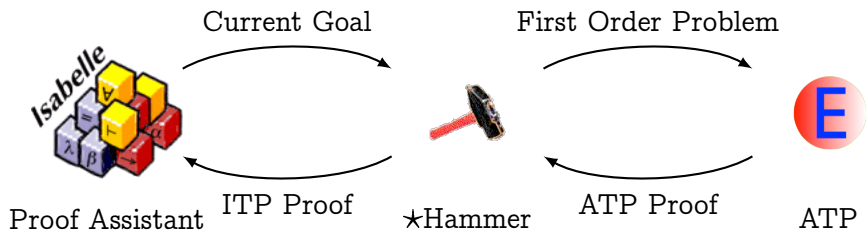


AI-ATP systems (★-Hammers)



How much can it do?

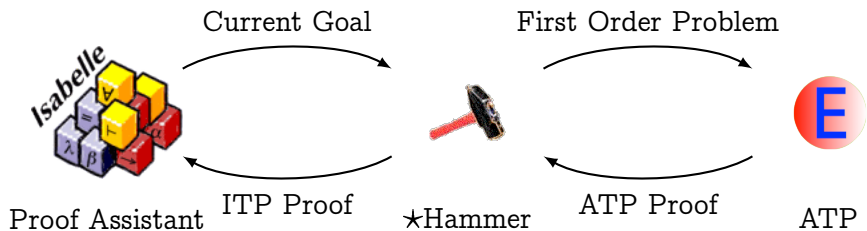
AI-ATP systems (★-Hammers)



How much can it do?

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

AI-ATP systems (★-Hammers)



How much can it do?

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

≈ 45%

Translation Overview

- ▶ Various exports to FOF
 - ▶ MESON-style monomorphisation
 - ▶ TFF-style type tagging
 - ▶ Isabelle-style type guards
- ▶ Export to TFF1
 - ▶ Additional provers (Alt-ergo)
 - ▶ Tools that do Monomorphisation of TPTP (Why3, tptp2X)
- ▶ Export to THF0
 - ▶ Satallax, Leo-II, ...
 - ▶ Monomorphisation makes the problems big and slow
- ▶ SMT solvers
 - ▶ Reconstruction
- ▶ Export to other ITPs
 - ▶ Rarely better

Translation overview (HOL)

- 1 Heuristic type instantiation
 - ▶ Similar for induction
- 2 Eliminate ϵ
- 3 Remove λ -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)

HOL(y) Hammer

*Learning-assisted automated
reasoning for HOL Light*



Request Advice:

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

-
-

(cache:OK)(session:OK)(parse:OK)SSSSAWAAWAW

Result (3.81s): CONVEX_RELATIVE_INTERIOR POLYHEDRON_IMP_CONVEX

Replaying: SUCCESS

(0.29s):SIMP_TAC[POLYHEDRON_IMP_CONVEX;CONVEX_RELATIVE_INTERIOR]

Examples:

Re-proving (Flyspeck, 30sec)

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

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

Features for Machine Learning

- ▶ A function that given a goal or premise returns a sparse vector
 - ▶ Optionally weights for kinds of features
 - ▶ Internal TF-IDF
- ▶ Types and type variables
- ▶ Constants
- ▶ **Subterms / Patterns**
 - ▶ No variable normalization
 - ▶ De-Bruijn indices
 - ▶ Types of variables
 - ▶ Normalization of type variables
- ▶ Meta information: Theory name, kind of rule, contains \exists , ...

Naive Bayes

- ▶ Each predictor
 - ▶ Given a vector of features of a goal g and a set of facts
 - ▶ Returns the predicted relevance for each fact f
- ▶ Assume independence between the features

