Learning and Reasoning over Big Proof Corpora

Josef Urban

Czech Technical University in Prague

Big Proof, May 30, 2019, Edinburgh

European Research Council
Established by the European Commission

Outline

Motivation, Learning vs. Reasoning

Learning of Theorem Proving

Demos

High-level Reasoning Guidance: Premise Selection

Low Level Guidance of Theorem Provers

Mid-level Reasoning Guidance

Autoformalization

How Do We Automate Math and Science?

- What is mathematical and scientific thinking?
- Pattern-matching, analogy, induction from examples
- Deductive reasoning
- Complicated feedback loops between induction and deduction
- Using a lot of previous knowledge - both for induction and deduction

- We need to develop such methods on computers
- Are there any large corpora suitable for nontrivial deduction?
- Yes! Large libraries of formal proofs and theories
- So let's develop strong AI on them!

History, Motivation, AI/TP/ML/DL

- Intuition vs Formal Reasoning – Poincaré vs Hilbert, Science & Method
- Turing's 1950 paper: Learning Machines, learn Chess?, undecidability??
- Lenat, Langley, etc: manually-written heuristics, learn Kepler laws,...
- Denzinger, Schulz, Goller, Fuchs – late 90's, ATP-focused:
- *Learning from Previous Proof Experience*
- My MSc (1998): Try ILP to learn rules and heuristics from IMPS/Mizar
- Since: Use large formal math (Big Proof) corpora: Mizar, Isabelle, HOL
- ... to combine/develop symbolic/statistical deductive/inductive ML/TP/AI
- ... hammer-style methods, feedback loops, internal guidance, ...
- More details – AGI'18 keynote: `https://slideslive.com/38909911/no-one-shall-drive-us-from-the-semantic-ai-paradise-of-computerunderstandable-math-and-science`
- AI vs DL: Ben Goertzel's 2018 Prague talk: `https://youtu.be/Zt2HSTuGBn8`

Using Learning to Guide Theorem Proving

- **high-level**: pre-select lemmas from a large library, give them to ATPs
- **high-level**: pre-select a good ATP strategy/portfolio for a problem
- **high-level**: pre-select good *hints* for a problem, use them to guide ATPs
- **low-level**: guide every inference step of ATPs (tableau, superposition)
- **low-level**: guide every kernel step of LCF-style ITPs
- **mid-level**: guide application of tactics in ITPs
- **mid-level**: invent suitable ATP strategies for classes of problems
- **mid-level**: invent suitable conjectures for a problem
- **mid-level**: invent suitable concepts/models for problems/theories
- **proof sketches**: explore stronger/related theories to get proof ideas
- **theory exploration**: develop interesting theories by conjecturing/proving
- **feedback loops**: (dis)prove, learn from it, (dis)prove more, learn more, ...
- **autoformalization**: (semi-)automate translation from LaTeX to formal
- ...

Large Datasets

- Mizar / MML / MPTP – since 2003
- MPTP Challenge (2006), MPTP2078 (2011), Mizar40 (2013)
- Isabelle (and AFP) – since 2005
- Flyspeck (including core HOL Light and Multivariate) – since 2012
- HOL4 – since 2014, CakeML – 2017, GRUNGE – 2019
- Coq – since 2013/2016
- ACL2 – 2014?
- Lean?, Stacks?, Arxiv?, ProofWiki?, ...

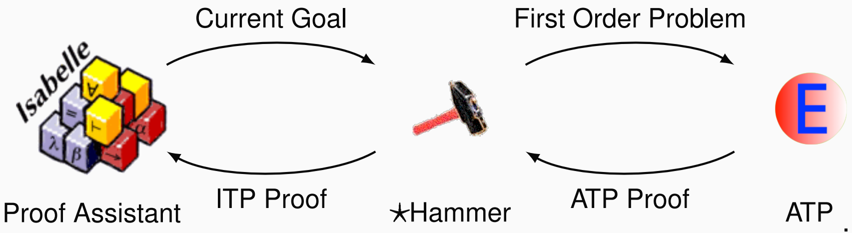
Demos

- Hammering Mizar: `http://grid01.ciirc.cvut.cz/~mptp/out4.ogv`
- TacticToe on HOL4: `http://grid01.ciirc.cvut.cz/~mptp/tactictoe_demo.ogv`
- Inf2formal over HOL Light: `http://grid01.ciirc.cvut.cz/~mptp/demo.ogv`

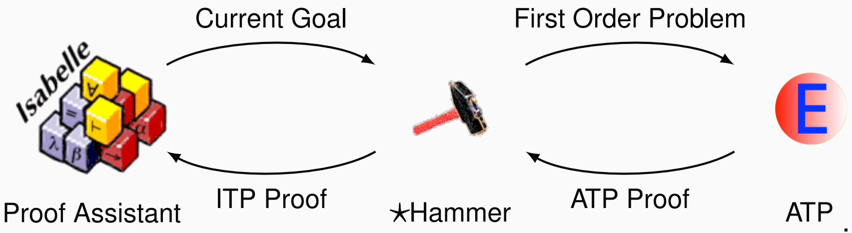
High-level ATP guidance: Premise Selection

- Early 2003: Can existing ATPs be used over the freshly translated Mizar library?
- About 80000 nontrivial math facts at that time – impossible to use them all
- Is good premise selection for proving a new conjecture possible at all?
- Or is it a mysterious power of mathematicians? (Penrose)
- Today: Premise selection is not a mysterious property of mathematicians!
- Reasonably good algorithms started to appear (more below).
- Will extensive human (math) knowledge get obsolete?? (cf. Watson, Debater, etc)

Today's AI-ATP systems (⋆-Hammers)

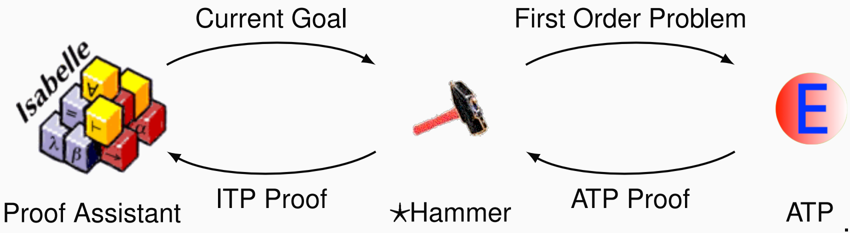


Today's AI-ATP systems ($\star$-Hammers)



How much can it do?

