Towards an Efficient Architecture for Intelligent Theorem Provers

Michael Rawson, Giles Reger

University of Manchester, UK

Background

"The problem with all this deep neural stuff is that it's *slow.*"

AITP '19 participant, paraphrased

Efficient ATP Context

- Fully automatic provers: “fire and forget”
- Supporting full first-order logic (with equality)
- Historically, little learning from experience
- Instead use efficient calculi and highly-tuned algorithms

Automatic theorem proving: an abstract view

1. Are we done yet?
2. No? Ugh, fine.
3. **Pick a Thingy.**
4. Do All the Things™ with your Thingy.
5. Go to (1)

What do we want?

- Learn from past experience proving things
- Guide future prover runs based on the knowledge gained
- Ideally without affecting “raw” performance too much

Guidance is Hard

- Optimal picking is not decidable in general
- Can work for human problems: human mathematicians exist
- *Thingies* (formulae, clauses…) generally hostile for learning:
  - “Lossy” representations: definitionally not as good as they could be
  - “Lossless” representations: **better (?), just really difficult**.

Guidance is Inefficient (?)

- Direct guidance means adding a heuristic “black box”
- Use it to pick your Thingies better
- Therefore, at least one heuristic call per loop
- If your heuristic does a lot of computation (neurally?), this is slow
- Claim: *neural networks are not low-throughput, merely high-latency*

A Solution

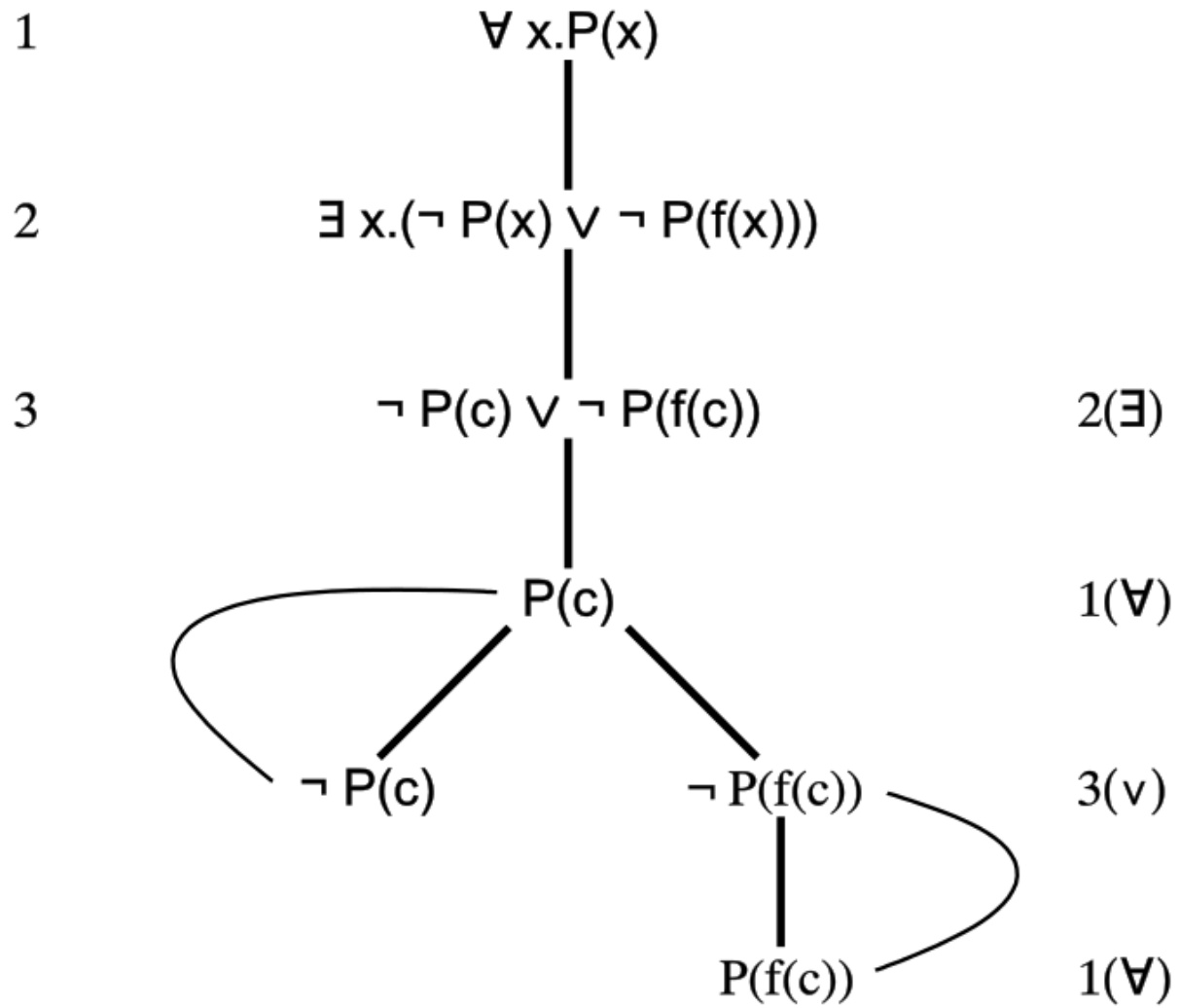
Well, maybe.

