Automated Theorem Proving for Metamath

Mario Carneiro, Chad E. Brown and Josef Urban

CMU, Pittsburgh and CTU in Prague

AITP 2023
September 6, 2023

Metamath

- proof assistant developed by Norman Megill in 1990
- set.mm - its largest library, 40338 theorems in ZFC
- analysis, topology, graph theory, number theory, Hilbert spaces, ...
- small but active community
- minimalist design vs a large library of advanced results
- no tactics and no hammers yet (this work)

Metamath on the Freek Wiedijk's 100 list

Table: Ranking of Proof Assistants as of 2023

Proof Assistant	Score
HOL Light	87
Isabelle	87
Coq	79
Lean	76
Metamath	74
Mizar	69
ProofPower	43
nqthm/ACL2	28
PVS	26
NuPRL/MetaPRL	8

Metamath and Bundling

- Metamath = “Metavariable Mathematics”
- Example: $\exists x.x = y$
- x and y may be the same:
- $\exists u.u = v$ vs $\exists u.u = u$
- “bundling:” one Metamath theorem represents many α-equivalence classes of theorems.

Metamath Zero

- MM0 between MM and HOL
- MM0 addresses the bundling issue: one theorem is one α-equivalence class of theorems in MM0
- Previous example splits into two MM0 theorems:
- $\exists x.x = y$ and $\exists x.x = x$
- Actually:

$$\text{wex } x \text{ (wceq (cv } x\text{) (cv } y\text{))}$$

Metamath Zero to Metamath-HOL

- The HOL version of metamath has three base types:
  - wff (formulas)
  - setvar (sets)
  - class (classes)
- As usual, we use λ to handle binding.
- Example: $(\forall x.\varphi) \to \varphi$
- MM0 version is wi (wall x φ) φ where x has type setvar and φ has type wff x.
- MM-HOL version is

$$\forall \varphi : \text{setvar} \to \text{wff}. \forall x : \text{setvar}. (\text{wi } (\text{wal } (\lambda x : \text{setvar}. \varphi\ x))\ (\varphi x))$$

Three translations from Metamath-HOL to TH0

- We have three translations to TH0.
- They differ in how much of the intended logical semantics we build in.
- In TH0 we reduce to two base types: o (formulas) and ι (sets).
- The type class translates to the type $\iota \to o$.
- In each case we use the Metamath proof to determine dependencies (what is needed to prove the theorem).
- We only include the dependencies in the corresponding TH0 problems.
- The three translations differ in how various MM-HOL primitives are translated.

Metamath-HOL primitives (v1 translation)

- There are many primitives in the MM-HOL source, e.g.:
  - wi : wff $\rightarrow$ wff $\rightarrow$ wff (implication)
  - wa : wff $\rightarrow$ wff $\rightarrow$ wff (conjunction)
  - wceq : class $\rightarrow$ class $\rightarrow$ wff (equality on classes)
  - wal : (setvar $\rightarrow$ wff) $\rightarrow$ wff (universal quantification)
  - wsb : (setvar $\rightarrow$ wff) $\rightarrow$ (setvar $\rightarrow$ wff) (substitution)
  - cab : (setvar $\rightarrow$ wff) $\rightarrow$ class (class abstraction)
- In the "v1" translation we leave all these as primitives, e.g.:
  - wi : $o \rightarrow o \rightarrow o$
  - wa : $o \rightarrow o \rightarrow o$
  - wceq : $(\iota \rightarrow o) \rightarrow (\iota \rightarrow o) \rightarrow o$
  - wal : $(\iota \rightarrow o) \rightarrow o$
  - wsb : $(\iota \rightarrow o) \rightarrow (\iota \rightarrow o)$
  - cab : $(\iota \rightarrow o) \rightarrow (\iota \rightarrow o)$
- Note that all the v1 TH0 problems will know about "wi" will be the dependencies of the problem that mention wi.

Metamath-HOL primitives (v2 translation)

- There are many primitives in the MM-HOL source, e.g.:
  - wi : wff $\to$ wff $\to$ wff (implication)
  - wa : wff $\to$ wff $\to$ wff (conjunction)
  - wceq : class $\to$ class $\to$ wff (equality on classes)
  - wal : (setvar $\to$ wff) $\to$ wff (universal quantification)
  - wsb : (setvar $\to$ wff) $\to$ (setvar $\to$ wff) (substitution)
  - cab : (setvar $\to$ wff) $\to$ class (class abstraction)
- In the “v2” translation we translate logical operators as their TH0 counterparts.
  - wi translates to $\lambda pq : o.p \to q$.
  - wa translates to $\lambda pq : o.p \wedge q$.
  - wceq translates to $\lambda XY : \iota \to o.X = Y$.
  - wal translates to $\lambda p : \iota \to o.\forall x : \iota.p\, x$.
- The other primitives are left as in the v1 translation:
  - wsb : $(\iota \to o) \to (\iota \to o)$
  - cab : $(\iota \to o) \to (\iota \to o)$
- Note that given a v2 TH0 problem, an ATP can reason directly about wi as implication.

Metamath-HOL primitives (v3 translation)

- There are many primitives in the MM-HOL source, e.g.:
  - wi : wff $\to$ wff $\to$ wff (implication)
  - wa : wff $\to$ wff $\to$ wff (conjunction)
  - wceq : class $\to$ class $\to$ wff (equality on classes)
  - wal : (setvar $\to$ wff) $\to$ wff (universal quantification)
  - wsb : (setvar $\to$ wff) $\to$ (setvar $\to$ wff) (substitution)
  - cab : (setvar $\to$ wff) $\to$ class (class abstraction)
- In the "v3" translation modifies the v2 translation to also interpret a number of other primitives.
  - wsb and cab both translate to $\lambda X : \iota \to o.X$.