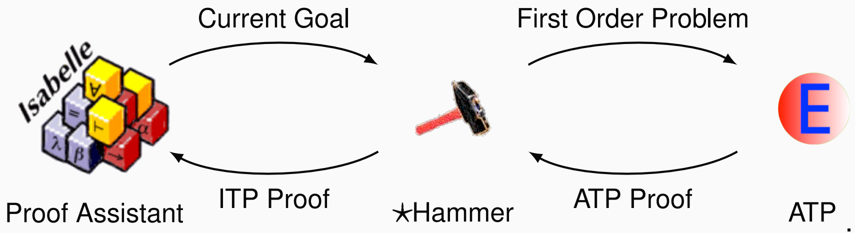
Today's AI-ATP systems (⋆-Hammers)



How much can it do?

- Mizar / MML – MizAR
- Isabelle (Auth, Jinja) – Sledgehammer
- Flyspeck (including core HOL Light and Multivariate) – HOL(y)Hammer
- HOL4 (Gauthier and Kaliszyk)
- CoqHammer (Czajka and Kaliszyk) - about 40% on Coq standard library

Today's AI-ATP systems (⋆-Hammers)



How much can it do?

- Mizar / MML – MizAR
- Isabelle (Auth, Jinja) – Sledgehammer
- Flyspeck (including core HOL Light and Multivariate) – HOL(y)Hammer
- HOL4 (Gauthier and Kaliszyk)
- CoqHammer (Czajka and Kaliszyk) - about 40% on Coq standard library

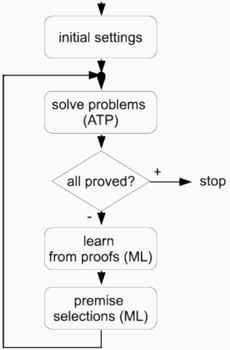
$\approx$ 45% success rate

Recent Improvements and Additions

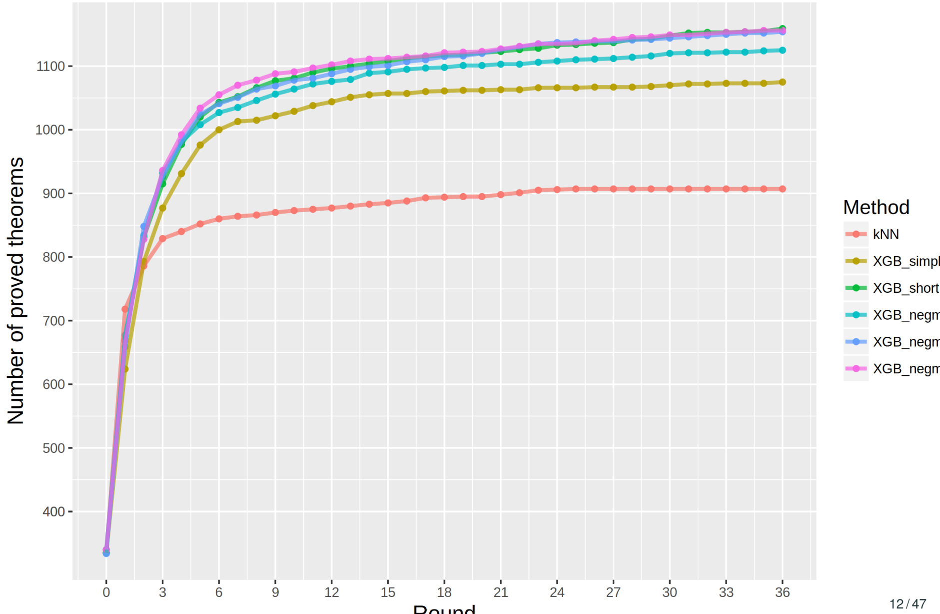
- Semantic features encoding term matching/unification [IJCAI'15]
- Distance-weighted k-nearest neighbor, LSI, boosted trees (XGBoost)
- Matching and transferring concepts and theorems between libraries (Gauthier & Kaliszyk) – allows “superhammers”, conjecturing, and more
- Lemmatization – extracting and considering millions of low-level lemmas
- First useful CoqHammer (Czajka & Kaliszyk 2016), 40%–50% reconstruction/ATP success on the Coq standard library
- Neural sequence models, definitional embeddings (with Google Research)
- Hammers combined with statistical tactical search: TacticToe (Gauthier - HOL4)
- Learning in binary setting from many alternative proofs
- Negative/positive mining (ATPBoost - Piotrowski & JU, 2018)