Desiderata for neural provers

- Proof state must be *reasonably small*
- Proof state must be *human-readable*
- Proof state must be *independent* and *self-contained*
- Proof state must be capable of *evaluation in (data)-parallel*

A suitable calculus

- Refutation tableaux (proof state is small, parallel)
- Non-clausal tableaux (proof state is small, human-readable)
- Tableaux without unification (proof state is independent, parallel)
- This is *horrible* for proof search…



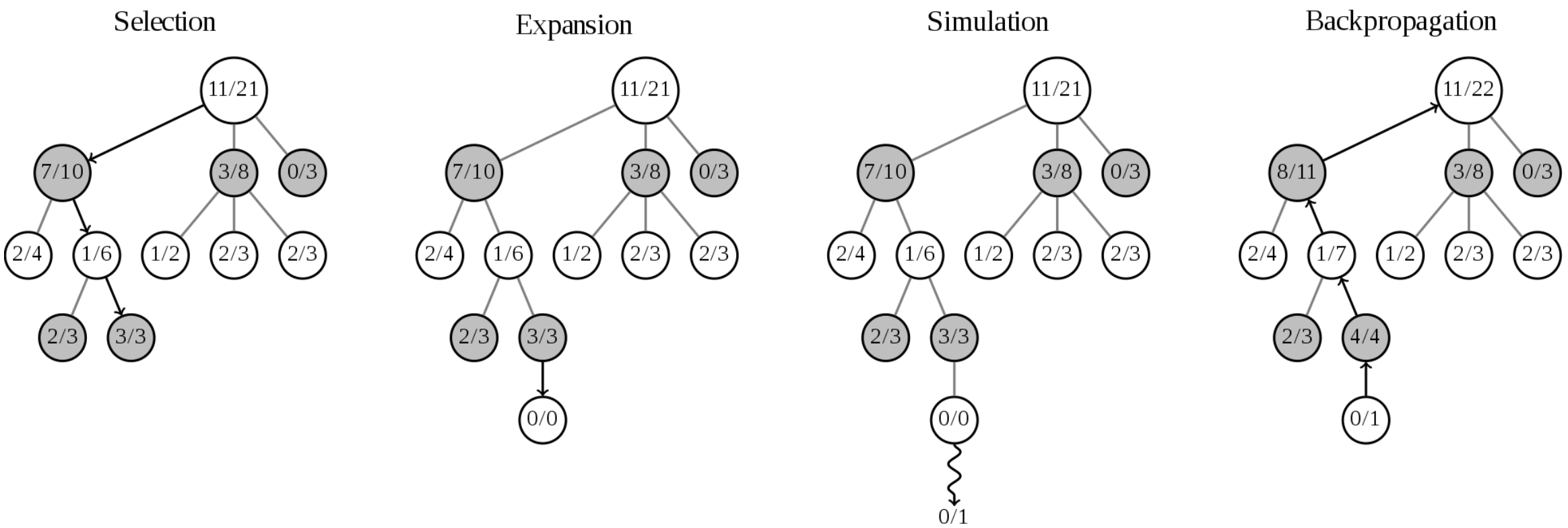
https://en.wikipedia.org/wiki/Method_of_analytic_tableaux#/media/File:First-order_tableau.svg

