universität innsbruck

Computational Logic

Reinforcement Learning for leanCoP

Cezary Kaliszyk Josef Urban
Henryk Michalewski Mirek Olšák

AITP 2018

March 28, 2018

Automated Theorem Proving

Historical dispute: Gentzen and Hilbert

- Today two communities: Resolution (-style) and Tableaux

Possible answer: What is better in practice?

- Say the CASC competition or ITP libraries?
- Since the late 90s: resolution (superposition)

But still so far from humans?

- We can do learning much better for Tableaux
- And with ML beating brute force search in games, maybe?

leanCoP: Lean Connection Prover

[*Otten 2010*]

Connected tableaux calculus

- Goal oriented, good for large theories

Regularly beats Metis and Prover9 in CASC (CADE ATP competition)

- despite their much larger implementation

Compact Prolog implementation, easy to modify

- Variants for other foundations: iLeanCoP, mLeanCoP
- First experiments with machine learning: MaLeCoP

Easy to imitate

- leanCoP tactic in HOL Light

Lean connection Tableaux

Very simple rules:

- Extension unifies the current literal with a copy of a clause
- Reduction unifies the current literal with a literal on the path

$$\frac{}{\{\}, M, Path} \quad Axiom$$

$$\frac{C, M, Path \cup \{L_2\}}{C \cup \{L_1\}, M, Path \cup \{L_2\}} \quad Reduction$$

$$\frac{C_2 \setminus \{L_2\}, M, Path \cup \{L_1\} \qquad C, M, Path}{C \cup \{L_1\}, M, Path} \quad Extension$$

First experiment: MaLeCoP in Prolog

[*Tableaux 2011*]

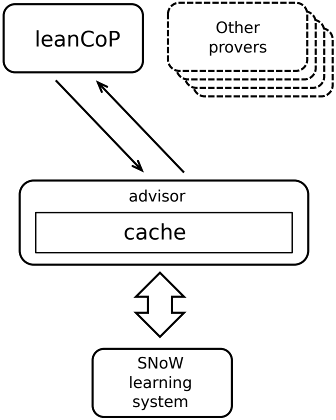
Select extension steps

- Using external advice

Slow implementation

- 1000 less inf per second

Can avoid 90% inferences!



What about efficiency: FEMaLeCoP

[*LPAR 2015*]

Very simple but very fast classifier

- Naive Bayes (with optimizations)

Approximate features and multi-level indexing

- Offline indexing
- Indexing for the current problem
- Discrimination tree stores NB data

Consistent clausification and skolemization

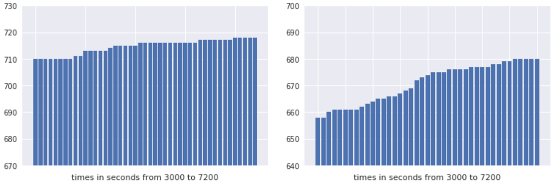
Performance is about 40% of non-learning leanCoP speed

- A few more theorems proved (3–15%)

What about stronger learning?

Yes, but...