High-level feedback loops – MALARea

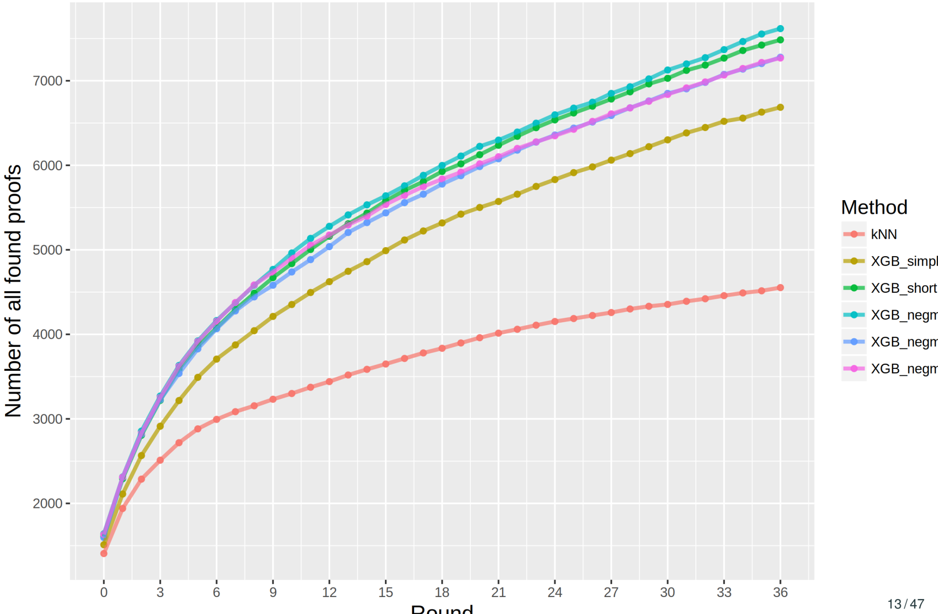
- Machine Learner for Autom. Reasoning (2006) – infinite hammering
- feedback loop interleaving ATP with learning premise selection
- both syntactic and semantic features for characterizing formulas:
- evolving set of finite (counter)models in which formulas evaluated
- winning AI/ATP benchmarks (MPTPChallenge, CASC 2008/12/13/18)
- ATPBoost (Piotrowski) - recent incarnation focusing on multiple proofs

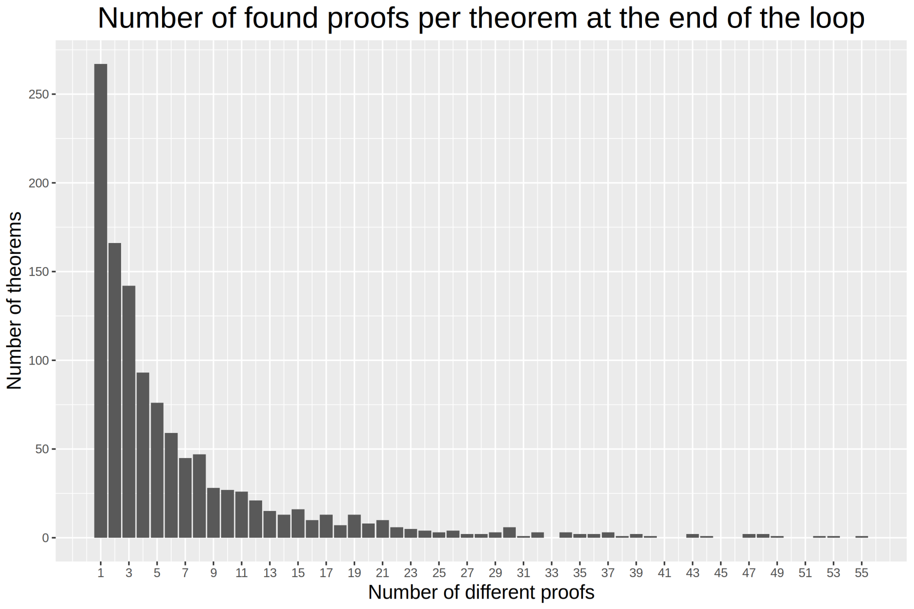


Prove-and-learn loop on MPTP2078 data set



Prove-and-learn loop on MPTP2078 data set





Number of found proofs per theorem at the end of the loop
Number of theorems
Number of different proofs
0
50
100
150
200
250
1
3
5
7
9
11
13
15
17
19
21
23
25
27
29
31
33
35
37
39
41
43
45
47
49
51
53
55

Low-level: Statistical Guidance of Connection Tableau

- learn guidance of every clausal inference in connection tableau (leanCoP)
- set of first-order clauses, *extension* and *reduction* steps
- proof finished when all branches are closed
- a lot of nondeterminism, requires backtracking
- *Iterative deepening* used in leanCoP to ensure completeness
- good for learning – the tableau compactly represents the proof state

Clauses:

$c_1 : P(x)$

$c_2 : R(x, y) \lor \neg P(x) \lor Q(y)$

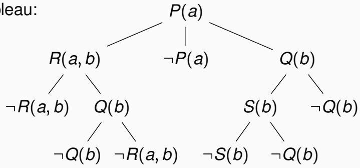
$c_3 : S(x) \lor \neg Q(b)$

$c_4 : \neg S(x) \lor \neg Q(x)$

$c_5 : \neg Q(x) \lor \neg R(a, x)$

$c_6 : \neg R(a, x) \lor Q(x)$

Closed Connection Tableau:



Statistical Guidance of Connection Tableau

- **MaLeCoP** (2011): first prototype Machine Learning Connection Prover
- extension rules chosen by naive Bayes trained on good decisions
- training examples: tableau features plus the name of the chosen clause
- initially slow: off-the-shelf learner 1000 times slower than raw leanCoP
- 20-time search shortening on the MPTP Challenge
- second version: 2015, with C. Kaliszyk
- both prover and naive Bayes in OCAML, fast indexing
- Fairly Efficient MaLeCoP = **FEMaLeCoP**
- 15% improvement over untrained leanCoP on the MPTP2078 problems
- using iterative deepening - enumerate shorter proofs before longer ones

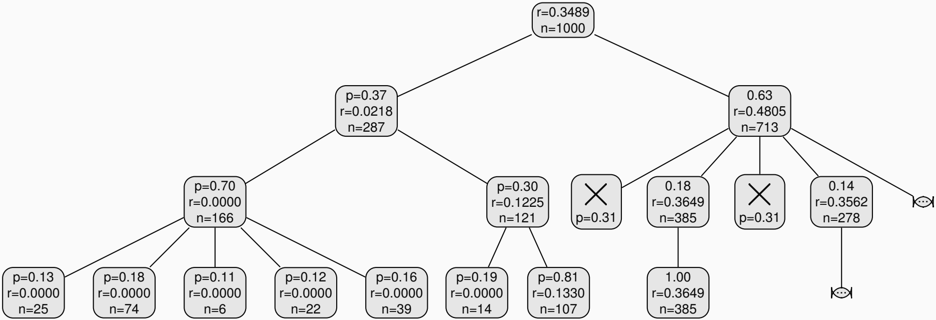
Statistical Guidance of Connection Tableau – rlCoP