Problem: explosive proof search

- Necessarily explosive calculus
- Solution: can be controlled if the heuristic is good enough

Problem: controlling exploitation

- Heuristic guides proof search, but it gets it wrong occasionally
- Proof search might become “stuck” and therefore incomplete
- Must balance *exploitation* versus *exploration*
- Solution: Monte-Carlo Tree Search, as used in *MonteCoP/rlCoP*



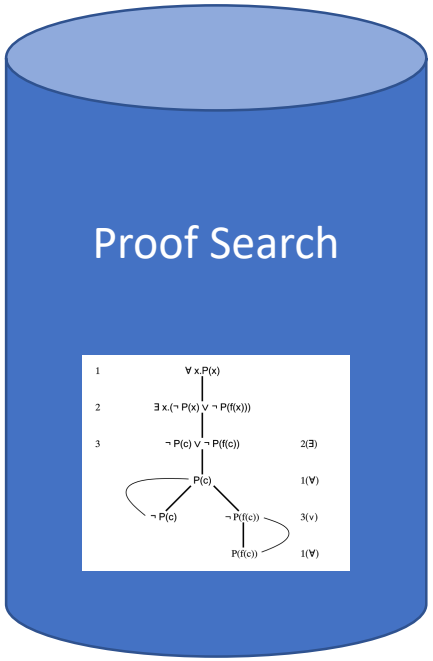
https://en.wikipedia.org/wiki/Monte_Carlo_tree_search#/media/File:MCTS_(English)_-_Updated_2017-11-19.svg

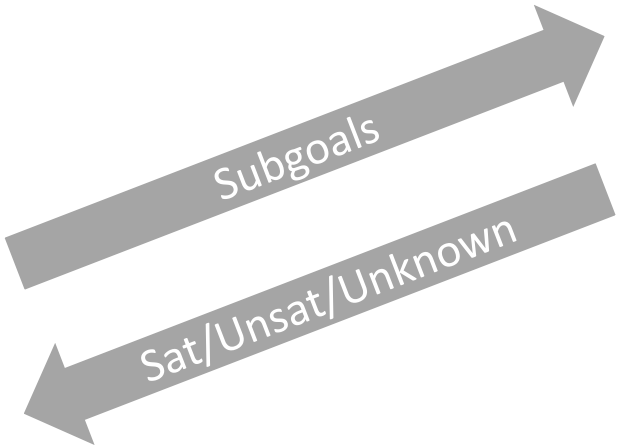
Problem: deep proofs

- Proofs can be significantly deep with this method
- Solution: apply an existing fast *oracle* ATP (Z3 with MBQI) to subgoals
- Sound because each sub-goal is independent of any other
- Could also be any first-order ATP or counter-example finder
- Oracle says:
  - “satisfiable”: you messed up, prune this branch
  - “unsatisfiable”: great, this subgoal is solved
  - “unknown”: keep going…

