Towards an Efficient Architecture for Intelligent Theorem Provers

Michael Rawson and Giles Reger

University of Manchester, Manchester, UK

High-performance automated theorem provers for first-order logic (e.g. CVC4 [2], E [11], iProver [6], Vampire [7]) include hand-coded heuristics to guide proof search, often exposed as individual prover options. These heuristics perform well, but have a number of disadvantages including a lack of generality over problems (necessitating *portfolio* modes [9, 10]), inability to learn from experience, and maintenance overhead. There is therefore interest in employing machine-learning techniques to guide proof search in automatic theorem provers, with approaches such as FEMaLeCoP [5], ENIGMA [4], or Deep Network Guided Proof Search [8] (DNGPS).

These systems experience a trade-off between the expressivity of their learning algorithms versus the impact of guidance on "raw" prover performance. At extremes:

- The heuristic is fast, but does not take into account the entire proof state (e.g. the MaLeCoP family), restricting the prover to learning from features.
- The heuristic takes into account the entire proof state (usually via neural networks), but is too slow to use all the time. The DNGPS system runs with the heuristic for a fixed amount of time, then reverts back to the old heuristics thereafter.

Ideally, an intelligent system would guide search based on the structure of the current proof state, while also remaining performant enough to run continuously without significantly affecting prover performance. We present a prover architecture which attempts to achieve this ideal, and show that it has several other desirable properties.

Desiderata In such a system we require the following:

1. *Proof state must be small.* Attempting to evaluate large proof states structurally requires a lot of resources. Saturation-based provers such as E or Vampire can have very large proof states, for example.
2. *Evaluation of states must be possible in parallel.* Machine-learning algorithms tend to operate more efficiently in batches. Tree-based approaches (tableau *etc.*) lend themselves to this, whereas saturation provers are inherently sequential.
3. *Subgoals must be independent.* If the prover has a notion of (sub-)goals which must be dispatched (such as in tableau or connection provers), these should be independent of the rest of the search space. Otherwise, the learning system is trying to learn while blind to the context of the search.

Calculus and Algorithm We implement a first-order tableau calculus without unification, with equality, on non-clausal formulae. By using this very "natural" representation, the hope is that inherent proof structure will be more apparent to machine-learning algorithms, which do not have to invert the process of clausification. The tableau space is explored in parallel by means of a UCT-maximising tree search (similar to that employed by MonteCoP [3]), with new goals placed on a global queue for evaluation in batches by means of arbitrary machine-learning methods, currently a GCN [1, 12].

Advantages This prover architecture satisfies requirements 1–3, but also shows promise in other areas. In terms of reasoning, the calculus used is relatively flexible, allowing for extension to reasoning with theories, induction, and full higher-order logic without modifying the whole prover. In terms of efficiency, such a prover can also make full use of multi-core systems, allowing for linear exploration speedup with the number of available cores, eventually saturating the device or core used for running machine-learned algorithms. The prover is also well-suited to a hybrid approach in which promising subgoals are dispatched to an existing first-order ATP.

Evaluation and Future Work Evaluation and implementation of an example prover system based on this architecture is ongoing, but initial results are promising, with the system appearing to "learn to prove" harder problems based on prior experience with easier problems. Future work includes exploring calculus options, optimisation, further exploration of machine-learning methods, and using the prover as a "pre-processor" for an existing first-order ATP.

References

[1] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pages 3844–3852, 2016.

[2] Morgan Deters, Andrew Reynolds, Tim King, Clark W. Barrett, and Cesare Tinelli. A tour of CVC4: how it works, and how to use it. In *Formal Methods in Computer-Aided Design, FMCAD 2014, Lausanne, Switzerland, October 21-24, 2014*, page 7, 2014.

[3] Michael Färber, Cezary Kaliszyk, and Josef Urban. Monte carlo connection prover. *arXiv preprint arXiv:1611.05990*, 2016.

[4] Jan Jakubuv and Josef Urban. ENIGMA: efficient learning-based inference guiding machine. In *International Conference on Intelligent Computer Mathematics*, pages 292–302. Springer, 2017.

[5] Cezary Kaliszyk and Josef Urban. FEMaLeCoP: Fairly efficient machine learning connection prover. In *Logic for Programming, Artificial Intelligence, and Reasoning*, pages 88–96. Springer, 2015.

[6] K. Korovin. iProver – an instantiation-based theorem prover for first-order logic (system description). In *Proceedings of the 4th International Joint Conference on Automated Reasoning, (IJCAR 2008)*, volume 5195 of *Lecture Notes in Computer Science*, pages 292–298. Springer, 2008.

[7] Laura Kovács and Andrei Voronkov. First-order theorem proving and Vampire. In *International Conference on Computer Aided Verification*, pages 1–35. Springer, 2013.

[8] Sarah Loos, Geoffrey Irving, Christian Szegedy, and Cezary Kaliszyk. Deep network guided proof search. *arXiv preprint arXiv:1701.06972*, 2017.

[9] Michael Rawson and Giles Reger. Dynamic strategy priority: Empower the strong and abandon the weak. In *Proceedings of the 6th Workshop on Practical Aspects of Automated Reasoning (PAAR)*, pages 58–71.

[10] Giles Reger, Martin Suda, and Andrei Voronkov. The challenges of evaluating a new feature in Vampire. In *Vampire Workshop*, pages 70–74, 2014.

[11] Stephan Schulz. E — a brainiac theorem prover. *AI Communications*, 15(2, 3):111–126, 2002.

[12] Mingzhe Wang, Yihe Tang, Jian Wang, and Jia Deng. Premise selection for theorem proving by deep graph embedding. In *Advances in Neural Information Processing Systems*, pages 2786–2796, 2017.