Project Proposal: Forward Reasoning In Hindsight

Michael Rawson[1], Zsolt Zombori[2,3],
Maximilian Doré[4], and Christoph Wernhard[5]

[1] TU Wien, Austria michael@rawsons.uk
[2] Alfréd Rényi Institute of Mathematics, Hungary zombori@renyi.hu
[3] Eötvös Loránd University, Budapest, Hungary
[4] University of Oxford, United Kingdom maximilian.dore@cs.ox.ac.uk
[5] University of Potsdam, Germany info@christophwernhard.com

Prologue: Data in ML4ATP

- Learn from “past experience” trying to prove things
- Traditional method requires *existing proofs*
- Assumption: this helps with other proofs!
- Drawback: sparse, precious data

“Generating data by [other means] is slowly gaining traction”

– Zsolt Zombori, AITP ‘24, slightly misquoted

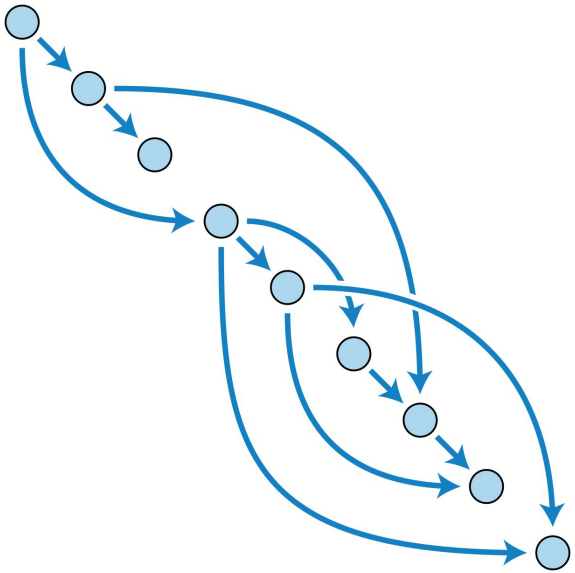
Hindsight Experience Replay

Imagine that you are learning how to play hockey and are trying to shoot a puck into a net. You hit the puck but it misses the net on the right side. The conclusion drawn by a standard RL algorithm in such a situation would be that the performed sequence of actions does not lead to a successful shot, and little (if anything) would be learned. It is however possible to draw another conclusion, namely that this sequence of actions ***would be successful if the net had been placed further to the right****.*

– Andrychowicz et al, “Hindsight Experience Replay”

Hindsight in ATP

- Start with *premises*
- Try to derive *conjecture*
- Likely fail
- Take some D you derived
- ***Pretend D is what you are trying to prove***
- Use this proof to learn some heuristic
  - For example: clause C used to prove G, but C' was not
  - Train clause-selection heuristic from (C, G, yes/no) triples
- Already done by Aygün et al (2022)
  - also presented at AITP!
- Good.



Inspiration: A Slight Hitch

- *Refutational* calculi are very popular
  - Resolution, superposition, instance-based methods, connection tableaux (ish), …
- Start with axioms and a negated conjecture, try to prove falsum
- Goal here arguably “false”
- But this is not very informative as a goal
- Aygün et al use input set S as a goal
  - If clause C derived, target is S + ¬C
- Works!
- **But wouldn’t it be better with a non-refutational, “forward” calculus?**

Proposal

- Use “forward” calculi that do not use the (negated) conjecture
  - E.g. condensed detachment — Wernhard
- Deduce consequences from axioms
- When goal is reached (subsumed), done
- Very explicit, concise, informative goals
- But: can’t use popular calculi

Forward Calculi

Condensed Detachment

- “Modus ponens with resolution”
- Surprisingly expressive
- Some really hard problems
  - Challenge: LCL073-1 (46 nodes in the proof DAG, near-impossible for ATPs)
- Simple proof structure (binary DAGs)
  - Wernhard et al have productive line of research using these

$$\mathrm{CD}\ \frac{P \qquad P' \to Q}{Q\theta}\ \theta = mgu(P, P')$$

Constructive Type Theory?

- Usually interactive theorem proving:
  - Types are propositions, terms are proofs
- Not obviously an ATP calculus, nor “forwards”
- But: ATP caveman apply terms to each other, deduce type

- SKI combinators + induction schemes reasonably expressive!
- Systems like Agda (mostly) happy with this ‘interaction’
- Thanks to Max for putting up with this horror!
- Totally sensible, readable proofs…

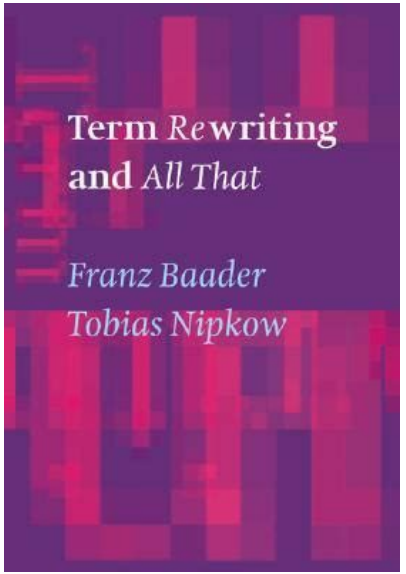
```
zero-lunit : (n : Nat) → zero + n ≡ n
zero-lunit = induct refl (K (cong suc))
```

```
comm : (x y : Nat) → y + x ≡ x + y
comm = S
  (S (K induct) (induct refl (K (cong suc))))
  (S
    (K (S (K (S (K (S (K (S S (K (cong suc)))) K')))) (S (K trans))))
    (S (K (S suc-left)) K')
  )
```

```
λ x →
induct (induct refl (λ z → cong suc) x)
(λ x₁ x₂ → trans (induct refl (λ z → cong suc) x) (cong suc x₂))
```

Unit Equational Reasoning?

- First-order logic, but you're only allowed universally-quantified equations.
- "Overlap" or "superpose" axioms
- Try to rewrite the goal away
- "Almost" forwards already
- Question: can this be rephrased in a totally forward way?
- Need (at least) some advanced goal check:
  - Have a = b and c = d
  - But the goal is f(a, c) = f(b, d)
- Would especially like to hear from people about this!



Initial Experiments

A Weird Experimental Setup

- Hard-code LCL073-1
- Monitor progress by known 46-step proof
- Loop:
  - Do 100 activations of given-clause CD search
  - Choose given clauses ε-greedily using neural network
  - Log how far we got
  - Sample 100 (intermediate, proxy goal, yes/no) tuples
  - Train network on experience tuples for 1 epoch
- Sales pitch: leave this thing chewing on a hard problem overnight
  - Even if it fails, maybe have interesting goal-conditioned theory exploration?

Results & Dynamics

- Random baseline: 2–4 steps of the proof
- **Rapid** improvement to 8–11 steps in ~30 iterations
  - :-O
- Tends to plateau after that
  - :-/
- Fluke: once went to 21 steps
  - :-O :-O :-O
- Have never managed this again
  - :-(
- Model tends to evaluate next step as very unlikely
  - :-(

Questions for You

- Good idea? Bad idea? Already done?
- Better ways of doing constructive type theory?
- Suggestions about UEQ?
- Other forward calculi you know of?
- Miscellaneous experimental suggestions?

Future Directions

- Receive your feedback!
- Improve stability/performance by Hard Work
- Make Agda work
- Make UEQ work
- More scale?

Feedback