Learning to Identify Useful Lemmas from Failure*

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

[1] TU Wien, Austria michael@rawsons.uk
[2] University of Potsdam, Germany info@christophwernhard.com
[3] Alfréd Rényi Institute of Mathematics, Hungary zombori@renyi.hu
[4] Eötvös Loránd University, Budapest, Hungary

Introduction We investigate learning to identify useful lemmas for ATP, where usefulness is defined in terms of 1) reducing proof search and 2) shortening the length of the overall proof. How can ATP performance be improved by the generation and selection of useful lemmas? In particular, we raise the question of what one can learn from a failed proof attempt. We present a proposal about how to learn from failed proof attempts of a single problem. This is in sharp contrast with previous works that rely on a large corpus of problems and aim to improve performance based on success obtained with easier problems. By way of motivation, we argue that human mathematicians learn from failed attempts as well.

Restriction to a class of problems with accessible and simple proof structures Interested in novel techniques, we work with a restricted class of first-order problems, *condensed detachment* (CD) problems [9, 7], due to Carew A. Meredith [5]. Inference steps can be characterized by detachment (modus ponens) combined with unification. Proof structures are particularly simple and accessible: full binary trees, or terms with a binary function symbol D, which we call D-terms. Constants in these terms label axioms. As examples of D-terms consider 1, a constant representing a use of the axiom labeled by 1; $\mathsf{D}(1,1)$, representing a detachment step applied to axiom 1 as major and minor premise; or $\mathsf{D}(1,\mathsf{D}(1,1))$, representing a proof with two detachment steps. These proof terms are closely related to proof structures of the connection method [3, 4].

Proof search and data extraction We rely on theorem prover SGCD [8] which performs proof search by structure enumeration of binary trees (interwoven with formula unification), until a suitable D-term is found. Enumeration can be axiom-driven, i.e. starting from axiom set As, producing D-terms that represent complete proofs of unit lemmas. We can also enumerate goal-driven, starting from conjecture C and obtaining partial proof trees of C. In practice we interleave goal-driven and axiom-driven phases. Using the idea of Hindsight Experience Replay [1], we can "pretend" that both sorts of failed proof attempts are in fact successful: In the axiom-driven case, we change the goal conjecture to the one that was actually proven and in the goal-driven case, we change the axioms to include whatever is needed to complete the proof. Hence, we end up enumerating complete proof trees of "some" problems. We note that the idea of Hindsight Experience Replay has already been applied to theorem proving in [2] in the context of training a policy model to guide saturation-style forward reasoning.

Given a proof tree P' of some formula C' from axiom set As', any connected subgraph S' of P' can be considered as the proof of a lemma candidate L. If S' is a full tree, it proves a unit lemma, which is the formula associated with its root. Otherwise, it proves a Horn clause, whose head is the root formula of S' and whose body consists of the open leaves of S'. If we

*Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project-ID 457292495, by the North-German Supercomputing Alliance (HLRN), by the ERC grant CoG ARTIST 101002685, by the Hungarian National Excellence Grant 2018-1.2.1-NKP-00008, the Hungarian Artificial Intelligence National Laboratory Program (RRF-2.3.1-21-2022-00004) and the ELTE TKP 2021-NKTA-62 funding scheme.

can measure how useful lemma L is for proving C' given axioms As', this can serve as a useful training signal for a guidance model. For the utility measure U, there are easy-to-compute logical candidates, such as the compression in tree size, tree height, DAG size etc. A more refined measure is obtained if we reprove C' with the lemma L added to the axioms As' and observe how the number of interence steps change. This is slower to compute, but takes into account the particularities of the prover, hence providing more focused guidance. In practice, we find that the best performance is obtained by reproving and then normalising the inference step reduction into $[-1, 1]$, where -1 means that the problem could not be solved within the original inference limit and 1 is assigned to the lemma that yields the greatest speedup. We end up with tuples $\langle C', As', L, U\rangle$ to learn from.

Iterating proof search and training During the proof search of conjecture C from axiom set As, we keep track of all produced proof trees P' and collect $\langle C', As', L, U\rangle$ tuples, forming a training dataset D. Once proof search is unsuccessful, we fit a neural lemma selector to D. This neural model $M(\text{conjecture}, \text{axioms}, \text{lemma})$ predicts the utility of the input lemma for proving the conjecture from the axioms. Model M is next evaluated on all collected lemmas, along with the original conjecture and axioms, i.e. we compute pairs

$$\{\langle L, U\rangle \mid \langle \_, \_, L, \_\rangle \in D,\ U = M(C, As, L)\}$$

Lemmas with the top k utilities are selected, where k is a hyperparameter to be tuned. The selected lemmas are added to the problem as axioms[1] and we can start proof search again. Each iteration produces novel proof attempts and novel training signal, hopefully guiding the prover closer to solving the target problem.

Current Status We have performed experiments on training a unit lemma selector from successful proof attempts, which is explained in [6]. We are currently working on extending the codebase to accommodate Horn lemmas and the extraction of training signal from failed attempts.

References

[1] Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Pieter Abbeel, O., Zaremba, W.: Hindsight experience replay. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc. (2017), https://proceedings.neurips.cc/paper_files/paper/2017/file/453fadbd8a1a3af50a9df4df899537b5-Paper.pdf

[2] Aygün, E., Anand, A., Orseau, L., Glorot, X., Mcaleer, S.M., Firoiu, V., Zhang, L.M., Precup, D., Mourad, S.: Proving theorems using incremental learning and hindsight experience replay. In: Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., Sabato, S. (eds.) Proceedings of the 39th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 162, pp. 1198–1210. PMLR (17–23 Jul 2022), https://proceedings.mlr.press/v162/aygun22a.html

[3] Bibel, W.: Automated Theorem Proving. Vieweg, Braunschweig (1982). https://doi.org/10.1007/978-3-322-90102-6, second edition 1987

[4] Bibel, W., Otten, J.: From Schütte's formal systems to modern automated deduction. In: Kahle, R., Rathjen, M. (eds.) The Legacy of Kurt Schütte, chap. 13, pp. 215–249. Springer (2020). https://doi.org/10.1007/978-3-030-49424-7_13

[1] If the prover has some special mechanism for handling lemmas, they need not be treated as axioms.

[5] Prior, A.N.: Logicians at play; or Syll, Simp and Hilbert. Australasian Journal of Philosophy **34**(3), 182–192 (1956). https://doi.org/10.1080/00048405685200181

[6] Rawson, M., Wernhard, C., Zombori, Z., Bibel, W.: Lemmas: Generation, selection, application. CoRR **abs/2303.05854** (2023). https://doi.org/10.48550/ARXIV.2303.05854

[7] Ulrich, D.: A legacy recalled and a tradition continued. J. Autom. Reasoning **27**(2), 97–122 (2001). https://doi.org/10.1023/A:1010683508225

[8] Wernhard, C.: CD Tools — Condensed detachment and structure generating theorem proving (system description). CoRR **abs/2207.08453** (2023). https://doi.org/10.48550/ARXIV.2207.08453

[9] Wernhard, C., Bibel, W.: Investigations into proof structures. CoRR **abs/2304.12827** (2023). https://doi.org/10.48550/ARXIV.2304.12827