- 2018: stronger learners via C interface to OCAML (boosted trees)
- remove iterative deepening, the prover can go arbitrarily deep
- added Monte-Carlo Tree Search (MCTS)
- MCTS search nodes are sequences of clause application
- a good heuristic to explore new vs exploit good nodes:

$$\frac{w_i}{n_i} + c \cdot p_i \cdot \sqrt{\frac{\ln N}{n_i}}$$ (UCT - Kocsis, Szepesvari 2006)

- learning both *policy* (clause selection) and *value* (state evaluation)
- clauses represented not by names but also by features (generalize!)
- binary learning setting used: | proof state | clause features |
- mostly term walks of length 3 (trigrams), hashed into small integers
- many iterations of proving and learning

Tree Example



Statistical Guidance of Connection Tableau – rlCoP

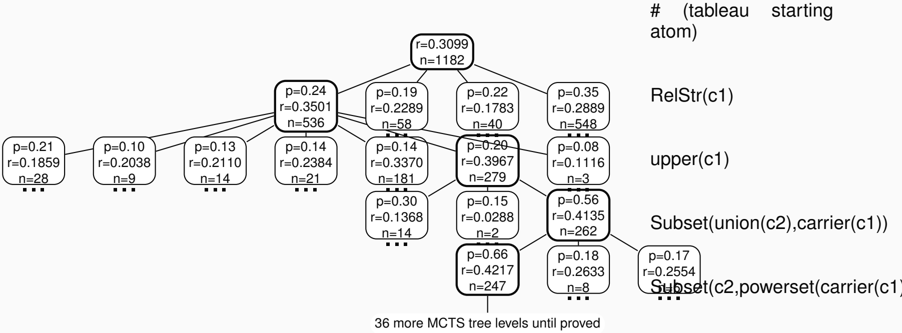
- On 32k Mizar40 problems using 200k inference limit
- nonlearning CoPs:

System	leanCoP	bare prover	rlCoP no policy/value (UCT only)
Training problems proved	10438	4184	7348
Testing problems proved	**1143**	431	804
Total problems proved	11581	4615	8152

- rlCoP with policy/value after 5 proving/learning iters on the training data
- $1624/1143 = 42.1$% improvement over leanCoP on the testing problems

Iteration	1	2	3	4	5	6	7	8
Training proved	12325	13749	14155	14363	14403	14431	14342	**14498**
Testing proved	1354	1519	1566	1595	**1624**	1586	1582	1591

More trees

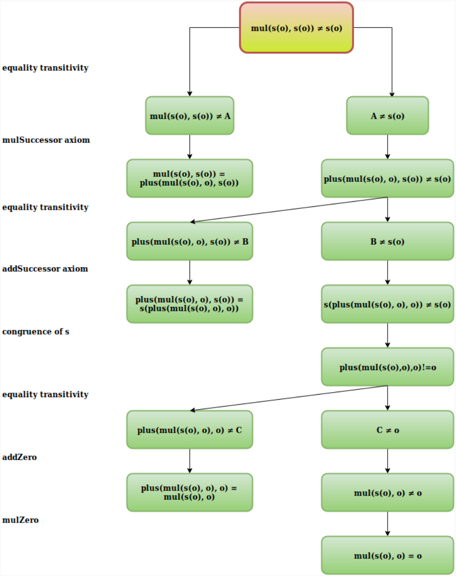


# (tableau starting atom)
RelStr(c1)
upper(c1)
Subset(union(c2),carrier(c1))
Subset(c2,powerset(carrier(c1)
r=0.3099 n=1182
p=0.24 r=0.3501 n=536
p=0.19 r=0.2289 n=58
p=0.22 r=0.1783 n=40
p=0.35 r=0.2889 n=548
p=0.21 r=0.1859 n=28
p=0.10 r=0.2038 n=9
p=0.13 r=0.2110 n=14
p=0.14 r=0.2384 n=21
p=0.14 r=0.3370 n=181
p=0.20 r=0.3967 n=279
p=0.08 r=0.1116 n=3
p=0.30 r=0.1368 n=14
p=0.15 r=0.0288 n=2
p=0.56 r=0.4135 n=262
p=0.66 r=0.4217 n=247
p=0.18 r=0.2633 n=8
p=0.17 r=0.2554
36 more MCTS tree levels until proved

Recent Variations – FLoP, RNN

- FLoP – Finding Longer Proofs (Zsombori et al, 2019)
- Curriculum Learning used for connection tableau over Robinson Arithmetic
- addition and multiplication learned perfectly from $1 * 1 = 1$
- headed towards learning algorithms/decision procedures from math data
- currently black-box, combinations with symbolic methods (ILP) our next target
- Using RNNs for better tableau encoding, prediction of actions ...
- ... even guessing (decoding) next tableau literals (Piotrowski & JU, 2019)

FLoP Training Proof



Side Note on Symbolic Learning with NNs

- Recurrent NNs with attention recently very good at the inf2formal task
- Experiments with using them for symbolic rewriting (Piotrowski et. all)
- We can learn rewrite rules from sufficiently many data
- 80-90% on algebra datasets, 70-99% on normalizing polynomials
- again, complements symbolic methods like ILP that suffer if too much data

Side Note on Symbolic Learning with NNs

Table: Examples in the AIM data set.

Rewrite rule:	Before rewriting:	After rewriting:
`b(s(e,v1),e)=v1`	`k(b(s(e,v1),e),v0)`	`k(v1,v0)`
`o(V0,e)=V0`	`t(v0,o(v1,o(v2,e)))`	`t(v0,o(v1,v2))`

Table: Examples in the polynomial data set.