$$P(f \text{ is relevant for proving } g)$$

$$= P(f \text{ is relevant} \mid g\text{'s features})$$

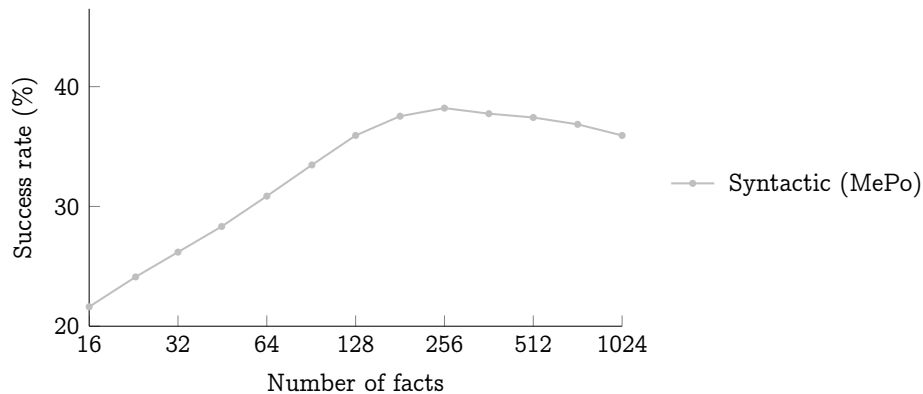
$$= P(f \text{ is relevant} \mid f_1, \dots, f_n)$$

$$\propto P(f \text{ is relevant}) \prod_{i=1}^n P(f_i \mid f \text{ is relevant})$$

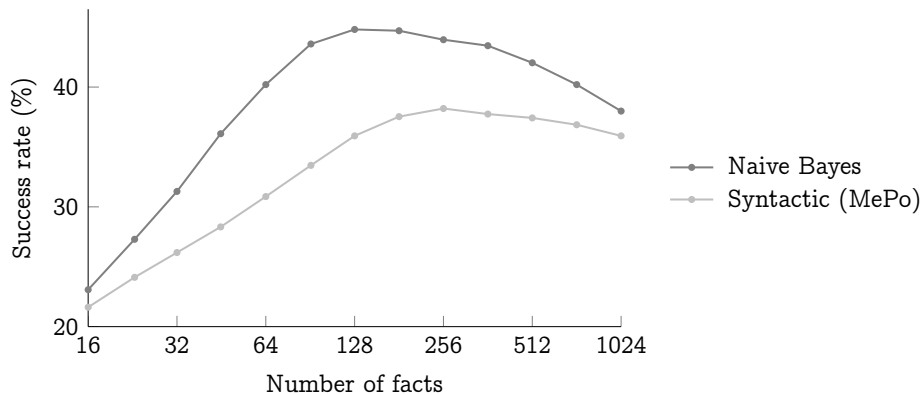
$$\propto \#f \text{ is a proof dependency} \cdot \prod_{i=1}^n \frac{\#f_i \text{ appears when } f \text{ is a proof dependency}}{\#f \text{ is a proof dependency}}$$

- ▶ Efficient
 - ▶ Fast predictions
 - ▶ Fast updates
 - ▶ Small models

Success Rates



Success Rates



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%

Outline

Automation for Interactive Proof

Translations

Evaluation

Machine Learning

Reconstruction

Towards Qed

Strength

Logics

Knowledge

Improve Percentage

- ▶ Is 100% possible?
 - ▶ Granularity of steps also increases
- ▶ Premise selection

- ▶ Encodings

- ▶ ATP-systems

- ▶ Reconstruction

Improve Percentage

- ▶ Is 100% possible?
 - ▶ Granularity of steps also increases
- ▶ Premise selection
 - ▶ Good machine learning algorithms are still **slow**
- ▶ Encodings
 - ▶ Efficient but more **complete**
- ▶ ATP-systems
 - ▶ Strategies and combinations
- ▶ Reconstruction
 - ▶ **Formalized** decision procedures

ITP logics

- ▶ MizAR
 - ▶ Set theory, dependent types, (almost) first order
- ▶ Sledgehammer, HOL(y)Hammer, ...
 - ▶ HOL, shallow polymorphism
- ▶ ACL2
 - ▶ Structure Irrelevance, Logic as lists
- ▶ Isabelle/ZF, ...
 - ▶ All features of meta-logic necessary
- ▶ Coq
 - ▶ Good machine-learning, but encodings hard

Sharing parts among systems

- ▶ Machine Learning Predictors
 - ▶ Already many shared
- ▶ Feature extraction
 - ▶ Given **common data format**
- ▶ Certain Transformations
 - ▶ λ -lifting, combinators, apply functor
 - ▶ Monomorphisation, Heuristic instantiation
 - ▶ Type encodings (tags, guards, soft-types, ...)
- ▶ Knowledge management
 - ▶ Namespaces, Browsing, Search, Refactoring, Change management
- ▶ **Readable** proof reconstruction

Common Functionality

- ▶ TPTP hierarchy: FOF, TFF1, THF0, ?

