Interactive Proofs: Automation, Computer Algebra, Presentation

Cezary Kaliszyk Josef Urban

University of Innsbruck Radboud University

August 25, 2014

Outline

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

HOL(y) Hammer

Learning-assisted automated reasoning for HOL Light

universität innsbruck

Radboud University Nijmegen

Request Advice:

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

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

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

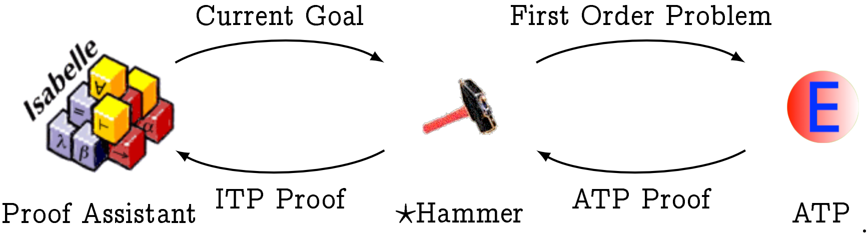
Result (3.81s): CONVEX_RELATIVE_INTERIOR POLYHEDRON_IMP_CONVEX

Replaying: SUCCESS

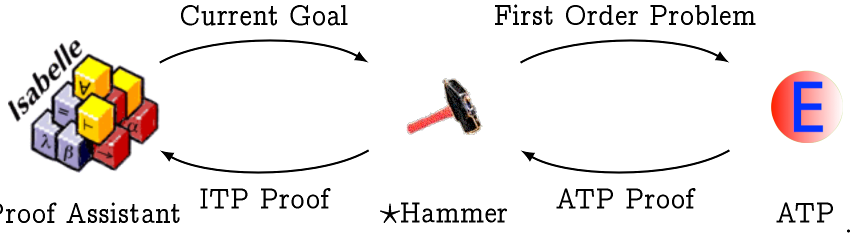
(0.29s):SIMP_TAC[POLYHEDRON_IMP_CONVEX;CONVEX_RELATIVE_INTERIOR]

Examples:

AI-ATP systems (⋆-Hammers)

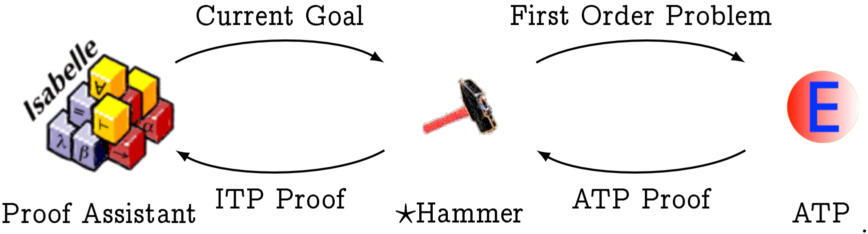


AI-ATP systems (⋆-Hammers)



How much can it do?

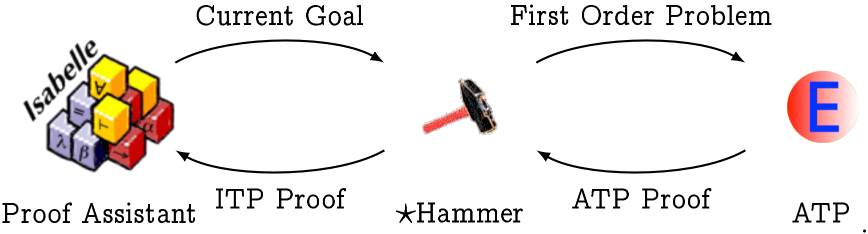
AI-ATP systems (⋆-Hammers)



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)

$$\approx 45\%$$

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 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)

Similar for TFF1, THF0, ...

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	-	-

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%

Formal Physics

Initial experiment

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

Outline

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)

$$\forall x.(f \;\; \text{diffl} \;\; g) \;\; x \quad \rightarrow \quad \text{diff} \;\; f = g$$

$$(f \;\; \text{diffl} \;\; g) \;\; x \quad \rightarrow \quad (\text{diff} \;\; f) \;\; x = g \;\; x$$

Computer Algebra Features

- Numerical approximation
  - Is not a computable function

- Evaluation (vs Verification)

$$\begin{aligned} \forall x.(f \ \ \mathrm{diffl} \ \ g) \ \ x & \ \ \rightarrow \ \ \mathrm{diff} \ \ f = g \\ (f \ \ \mathrm{diffl} \ \ g) \ \ x & \ \ \rightarrow \ \ (\mathrm{diff} \ \ f) \ \ x = g \ \ x \end{aligned}$$

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

Computer Algebra Features

- Numerical approximation
  - Is not a computable function
- Evaluation (vs Verification)

$$\begin{array}{rcl} \forall x.(f \;\; \text{diffl} \;\; g) \;\; x & \to & \text{diff} \;\; f = g \\ (f \;\; \text{diffl} \;\; g) \;\; x & \to & (\text{diff} \;\; f) \;\; x = g \;\; x \end{array}$$

- Partiality
  - On paper: $\ldots \frac{1}{x} \ldots$
  - In a PA: $\forall x \in \mathbb{C}. x \neq 0 \Rightarrow \ldots \dfrac{1}{x} \ldots$
- Real numbers
  - Either constructive and computable or classical

Satisfiability Modulo Theories

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

$$\forall x : num.\ F[x] \ \rightsquigarrow\ \forall x' : int.\ (x' \geq 0) \Rightarrow F[x']$$

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

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

- 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.
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/Calculemus/DML*, volume 7961 of *Lecture Notes in Computer Science*, pages 152–167. Springer, 2013.