SAT-based Learning of Temporal Logics

Internship Proposal

Adrien Pommellet `adrien [at] lrde.epita.fr`
Daniel Stan `ds [at] tudo.re`

February 3, 2025

Topics: passive learning; linear-time temporal logic; computation tree logic; SAT solving; C++ programming

Context. *Passive learning* is the act of computing a theoretical model of a system from a given set of data, without being able to acquire further information by actively querying said system. The input data may have been gathered through monitoring, collecting executions and outputs of systems. Automata and logic formulas tend to be the most common models, as they allow one to better express the behaviour and properties of systems of complex or even entirely opaque design.

Linear-time Temporal Logic LTL and *Computation Tree Logic* CTL remain two of the most widely used formalisms for specifying temporal properties of systems. The former applies to finite or infinite execution traces, the latter, to execution trees. They fit the passive learning framework very well, being a concise way to distinguish between correct and incorrect executions. The LTL and CTL learning problems are complex: even simple fragments on finite traces are NP-complete [4, 1], and consequently recent algorithms tend to leverage SAT solvers [5, 6, 2].

Objectives. The student is expected to contribute to the following ongoing research topics:

1. Tackle the LTL passive learning problem on samples of Kripke structures, as only isolated traces have been considered so far [5].
2. Design a learning algorithm for CTL*, a logic that subsumes both LTL and CTL.
3. Experimentally compare the expressivity, conciseness, and ease of learning of LTL, CTL, and CTL*.
4. Explore the learning problem for the hyperlogic HyperLTL [3] that enriches LTL with path variables and quantifiers.

Qualifications. This internship requires both theoretical and practical skills. On the one hand, elementary knowledge of model-checking theory is helpful but not mandatory. On the other hand, the intern is expected to contribute to the existing C++ tool `LearnCTL`[1] and employ state-of-the-art SAT solvers. For that reason, some C++ experience is strongly recommended.

Practical information. The internship is supervised by Adrien Pommellet and Daniel Stan, both assistant professors at EPITA and members of LRE. It will preferably be located at the EPITA campus in Le Kremlin-Bicêtre (14-16 Rue Voltaire, 94270 Le Kremlin-Bicêtre). If you are interested, do not hesitate to get in touch with us for further details.

References

[1] Benjamin Bordais, Daniel Neider, and Rajarshi Roy. Learning temporal properties is np-hard, 2023.

[2] Benjamin Bordais, Daniel Neider, and Rajarshi Roy. Learning branching-time properties in ctl and atl via constraint solving. In André Platzer, Kristin Yvonne Rozier, Matteo Pradella, and Matteo Rossi, editors, *Formal Methods*, pages 304–323, Cham, 2025. Springer Nature Switzerland.

[3] Michael R. Clarkson, Bernd Finkbeiner, Masoud Koleini, Kristopher K. Micinski, Markus N. Rabe, and César Sánchez. Temporal logics for hyperproperties. In Martín Abadi and Steve Kremer, editors, *Principles of Security and Trust*, pages 265–284, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[4] Nathanaël Fijalkow and Guillaume Lagarde. The complexity of learning linear temporal formulas from examples. In Jane Chandlee, Rémi Eyraud, Jeff Heinz, Adam Jardine, and Menno van Zaanen, editors, *Proceedings of the 15th International Conference on Grammatical Inference, 23-27 August 2021, Virtual Event*, volume 153 of *Proceedings of Machine Learning Research*, pages 237–250. PMLR, 2021.

[5] Daniel Neider and Ivan Gavran. Learning linear temporal properties. In Nikolaj S. Bjørner and Arie Gurfinkel, editors, *2018 Formal Methods in Computer Aided Design, FMCAD 2018, Austin, TX, USA, October 30 - November 2, 2018*, pages 1–10. IEEE, 2018.

[6] Adrien Pommellet, Daniel Stan, and Simon Scatton. Sat-based learning of computation tree logic. In *Automated Reasoning*, pages 366–385, Cham, 2024. Springer Nature Switzerland.

[1] publicly available at https://gitlab.lre.epita.fr/adrien/learnctl