COM S 572 : Principles of Artificial Intelligence
MLTL Inference – Proposal

Luke Marzen
ljmarzen@iastate.edu

Jayaraman, Swaminathan
swamjay@iastate.edu

Nhan Tran
nhtran@iastate.edu

Zili Wang
ziliw1@iastate.edu

March 8, 2024

Mission-time Linear Temporal Logic (MLTL) is a discrete time, finite interval bounded temporal logic that has found numerous recent applications. For example, MLTL was the specification logic for NASA's Robonaut2 verification project [6], as well as for the design-time and runtime verification of the NASA Lunar Gateway Vehicle System Manager [3]. Other applications of MLTL include autonomous satellite [9], UAV Traffic Management [5], and more.

Given a (finite) set of atomic propositions $\mathcal{AP}$, the syntax of MLTL formulas φ and ψ are defined recursively:

$$\varphi, \psi := true \mid false \mid p \mid \neg\varphi \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid F_{[a,b]}\varphi \mid G_{[a,b]}\varphi \mid \varphi U_{[a,b]}\psi \mid \varphi R_{[a,b]}\psi,$$

where $p \in \mathcal{AP}$, and $a, b \in \mathbb{Z}$ such that $0 \leq a \leq b$. The symbols F, G, U, R denote the temporal operators Future, Globally, Until, and Release, respectively. A trace π represents a sequence of truth assignments to the atomic propositions in $\mathcal{AP}$ over time, and π satisfies a MLTL formula φ is denoted as $\pi \models \varphi$. If π does not satisfy φ, then $\pi \not\models \varphi$. The explicit semantics of each MLTL operator can be found in [4].

The object of this proposed project is to explore various approaches to MLTL inference, formally described as follows: Given a set traces $T = T^+ \cup T^-$ over n variables, learn a MLTL formula φ such that for all $\pi \in T^+$, $\pi \models \varphi$, and for all $\pi \in T^-$, $\pi \not\models \varphi$. Intuitively, T^+ represents the set of positive examples, or desirable behaviors, and T^- represents the set of negative examples, or undesirable behaviors. These traces can come from simply observing the behavior of some system of interest, or they can be generated by some other means, such as a model checker or a simulator. The goal is to learn a MLTL formula that specifies the desired behavior of the system, and to do so in a way that generalizes to new, unseen traces.

There is a large corpus of work on learning regular languages, stemming from Angluin's L^* algorithm [1] that learns a minimal deterministic finite automata (DFA) from positive and negative examples. Work in learning temporal logic formulas have also recently been explored, with approaches ranging from symbolic learning algorithms [10, 2] to deep learning algorithms [7, 8]. There is no published work on learning MLTL formulas, and the goal of this project is to explore various approaches to this problem, and to compare and contrast their performance.

Previously, Zili Wang has developed and evaluated Genetic Programming (GP) based approach to learning MLTL formulas, and lays the groundwork for the proposed project. The repository containing the code and datasets for this project can be found at `https://github.com/zwang271/mltl-inference`. Work that will be reused includes the following components:

1. A parser (Python) for MLTL formulas, that can compute various properties of the formula, such as the number of atomic propositions, the syntax tree, worst propagation delay (wpd) (defined in [6]), and various other useful functions.

2. An MLTL interpreter (C++): on input trace π and a MLTL formula φ, determines if $\pi \models \varphi$.

3. A dataset generator (Python) that given an MLTL formula, uses either random trace sampling or regular expression sampling (see [4]) to generate a set of positive and negative examples.

4. 9 datasets, each with 500 positive and 500 negative examples, split into 80% training and 20% testing sets.

As a group, we aim evaluate various approaches using a combination of metrics such as run time, formula accuracy as a percent of correctly classified traces, and formula length/-complexity. Each team member will be responsible for implementing a different technique, though if some techniques are found to be non-viable or require more effort than reasonable to complete in a semester we will remain flexible and collaborate to ensure the project is completed. A description of each approach and the team member responsible for it is as follows:

- Zili - Graph Neural Network with weight extraction

  Prior work applies Graph Neural Networks (GNN) to learn Linear Temporal Logic over finite trace (LTLf) formulas by extracting formula from the weights of trained GNNs

[8]. The approach will be adapted to MLTL, and we seek generalize the methodology in interesting ways. For example, we may explore the use of different GNN architectures, or applying deep reinforcement learning techniques to a game playing reformulation of the problem.