Before rewriting:	After rewriting:
`(x * (x + 1)) + 1`	`x ^ 2 + x + 1`
`(2 * y) + 1 + (y * y)`	`y ^ 2 + 2 * y + 1`
`(x + 2) * ((2 * x) + 1) + (y + 1)`	`2 * x ^ 2 + 5 * x + y + 3`

Side Note on Model Learning with NNs

- Smolik 2019 (MSc thesis): modelling mathematical structures with NNs
- NNs reasonably learn cyclic groups and their extensions
- ... so far struggle in learning bigger permutation groups
- Plan: learn composition/variation of complicated math structures
- Use for model-style evaluation of formulas, conjectures, etc. – similarly to the finite models in Malarea, etc.

Statistical Guidance the Given Clause in E Prover

- harder for learning than tableau
- the proof state are two large heaps of clauses *processed/unprocessed*
- 2017: ENIGMA - manual feature engineering (Jakubuv & JU 2017)
- 2017: Deep guidance (neural nets) (Loos et al. 2017)
- both learn on E's proof search traces, put classifier in E
- positive examples: given clauses used in the proof
- negative examples: given clauses not used in the proof
- ENIGMA: fast feature extraction followed by fast/sparse linear classifier
- about 80% improvement on the AIM benchmark
- Deep guidance: convolutional nets - no feature engineering but slow
- ENIGMA-NG: better features and ML, gradient-boosted trees, tree NNs
- NNs made competitive in real-time, boosted trees still best

Feedback loop for ENIGMA on Mizar data

- Similar to rlCoP - interleave proving and learning of ENIGMA guidance
- Done on 57880 Mizar problems very recently
- Ultimately a 70% improvement over the original strategy
- Example proof found by ENIGMA: `http://grid01.ciirc.cvut.cz/~mptp/7.13.01_4.181.1147/html/knaster#T21`

	$\mathcal{S}$	$\mathcal{S} \odot \mathcal{M}_9^0$	$\mathcal{S} \oplus \mathcal{M}_9^0$	$\mathcal{S} \odot \mathcal{M}_9^1$	$\mathcal{S} \oplus \mathcal{M}_9^1$	$\mathcal{S} \odot \mathcal{M}_9^2$	$\mathcal{S} \oplus \mathcal{M}_9^2$	$\mathcal{S} \odot \mathcal{M}_9^3$	$\mathcal{S} \oplus \mathcal{M}_9^3$
solved	14933	16574	20366	21564	22839	22413	23467	22910	23753
$\mathcal{S}\%$	+0%	+10.5%	+35.8%	+43.8%	+52.3%	+49.4%	+56.5%	+52.8%	+58.4
$\mathcal{S}+$	+0	+4364	+6215	+7774	+8414	+8407	+8964	+8822	+9274
$\mathcal{S}-$	-0	-2723	-782	-1143	-508	-927	-430	-845	-454

	$\mathcal{S} \odot \mathcal{M}_{12}^3$	$\mathcal{S} \oplus \mathcal{M}_{12}^3$	$\mathcal{S} \odot \mathcal{M}_{16}^3$	$\mathcal{S} \oplus \mathcal{M}_{16}^3$
solved	24159	24701	25100	25397
$\mathcal{S}\%$	+61.1%	+64.8%	+68.0%	+70.0%
$\mathcal{S}+$	+9761	+10063	+10476	+10647
$\mathcal{S}-$	-535	-295	-309	-183

ENIGMA Proof Example – Knaster

```
theorem Th21:
  ex a st a is_a_fixpoint_of f
proof
  set H = {h where h is Element of L: h [= f.h};
  set fH = {f.h where h is Element of L: h [= f.h};
  set uH = "\/"(H, L);
  set fuH = "\/"(fH, L);
  take uH;
  now
    let fh be Element of L;
    assume fh in fH;
    then consider h being Element of L such that
A1: fh = f.h and
A2: h [= f.h;
    h in H by A2;
    then h [= uH by LATTICE3:38;
    hence fh [= f.uH by A1,QUANTAL1:def 12;
  end;
  then fH is_less_than f.uH by LATTICE3:def 17;
  then
A3: fuH [= f.uH by LATTICE3:def 21;
  now
    let a be Element of L;
    assume a in H;
    then consider h being Element of L such that
A4: a = h & h [= f.h;
    reconsider fh = f.h as Element of L;
    take fh;
    thus a [= fh & fh in fH by A4;
  end;
  then uH [= fuH by LATTICE3:47;
  then
A5: uH [= f.uH by A3,LATTICES:7;
  then f.uH [= f.(f.uH) by QUANTAL1:def 12;
  then f.uH in H;
  then f.uH [= uH by LATTICE3:38;
  hence uH = f.uH by A5,LATTICES:8;
end;
```

ProofWatch: Statistical/Semantic Guidance of E (Goertzel et al. 2018)

- Bob Veroff's *hints* method used for Prover9/AIM
- solve many easier/related problems
- load their useful lemmas on the *watchlist* (kind of conjecturing)
- boost inferences on clauses that subsume a watchlist clause
- watchlist parts are fast thinking, bridged by standard (slow) search
- ProofWatch (2018): load many proofs separately
- dynamically boost those that have been covered more
- needed for heterogeneous ITP libraries
- statistical: watchlists chosen using similarity and usefulness
- semantic/deductive: dynamic guidance based on exact proof matching
- results in better vectorial characterization of saturation proof searches

ProofWatch: Statistical/Symbolic Guidance of E

```
theorem Th36: :: YELLOW_5:36
for L being non empty Boolean RelStr for a, b being Element of L
holds ( 'not' (a "\/" b) = ('not' a) "/\" ('not' b)
        & 'not' (a "/\" b) = ('not' a) "\/" ('not' b) )
```