- ▶ THF1 already used

- ▶ Sledgehammer \leftrightarrow HOL(y)Hammer

- ▶ HOL4

- ▶ Type-classes

- ▶ Property of a universally quantified type

- ▶ Already in some Isabelle/HOL version of THF1

```
com_ring : $tType > $o
```

- ▶ Dependent types and intersection types

- ▶ Already in MPTP

```
![X : int, K : matrix(X)]: ...
```

```
![X : t1 & t2]: ...
```

- ▶ Universes

```
![X : int]: $type(X) : $tType
```

- ▶ General Π - and *Sigma*-types

```
![W : ![X]: X = X]: ...
```

- ▶ ...

Matching concepts across libraries

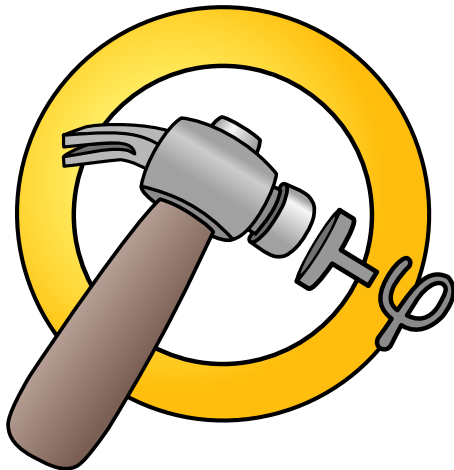
- ▶ Same concepts in different proof assistants
 - ▶ Problem for proof translation
 - ▶ Manually found 7-70 pairs
- ▶ Same properties
 - ▶ Patterns, like associativity, distributivity ...
 - ▶ Same algebraic structures do differ.
- ▶ Automatically finds 400 pairs of same concepts
 - ▶ In HOL Light, HOL4, Isabelle/HOL
 - ▶ Coq: so far only lists analyzed
- ▶ Proof advice can be universal?

Conclusion and Future work

- ▶ Hammer-systems
 - ▶ Until recently unappreciated by developers
 - ▶ A large number of top-level proofs found automatically
 - ▶ Try it!
- ▶ Interoperation between HOL Light, HOL4 and Isabelle/HOL
 - ▶ Cross-Prover Advice Service
- ▶ More logics, ITPs, ATPs, and more effective

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.
CoRR, abs/1310.2805, 2013.



C. Kaliszyk and J. Urban.
PRoCH: Proof reconstruction for HOL Light.
In M. P. Bonacina, editor, *CADE*, volume 7898 of *Lecture Notes in Computer Science*, pages 267–274. Springer, 2013.



C. Kaliszyk and J. Urban.
HOL(y)Hammer: Online ATP service for HOL Light.
Mathematics in Computer Science, 2014.
<http://dx.doi.org/10.1007/s11786-014-0182-0>.



C. Kaliszyk and J. Urban.
Learning-assisted automated reasoning with Flyspeck.
Journal of Automated Reasoning, 2014.
<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.
In S. Blazy, C. Paulin-Mohring, and D. Pichardie, editors, *Proc. of the 4th International Conference on Interactive Theorem Proving (ITP'13)*, volume 7998 of *LNCS*, pages 35–50. Springer, 2013.



C. Tankink, C. Kaliszyk, J. Urban, and H. Geuvers.
Formal mathematics on display: A wiki for Flyspeck.
In J. Carette, D. Aspinall, C. Lange, P. Sojka, and W. Windsteiger, editors, *MKM/Calcuemus/DML*, volume 7961 of *Lecture Notes in Computer Science*, pages 152–167. Springer, 2013.