LINEAR TEMPORAL LOGIC MODULO THEORIES OVER FINITE TRACES

Nicola Gigante
Free University of Bozen-Bolzano, Italy

AAAI Spring Symposium 2023
San Francisco, CA, USA
March 29, 2023

Who are we?

Luca Geatti
University of Udine

Formal verification,
automated synthesis,
temporal logics.

Alessandro Gianola
University of Bolzano

Formal verification,
data-aware systems,
business process modeling.

Nicola Gigante
University of Bolzano

Formal verification,
temporal planning,
temporal logics.

Linear Temporal Logic

Linear Temporal Logic (LTL) is the most common formalism to specify temporal properties in **formal verification** and **artificial intelligence**.

Limits of propositional logics

The **propositional** nature of LTL and similar logics limits them to **finite-state** systems.

However, many scenarios are difficult or impossible to abstract finitely:

- systems involving **arithmetics**
- systems involving complex and unbounded **data structures**
- systems involving **relational databases**

LTL modulo theories

For this reason, we introduced **LTLf modulo theories** ($\mathsf{LTL}^{\mathsf{MT}}$) [GGG22]:

- first-order extension of LTLf
- propositions are replaced by **first-order sentences** over arbitrary theories, *à la* SMT
- (semi-)decision procedures based on off-the-shelf **SMT solvers**

LTL modulo theories

Many first-order extensions of LTL have been studied, however:

- many first-order temporal logics have been extensively studied from **theoretical** perspectives but without any practical development (see, *e.g.* [Kon+04])
- others led to practically applicable approaches but support quite **ad-hoc** syntax and semantics (see, *e.g.* [Cim+20])

Our approach is at the same time **theoretically** well-grounded, **general**, and **practically** oriented.

LTLf^{MT} is supported by our **BLACK**[1] temporal reasoning framework:[2]

- a software library and tool for **temporal reasoning** in linear-time logics
- supports LTL/LTLf and LTLf^{MT} in many flavors
- playground for many of our research directions

[1] Bounded LTL sAtisfiability ChecKer
[2] https://www.black-sat.org

Data-aware systems

Data-aware systems

Systems that involve the processing and manipulation of **data** taken from an **infinite** domain.

Examples:

- (relational) database-driven systems
- systems involving complex data-structures
- systems involving arithmetics
- any combination of the above!

Data-aware systems are **infinite-state**, leading very easily to **undecidability** of verification, model-checking, satisfiability etc ...

But they are still worth studying!

LTLf modulo theories

LTLf$^{\text{MT}}$ is our take at the verification of **infinite-state** systems.

LTLf$^{\text{MT}}$ extends LTLf by replacing **propositions** with **first-order sentences**.

- symbols can be uninterpreted, or interpreted by arbitrary first-order theories
  - *e.g.*, $+$, $<$ interpreted as integer sum/comparison
- constants, relational/function symbols, *etc.* can be both **rigid** or **non-rigid**
- interpreted over **finite-traces**

Examples

$$\mathsf{G}(x = 2y) \qquad (x < y)\,\mathsf{U}\,(y = 0) \qquad \mathsf{G}(x > 5) \wedge \mathsf{F}(x = 0)$$

$$\mathsf{G}(\exists y(x = 2y))$$

Examples

$$\mathsf{G}(x = 2y) \qquad (x < y)\ \mathsf{U}\ (y = 0) \qquad \mathsf{G}(x > 5) \wedge \mathsf{F}(x = 0)$$

$$\mathsf{G}(\exists y (x = 2y))$$

Examples

$$\mathsf{G}(x = 2y) \qquad (x < y)\,\mathsf{U}\,(y = 0) \qquad \mathsf{G}(x > 5) \wedge \mathsf{F}(x = 0)$$

$$\mathsf{G}(\exists y (x = 2y))$$

Examples

$$\mathsf{G}(x = 2y) \qquad (x < y)\,\mathsf{U}\,(y = 0) \qquad \mathsf{G}(x > 5) \wedge \mathsf{F}(x = 0)$$

$$\mathsf{G}(\exists y (x = 2y))$$

Examples

$$\mathsf{G}(x = 2y) \qquad (x < y)\,\mathsf{U}\,(y = 0) \qquad \mathsf{G}(x > 5) \wedge \mathsf{F}(x = 0)$$

$$\mathsf{G}(\exists y(x = 2y))$$

Examples

$$x = 0 \land ((\bigcirc x = x + 1) \, \mathsf{U} \, x = 42)$$

$$y = 1 \land \mathsf{G}(\widetilde{\bigcirc} \, y = y + 1 \land x = 2y)$$

$$p(0) \land \mathsf{G}\forall x(p(x) \to \widetilde{\mathsf{X}} p(x + 1)) \land \mathsf{F} p(42)$$