- If put directly, huge times needed
- Still improvement small

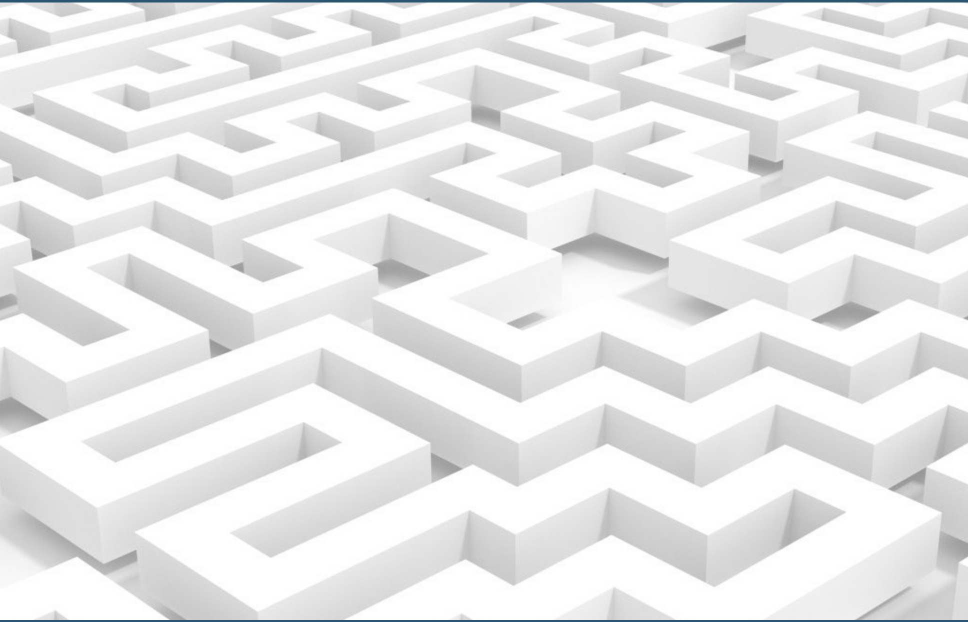


NBayes vs XGBoost on 2h timeout

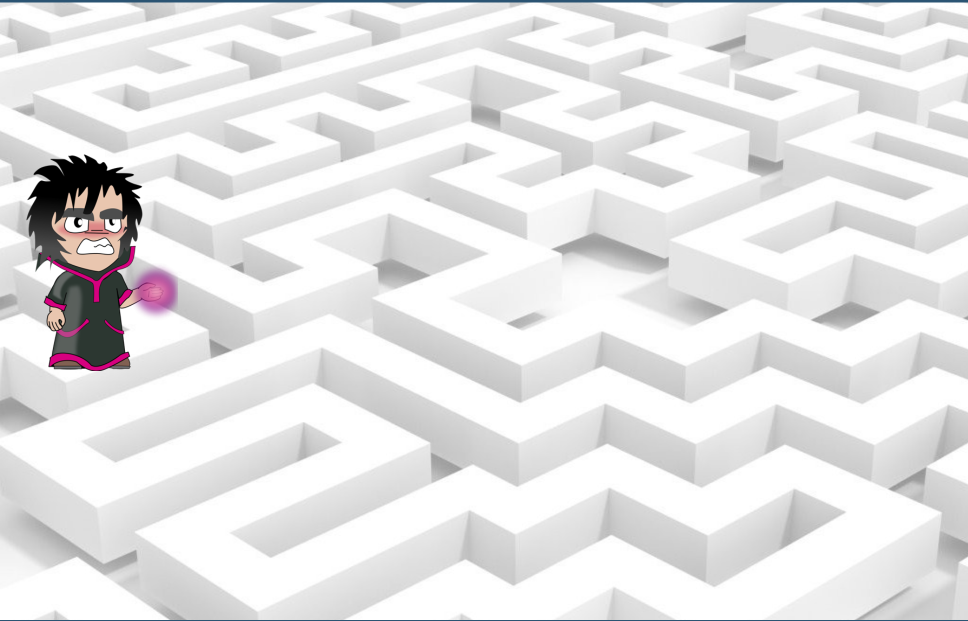
Preliminary experiments with deep learning

- So far quite slow

Is theorem proving just a maze search?



Is theorem proving just a maze search?



Is theorem proving just a maze search?

Yes and NO!

- The proof search tree is not the same as the tableau tree!
- Unification can cause other branches to disappear.

Provide an external interface to proof search

- Usable in OCaml, C++, and Python
- Two functions suffice

$$\text{start} : \text{problem} \rightarrow \text{state}$$

$$\text{action} : \text{action} \rightarrow \text{state}$$

- where

$$\text{state} = \langle \text{action list} \times \text{goal} \times \text{path} \times \text{remaining} \rangle$$

Is it ok to change the tree?

Most learning for games sticks to game dynamics

- Only tell it how to do the moves

Why is it ok to skip other branches

- Theoretically ATP calculi are complete
- Practically most ATP strategies incomplete

In usual 30s – 300s runs

- Depth of proofs with backtracking: 5–7 (complete)
- Depth with restricted backtracking: 7–10 (more proofs found!)

But with random playouts: depth hundreds of thousands!

- Just unlikely to find a proof → learning

Monte Carlo First Try: MonteCoP

Use Monte Carlo playouts to guide restricted backtracking

- Improves on leanCoP, but not by a big margin
- Potential still limited by depth

Can we do better?

- Arbitrarily long playouts
- Learn from the playouts

How to search a tree?

- Given some prior probabilities
- Given success (*fail*) rates so far

UCT: Select node n maximizing

$$\frac{w_i}{n_i} + c \cdot p_i \cdot \sqrt{\frac{\ln N}{n_i}}$$

Intuition

- Initially proportional to the prior
- Later win ratio dominates
- We will learn the win ratio

Monte Carlo Tree Search + Upper Confidence Bounds for Trees [*Szepesvari 2006*]

How to search a tree?

- Given some prior probabilities
- Given success (*fail*) rates so far

UCT: Select node n maximizing

$$\frac{w_i}{n_i} + c \cdot p_i \cdot \sqrt{\frac{\ln N}{n_i}}$$

Intuition

- Initially proportional to the prior
- Later win ratio dominates
- We will learn the win ratio

MCTS tree for `t9_zfmisc_1`

	prior, p_i	$\frac{w_i}{n_1}$	visits, n_i	
O	1.00	0.799	10000	
├── O	0.17	0.606	5625	
│ ├── O	0.64	0.719	4713	...
│ └── O	0.36	0.023	912	...
├── O	0.08	0.013	622	
│ ├── X				
│ ├── O	0.20	0.014	76	...
│ ├── O	0.32	0.024	113	...
│ └── X				
└── O	0.08	0.011	68	
└── O	0.10	0.007	5	...

Learn Policy: Which actions to take?

Even for a single problem

- If we know that some branches failed
- We can avoid such branches in other parts of the “maze”

Playouts following UCT

- After a number of playouts, select the most visited branch
- Only continue inside that branch (called **big step**)

A sequence of big steps ends in a proof, dead end, or is too long

- We can either way learn which actions were chosen
- With some initial win heuristic (remaining goals, size, constant)

Learn Value: How likely is a proof state to be provable?

Learn from all bigstep states

- One if theorem, zero otherwise

Learn Value: How likely is a proof state to be provable?

Learn from all bigstep states

- One if theorem, zero otherwise

With 150K good value training samples and 250K good policy training samples

- XGBoost policy train time: 4 min, Value train time: 8 min
- 2000 problems run with 100K inferences, no bigsteps

	time (min)	Theorems
No learning	1.5	440
Only learn values	5.0	535
Only learn policy	10.5	790
Learn both	11.5	871

Reinforcement from scratch

Starting with no data, and with 1500 playouts per bigstep

round	thms
MC	665
1	654
2	718
3	727
4	754
5	748
6	769
7	760
8	776
9	776

....

....

10	782
11	797
12	796
13	800
14	795
15	794
16	792
17	804
.....	....
29	815
30	820

Conclusion

- Reinforcement learning on small Mizar data project
  - UCT, action, value work in connection based setup
  - Learning from scratch can work even for a single problem
- Lots of things to try
  - Other cost functions
  - Other learning frameworks
  - Larger experiments