- De Morgan's laws for Boolean lattices
- guided by 32 related proofs resulting in 2220 watchlist clauses
- 5218 given clause loops, resulting ATP proof is 436 clauses
- 194 given clauses match the watchlist and 120 (61.8%) used in the proof
- most helped by the proof of WAYBEL_1:85 – done for lower-bounded Heyting

```
theorem :: WAYBEL_1:85
for H being non empty lower-bounded RelStr st H is Heyting holds
for a, b being Element of H holds
  'not' (a "/\" b) >= ('not' a) "\/" ('not' b)
```

ProofWatch: Vectorial Proof State

Final state of the proof progress for the 32 proofs guiding `YELLOW_5:36`

0	0.438	42/96	1	0.727	56/77	2	0.865	45/52	3	0.360	9/25
4	0.750	51/68	5	0.259	7/27	6	0.805	62/77	7	0.302	73/242
8	0.652	15/23	9	0.286	8/28	10	0.259	7/27	11	0.338	24/71
12	0.680	17/25	13	0.509	27/53	14	0.357	10/28	15	0.568	25/44
16	0.703	52/74	17	0.029	8/272	18	0.379	33/87	19	0.424	14/33
20	0.471	16/34	21	0.323	20/62	22	0.333	7/21	23	0.520	26/50
24	0.524	22/42	25	0.523	45/86	26	0.462	6/13	27	0.370	20/54
28	0.411	30/73	29	0.364	20/55	30	0.571	16/28	31	0.357	10/28

EnigmaWatch: ProofWatch used with ENIGMA

- Use the proof completion ratios as features for characterizing proof state
- Instead of just static conjecture features - the vectors evolve
- Feed them to ML systems along with other features
- Relatively good improvement
- To be extended in various ways

EnigmaWatch: ProofWatch used with ENIGMA

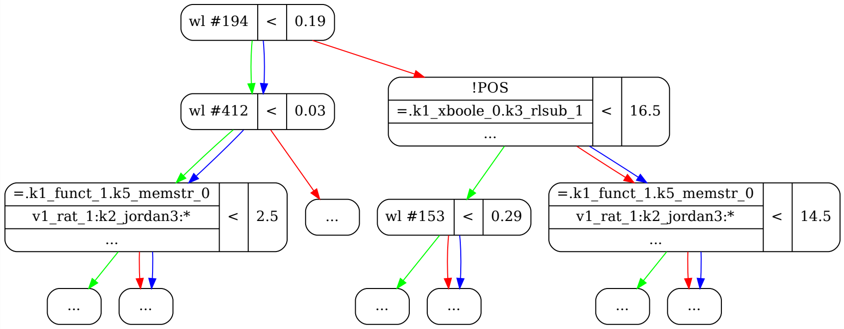
Baseline	Mean	Var	Corr	Rand	Baseline ∪ Mean	Total
1140	1357	1345	1337	1352	1416	1483

Table: ProofWatch evaluation: Problems solved by different versions.

loop	ENIGMA	Mean	Var	Corr	Rand	ENIGMA ∪ Mean	Total
0	1557	1694	1674	1665	1690	1830	1974
1	1776	1815	1812	1812	1847	1983	2131
2	1871	1902	1912	1882	1915	2058	2200
3	1931	1954	1946	1920	1926	2110	2227

Table: ENIGMAWatch evaluation: Problems solved and the effect of looping.

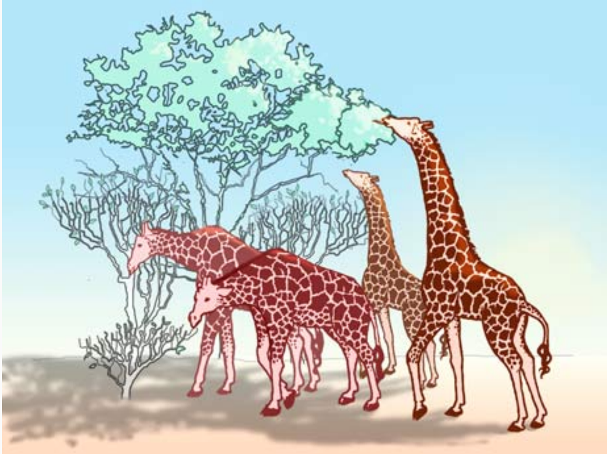
Example of an XGBoost decision tree



TacticToe: mid-level ITP Guidance (Gauthier et al.'18)

- learns from human tactical HOL4 proofs to solve new goals
- no translation or reconstruction needed
- similar to rlCoP: policy/value learning
- however much more technically challenging:
    - tactic and goal state recording
    - tactic argument abstraction
    - absolutization of tactic names
    - nontrivial evaluation issues
- policy: which tactic/parameters to choose for a current goal?
- value: how likely is this proof state succeed?
- 66% of HOL4 toplevel proofs in 60s (better than a hammer!)
- similar recent work for Isabelle (Nagashima 2018)
- work in progress for Coq (us, OpenAI) and HOL Light (us, Google)

More Mid-level guidance: BliStr: Blind Strategymaker



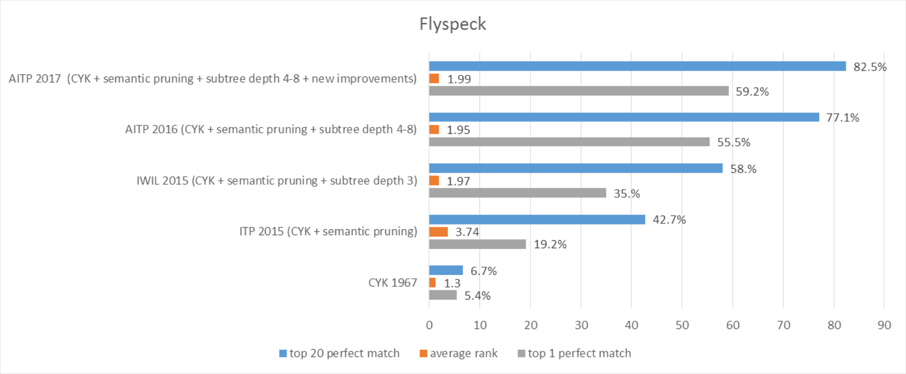
- The strategies are like giraffes, the problems are their food
- The better the giraffe specializes for eating problems unsolvable by others, the more it gets fed and further evolved