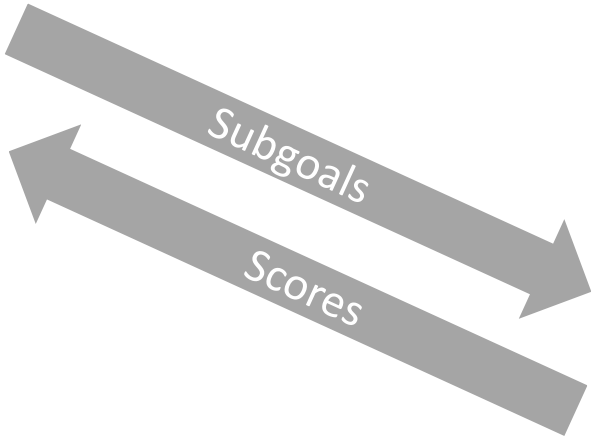
A Prover Design

- Tableaux search via MCTS
- Fresh nodes placed on a queue, heuristic evaluates in *batches*
- Heuristic estimates “truthiness” of current subgoal
- Update nodes with scores when they arrive from the heuristic
- Explore other areas in the meantime
- Whack subgoals with a Z3 hammer occasionally, in parallel





(saturates CPU)



(saturates GPU)

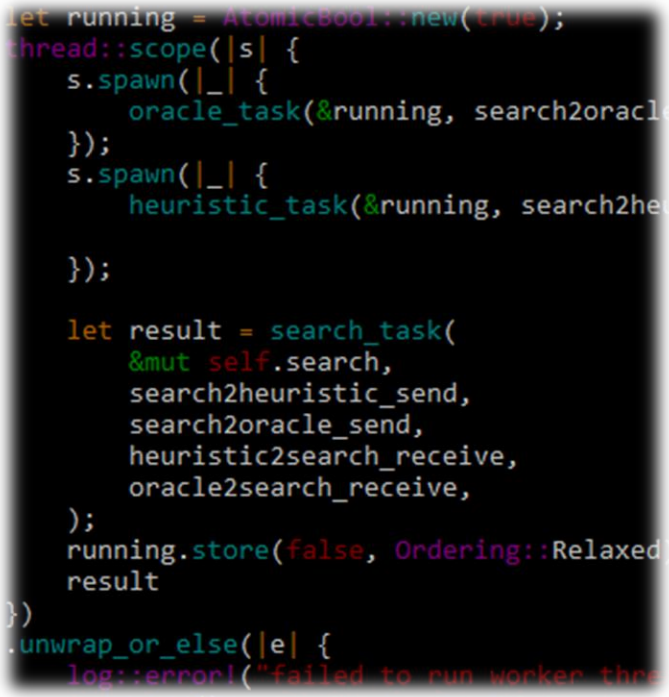
Some advantages

- Common subgoals can be *shared*
- Quite general: new inference rules, other logics?
- All available CPU/GPU cores utilised
- Possible fast incomplete mode: drop poor branches
- Oracle generates training examples during proof search
- Pluggable oracle – *is this a new domain for traditional ATPs*?
- Pluggable heuristic – *I might make this a competition*!

Findings

Engineering

- Relatively simple to implement: one (definitely non-expert) author
- However, parallel DAG traversal/update very difficult to get right!
- ≈ 2,000 lines of Rust code
- Batching neural heuristic much more efficient
- Z3 quite expensive, but definitely worthwhile



Mizar benchmark

- MPTP dataset, minimised (“m40” - thanks to Josef Urban)
- A mathematical benchmark: unclear how other domains fare
- Results promising, but Z3 is a strong prover already.
- Apologies for no numbers…

Learning from experience

- Simple database lookup of previously-proved sat/unsat subgoals proves ≈5% more, with significant speedup
- Neural heuristic learns to 55% accuracy – surely this can be improved!
- Can bootstrap from a problem set, *even if no problems are solved initially*

Conclusions

Results

- Neural ATPs are not *necessarily* slow, just different
- Need new calculi/provers
- Parallel theorem provers are a necessary evil for the future
- Significant advantages (and disadvantages!) to doing it the stupid way

Future work

- Make sure the thing is sound!
- Evaluation on MPTP
- More training data, better heuristics
- “FOL truthiness” ML competition?
- Engineering for efficiency

Questions