- Nhan - Transformer Neural Network

  Transformer Neural Network is a subset of Recurrent Neural Network. Transformers are popular in Natural Language Processing for their ability to identify context in text. In terms of this project, transformer will be used for its positional embedding through Attention and Self-attention mechanisms to identify the relationship between each token in the MLTL traces and its formula. The input to the transformer will be the MLTL traces. The output will be the MLTL formula.

- Swaminathan - Template driven search

  A template-driven approach for MLTL inference uses predefined formula templates to guide the search for an MLTL formula consistent with given positive and negative traces. In this project, candidate MLTL formulas will be systematically generated by filling template placeholders through an automated search mechanism, prioritizing the efficiency of the search process. Each candidate is then evaluated against a set of positive and negative traces. Formula selection will be performed based on additional considerations such as simplicity and generalizability to ensure practical utility.

- Luke - Informed search (A*)

  This approach will formulate MLTL inference into a search problem then apply A*. Multiple heuristic functions will be evaluated and compared. Heuristic functions may consider factors such as accuracy against positive and negative traces and length or complexity of the formula. Due to the search space growing exponentially as formula length increases it will likely be necessary to employ inadmissible heuristics which could greatly accelerate the search but could result in suboptimal solutions. An analysis will be conducted to assess the trade-offs between execution time and solution optimality using admissible versus inadmissible heuristic functions.

References

[1] Dana Angluin. Learning regular sets from queries and counterexamples. *Information and Computation*, 75(2):87–106, 1987.

[2] Alberto Camacho and Sheila A. McIlraith. Learning interpretable models expressed in linear temporal logic. *Proceedings of the International Conference on Automated Planning and Scheduling*, 29(1):621–630, May 2021.

[3] James B Dabney, Julia M Badger, and Pavan Rajagopal. Adding a verification view for an autonomous real-time system architecture. In *AIAA Scitech 2021 Forum*, page 0566, 2021.

[4] Jenna Elwing, Laura Gamboa-Guzman, Jeremy Sorkin, Chiara Travesset, Zili Wang, and Kristin Yvonne Rozier. Mission-time LTL (MLTL) formula validation via regular expressions. In Paula Herber and Anton Wijs, editors, *iFM 2023*, pages 279–301, Cham, 2024. Springer Nature Switzerland.

[5] Abigail Hammer, Matthew Cauwels, Benjamin Hertz, Phillip Jones, and Kristin Yvonne Rozier. Integrating Runtime Verification into an Automated UAS Traffic Management System. *Innovations in Systems and Software Engineering: A NASA Journal*, July 2021.

[6] Brian Kempa, Pei Zhang, Phillip H. Jones, Joseph Zambreno, and Kristin Yvonne Rozier. Embedding Online Runtime Verification for Fault Disambiguation on Robonaut2. In *Proceedings of the 18th International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS)*, volume 12288 of *Lecture Notes in Computer Science (LNCS)*, pages 196–214, Vienna, Austria, September 2020. Springer.

[7] Danyang Li, Mingyu Cai, Cristian-Ioan Vasile, and Roberto Tron. Learning signal temporal logic through neural network for interpretable classification. In *2023 American Control Conference (ACC)*, pages 1907–1914, 2023.

[8] Weilin Luo, Pingjia Liang, Jianfeng Du, Hai Wan, Bo Peng, and Delong Zhang. Bridging ltlf inference to gnn inference for learning ltlf formulae. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(9):9849–9857, Jun. 2022.

[9] Naoko Okubo. Using R2U2 in JAXA program. Electronic correspondence, November–December 2020. Series of emails and zoom call from JAXA with technical questions about embedding MLTL formula monitoring into an autonomous satellite mission with a provable memory bound of 200KB.

[10] Rajarshi Roy, Jean-Raphaël Gaglione, Nasim Baharisangari, Daniel Neider, Zhe Xu, and Ufuk Topcu. Learning interpretable temporal properties from positive examples only. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'23/IAAI'23/EAAI'23. AAAI Press, 2023.