Autoformalization based on PCFG and semantics

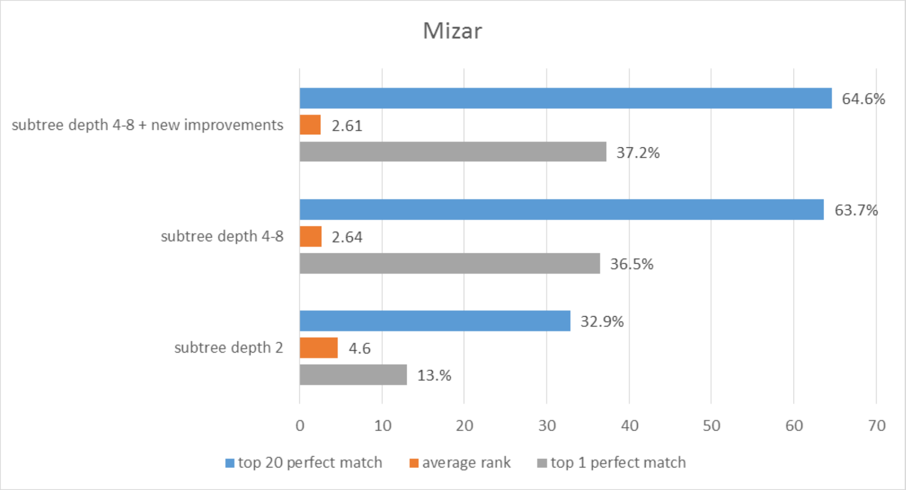
- "`sin ( 0 * x ) = cos pi / 2`"
- produces 16 parses
- of which 11 get type-checked by HOL Light as follows
- with all but three being proved by HOL(y)Hammer
- demo: `http://grid01.ciirc.cvut.cz/~mptp/demo.ogv`

```
sin (&0 * A0) = cos (pi / &2) where A0:real
sin (&0 * A0) = cos pi / &2 where A0:real
sin (&0 * &A0) = cos (pi / &2) where A0:num
sin (&0 * &A0) = cos pi / &2 where A0:num
sin (&(0 * A0)) = cos (pi / &2) where A0:num
sin (&(0 * A0)) = cos pi / &2 where A0:num
csin (Cx (&0 * A0)) = ccos (Cx (pi / &2)) where A0:real
csin (Cx (&0) * A0) = ccos (Cx (pi / &2)) where A0:real^2
Cx (sin (&0 * A0)) = ccos (Cx (pi / &2)) where A0:real
csin (Cx (&0 * A0)) = Cx (cos (pi / &2)) where A0:real
csin (Cx (&0) * A0) = Cx (cos (pi / &2)) where A0:real^2
```

Flyspeck Progress



First Mizar Results (100-fold Cross-validation)



Neural Autoformalization (Wang et al., 2018)

- generate about 1M Latex - Mizar pairs based on Bancerek's work
- train neural seq-to-seq translation models (Luong – NMT)
- evaluate on about 100k examples
- many architectures tested, some work much better than others
- very important latest invention: *attention* in the seq-to-seq models
- more data very important for neural training – our biggest bottleneck (you can help!)
- Recent addition: unsupervised methods (Lample et all 2018) – no need for aligned data!

Neural Autoformalization data

Rendered LaTeX	If $X \subseteq Y \subseteq Z$, then $X \subseteq Z$.
Mizar	`X c= Y & Y c= Z implies X c= Z;`
Tokenized Mizar	`X c= Y & Y c= Z implies X c= Z ;`
LaTeX	`If $X \subseteq Y \subseteq Z$, then $X \subseteq Z$.`
Tokenized LaTeX	`If $ X \subseteq Y \subseteq Z $ , then $ X \subseteq Z $ .`

Neural Autoformalization results

Parameter	Final Test Perplexity	Final Test BLEU	Identical Statements (%)	Identical No-overlap (%)
128 Units	3.06	41.1	40121 (38.12%)	6458 (13.43%)
256 Units	1.59	64.2	63433 (60.27%)	19685 (40.92%)
512 Units	1.6	**67.9**	66361 (63.05%)	21506 (44.71%)
1024 Units	**1.51**	61.6	**69179 (65.73%)**	**22978 (47.77%)**
2048 Units	2.02	60	59637 (56.66%)	16284 (33.85%)

Neural Fun – Performance after Some Training

Rendered LaTeX	Suppose s_8 is convergent and s_7 is convergent . Then $\mathop{\rm lim}(s_8 + s_7) = \mathop{\rm lim} s_8 + \mathop{\rm lim} s_7$
Input LaTeX	`Suppose $ { s _ { 8 } } $ is convergent and $ { s _ { 7 } } $ is convergent . Then $ \mathop { \rm lim } ( { s _ { 8 } } { + } { s _ { 7 } } } ) \mathrel { = } \mathop { \rm lim } { s _ { 8 } } { + } \mathop { \rm lim } { s _ { 7 } } $ .`
Correct	`seq1 is convergent & seq2 is convergent implies lim ( seq1 + seq2 ) = ( lim seq1 ) + ( lim seq2 ) ;`
Snapshot-1000	`x in dom f implies ( x * y ) * ( f | ( x | ( y | ( y | y ) ) ) ) = ( x | ( y | ( y | ( y | y ) ) ) ) ) ;`
Snapshot-2000	`seq is summable implies seq is summable ;`
Snapshot-3000	`seq is convergent & lim seq = 0c implies seq = seq ;`
Snapshot-4000	`seq is convergent & lim seq = lim seq implies seq1 + seq2 is convergent ;`
Snapshot-5000	`seq1 is convergent & lim seq2 = lim seq2 implies lim_inf seq1 = lim_inf seq2 ;`
Snapshot-6000	`seq is convergent & lim seq = lim seq implies seq1 + seq2 is convergent ;`
Snapshot-7000	`seq is convergent & seq9 is convergent implies lim ( seq + seq9 ) = ( lim seq ) + ( lim seq9 ) ;`