Translating Metamath-HOL to First-Order Class Theory

- By treating set variables as objects, we can translate Metamath-HOL into a first-order class theory.
  - Translate wff to a first-order term (a class) and use a predicate p to determine if the term corresponding to the wff is “true.”
  - Translate a set variable x_i in a context $x_1, \ldots, x_n$ as a constant $\mathrm{var}^{\mathrm{i}}_{\mathrm{n}}$.
  - Translate classes as first-order terms.
- We do not include properties of class theory in the first-order ATP problem.
- So: often MM-HOL theorems will translate to FO non-theorems.
- But: All FO translations of MM-HOL yield Horn clauses
- Helps with proof reconstruction (see later)

Higher-order ATP Benchmark

- `https://github.com/ai4reason/mm-atp-benchmark`
- The three HO versions for the re-proving (small/bushy) problems
- For v3 we also provide the large (hammering/chainy) problems
- The 40338 Metamath theorems expand (via MM0) to 40556 THF theorems/problems
- The 218 extra theorems are those used in their α-degenerate form later in the library
- A new source of problems for evaluating and improving higher-order ATPs

HO ATP Evaluation

System	mode	version	time (s)	solved
Z	portfolio	v3	280	25420
Z	portfolio	v2	280	24959
V	portfolio	v3	280	23555
Z	portfolio	v2	140	23518
V	portfolio	v3	120	22976
V	portfolio	v3	60	21123
E	portfolio	v2	60	21001
E	portfolio	v3	60	20799
E	portfolio	v2	10	20352
E	strat. f17	v3	120	19782
E	strat. f17	v2	10	19624
V	portfolio	v2	60	18482
Z	fo-complete-basic	v2	10	17295
V	portfolio	v2	10	17160
Z	ho-pragmatic	v2	10	16115
E	portfolio	v1	10	11456

Table: The complete runs of the systems on the benchmark, ordered by performance. Z is Zipperposition, V is Vampire and E is E.

HO ATP Evaluation - Greedy Portfolio

System	mode	version	time (s)	added	sum
Z	portfolio	v3	280	25420	25420
V	portfolio	v3	600	960	26380
V	portfolio	v3	1200	415	26795
E	portfolio	v3	600	279	27074
Z	portfolio	v2	280	124	27198

Table: The top 5 methods in the greedy sequence. Note that we use different (and also high) time limits and that the high-time runs are only done on previously unsolved problems.

Example: Arithmetic and Geometric Means

- `amgm2d`:

$$(A \cdot B)^{\frac{1}{2}} \leq \frac{A + B}{2}$$

- `amgm3d`:

$$(A \cdot B \cdot C)^{\frac{1}{3}} \leq \frac{A + B + C}{3}$$

- `amgm4d`:

$$(A \cdot B \cdot C \cdot D)^{\frac{1}{4}} \leq \frac{A + B + C + D}{4}$$

- Zipperposition and E can prove the v3 version of each using the following main dependency:
- `amgmlem`:

$$(\Sigma^M F)^{\frac{1}{|A|}} \leq \frac{\Sigma^C F}{|A|}$$

  - A finite set and
  - F function from A to positive reals.

FO ATP Evaluation

- Vampire, E and Prover9 run for 60 seconds
- Vampire: 15938, E: 15136, P9: 14693
- Likely demonstrates the inefficiency of the current FO encoding compared to the more advanced HO encodings
- Practically none of the standard logical connectives are mapped in a shallow way to their FO TPTP counterparts
- The V, E and P9 performance is similar likely because the problems are Horn and small

Premise Selection

- On HO v3, Vampire-LTB: 8509, Vampire-HOL: 4013
- Premise selection with k-NN:

Premises	10	20	40	80	120	160	240
V-thf v3	9112	10078	11060	11863	12043	11997	11582
V-fof v1	2600	4239	6294	8366	9416	9875	10352

Proof Reconstruction

- Prover9 can produce IVY proof objects
  - input (translation of a dependency for the theorem; or part of negation of conclusion)
  - instantiate
  - resolve
  - propositional (e.g., for factoring clauses with repeated literals)
- Note: Each IVY step preserves clauses being Horn
- When translating back to Metamath, using a Horn clause corresponds to applying a dependency in a straightforward way.
- The instantiate steps give the substitution arguments to each Metamath theorem step

Proof Objects

Problem	mercolem6	tgbtwnconn1lem1	hdmap14lem9	isoas	lclkrlem2a
IVY	674	480	392	375	316
Problem	mercolem6	mercolem2	merlem5	mercolem7	minimp_sylsimp
Metamath	5660830	849	77	50	45

Table: Length of the longest proof objects in IVY steps and Metamath lines.

Proof Blowup

- An outlier is `mercolem6`, which is a lemma in the proof that Meredith's axiom

$$((\varphi \to \psi) \to (\bot \to \chi) \to \theta) \to (\theta \to \varphi) \to \tau \to \eta \to \varphi$$

  is complete for propositional logic
- Prover9 is able to return a proof with only 674 lines
- it balloons to 5 660 830 lines after Metamath reconstruction (over 7 times the size of `set.mm`)
- This is because if an IVY proof step is applied multiple times with different substitution instances, the subproof is monomorphized for each substitution
- In practice, a human would split out a lemma for this
  - In fact, the name `mercolem6` indicates that this is lemma 6 of something, so this technique is already being used here.
- but our prover structurally cannot produce proofs with lemmas, so the different cost model between IVY and metamath proofs can produce these pathologies

Packaging

- The full hammer system is available at `https://github.com/digama0/mm-hammer`
- The installation script installs all the dependencies (premise selector, Vampire, Prover9)
- The user passes a Metamath theorem statement and it produces an output compressed proof object suitable for insertion in a Metamath database