Unsupervised NMT Fun on Short Formulas

```
len <* a *> = 1 ;                    len <* a *> = 1 ;
assume i < len q ;                   i < len q ;
len <* q *> = 1 ;                    len <* q *> = 1 ;
s = apply ( v2 , v1 ast t ) ;        s = apply ( v2 , v1 ) . t ;
s . ( i + 1 ) = tt . ( i + 1 )       s . ( i + 1 ) = tau1 . ( i + 1 )
1 + j <= len v2 ;                    1 + j <= len v2 ;
1 + j + 0 <= len v2 + 1 ;            1 + j + 0 <= len v2 + 1 ;
let i be Nat ;                       i is_at_least_length_of p ;
assume v is_applicable_to t ;        not v is applicable ;
let t be type of T ;                 t is_orientedpath_of v1 , v2 , T ;
a ast t in downarrow t ;             a *' in downarrow t ;
t9 in types a ;                      t `2 in types a ;
a ast t <= t ;                       a *' <= t ;
A is_applicable_to t ;               A is applicable ;
Carrier ( f ) c= B                   support ppf n c= B
u in B or u in { v } ;               u in B or u in { v } ;
F . w in w & F . w in I ;            F . w in F & F . w in I ;
GG . y in rng HH ;                   G0 . y in rng ( H1 ./. y ) ;
a * L = Z_ZeroLC ( V ) ;             a * L = ZeroLC ( V ) ;
not u in { v } ;                     u >> v ;
u <> v ;                             u <> v ;
v - w = v1 - w1 ;                    vw = v1 - w1 ;
v + w = v1 + w1 ;                    v + w = v1 + w1 ;
x in A & y in A ;                    assume [ x , y ] in A ;
```

Acknowledgments

- Prague Automated Reasoning Group `http://arg.ciirc.cvut.cz/`:
  - Jan Jakubuv, Chad Brown, Martin Suda, Karel Chvalovsky, Bob Veroff, Zar Goertzel, Bartosz Piotrowski, Lasse Blaauwbroek, Martin Smolik, Jiri Vyskocil, Petr Pudlak, David Stanovsky, Krystof Hoder, ...
- HOL(y)Hammer group in Innsbruck:
  - Cezary Kaliszyk, Thibault Gauthier, Michael Faerber, Yutaka Nagashima, Shawn Wang
- ATP and ITP people:
  - Stephan Schulz, Geoff Sutcliffe, Andrej Voronkov, Kostya Korovin, Larry Paulson, Jasmin Blanchette, John Harrison, Tom Hales, Tobias Nipkow, Andrzej Trybulec, Piotr Rudnicki, Adam Pease, ...
- Learning2Reason people at Radboud University Nijmegen:
  - Herman Geuvers, Tom Heskes, Daniel Kuehlwein, Evgeni Tsivtsivadze, ...
- Google Research: Christian Szegedy, Geoffrey Irving, Alex Alemi, Francois Chollet, Sarah Loos
- ... and many more ...
- Funding: Marie-Curie, NWO, ERC

Some References

- C. Kaliszyk, J. Urban, H. Michalewski, M. Olsak: Reinforcement Learning of Theorem Proving. CoRR abs/1805.07563 (2018)
- Z. Goertzel, J. Jakubuv, S. Schulz, J. Urban: ProofWatch: Watchlist Guidance for Large Theories in E. CoRR abs/1802.04007 (2018)
- T. Gauthier, C. Kaliszyk, J. Urban, R. Kumar, M. Norrish: Learning to Prove with Tactics. CoRR abs/1804.00596 (2018).
- J. Jakubuv, J. Urban: ENIGMA: Efficient Learning-Based Inference Guiding Machine. CICM 2017: 292-302
- S. M. Loos, G. Irving, C. Szegedy, C. Kaliszyk: Deep Network Guided Proof Search. LPAR 2017: 85-105
- L. Czajka, C. Kaliszyk: Hammer for Coq: Automation for Dependent Type Theory. J. Autom. Reasoning 61(1-4): 423-453 (2018)
- J. C. Blanchette, C. Kaliszyk, L. C. Paulson, J. Urban: Hammering towards QED. J. Formalized Reasoning 9(1): 101-148 (2016)
- G. Irving, C. Szegedy, A. Alemi, N. Eén, F. Chollet, J. Urban: DeepMath - Deep Sequence Models for Premise Selection. NIPS 2016: 2235-2243
- C. Kaliszyk, J. Urban, J. Vyskocil: Efficient Semantic Features for Automated Reasoning over Large Theories. IJCAI 2015: 3084-3090
- J. Urban, G. Sutcliffe, P. Pudlák, J. Vyskocil: MaLARea SG1- Machine Learner for Automated Reasoning with Semantic Guidance. IJCAR 2008: 441-456
- C. Kaliszyk, J. Urban, J. Vyskocil: Automating Formalization by Statistical and Semantic Parsing of Mathematics. ITP 2017: 12-27
- Q. Wang, C. Kaliszyk, J. Urban: First Experiments with Neural Translation of Informal to Formal Mathematics. CoRR abs/1805.06502 (2018)
- J. Urban, J. Vyskocil: Theorem Proving in Large Formal Mathematics as an Emerging AI Field. LNCS 7788, 240-257, 2013.

Thanks and Advertisement

- Thanks for your attention!
- AITP – Artificial Intelligence and Theorem Proving
- March 22–27, 2020, Aussois, France, `aitp-conference.org`
- ATP/ITP/Math vs AI/Machine-Learning people, Computational linguists
- Discussion-oriented and experimental - submit a talk abstract!
- Grown to 80 people in 2019