Examples

$$x = 0 \land ((\bigcirc x = x + 1) \; \mathsf{U} \; x = 42)$$

$$y = 1 \land \mathsf{G}(\circledsim y = y + 1 \land x = 2y)$$

$$p(0) \land \mathsf{G}\forall x(p(x) \to \widetilde{\mathsf{X}}p(x+1)) \land \mathsf{F}p(42)$$

Examples

$$x = 0 \wedge ((\bigcirc x = x + 1) \mathbin{\mathsf{U}} x = 42)$$

$$y = 1 \wedge \mathsf{G}(\widetilde{\bigcirc}\, y = y + 1 \wedge x = 2y)$$

$$p(0) \wedge \mathsf{G}\forall x(p(x) \rightarrow \widetilde{\mathsf{X}} p(x + 1)) \wedge \mathsf{F} p(42)$$

Examples

$$x = 0 \land ((\bigcirc x = x + 1) \, \mathsf{U} \, x = 42)$$

$$y = 1 \land \mathsf{G}(\odot\!\!\!\sim y = y + 1 \land x = 2y)$$

$$p(0) \land \mathsf{G}\forall x(p(x) \to \widetilde{\mathsf{X}} p(x + 1)) \land \mathsf{F} p(42)$$

Where's the catch?

LTLf^{MT} is clearly **undecidable**, but:

- over **decidable** first-order theories/fragments, it is **semi-decidable**
- our **semi-decision** procedure always answers **yes** for satisfiable formulas, **may** not terminate for unsatisfiable ones (but sometimes does)
- decidable theories and first-order fragments abound, *e.g.*:
  - linear integer/real **arithmetic** (LIA/LRA)
  - quantifier-free equality and **uninterpreted** functions (QF_EUF)
  - arrays, fixed-size bitvectors, algebraic data types, **floating-point** numbers, *etc.*
  - **effectively propositional** (EPR) logic: $\exists^*\forall^*\phi$
  - **two-variables** first-order logic (FO^2)

Why finite traces?

In propositional LTLf, finite traces makes everything simpler.

- *e.g.*, NFAs vs Büchi automata

However, complexities remain the same.

In the first-order world, this is not the case!

- LTLf$^{\mathsf{MT}}$ is semi-decidable for decidable first-order theories
- instead, for many decidable theories, LTL$^{\mathsf{MT}}$ is **not even semi-decidable**!

Why?

- the difference between **tiling** and recurrent tiling

So the finite-traces semantics is the only one giving us any **hope** of solving anything.

How to solve LTLf modulo theories

How do we test satisfiability of $\mathrm{LTLf}^{\mathrm{MT}}$ formulas?

- an **iterative** procedure tests the existence of models of length up to $k \geqslant 0$, for increasing values of k
- given an $\mathrm{LTLf}^{\mathrm{MT}}$ formula ϕ and a k, we build a **purely first-order formula** $\langle\phi\rangle_k$ that is **satisfiable** if and only if there is a model for ϕ of length at most k
- $\langle\phi\rangle_k$ is given to an off-the-shelf SMT solver

Some experiments

That's cool, but **does it work**?

- everything here is **undecidable**

Some experiments

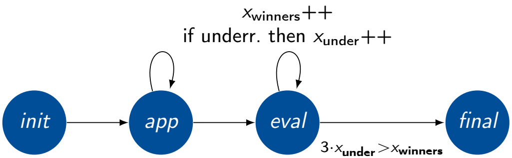
That's cool, but **does it work**?

- everything here is **undecidable**
- but. . .

Some early experiments

Test setting:

- simulation of a company hiring process
- nondeterministic transitions:
  - dependent on arithmetic constraints
  - acting on unbounded relational data
- minimal length of the counterexamples dependent over scalable parameter N
- two modelings of the same system:
  - P_1 employs arithmetic constraints
  - P_2 avoids arithmetics, simulates constraints by other means
- two different properties for each variant



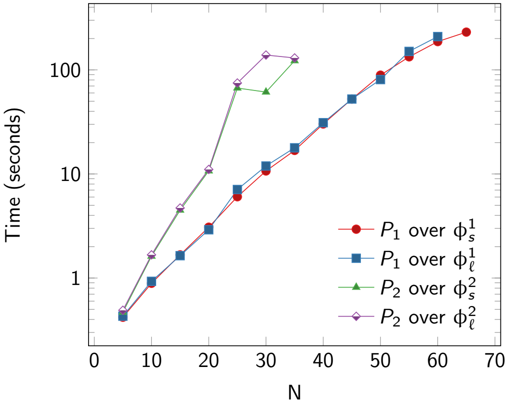
$$\phi_s^1 \equiv \mathsf{G}(x_{state} = \mathit{final} \rightarrow 2x_{\text{under}} > x_{\text{winners}})$$

$$\phi_\ell^1 \equiv \mathsf{G}\begin{pmatrix} x_{state} = \mathit{app} \rightarrow \\ F(x_{\text{state}} = \mathit{final} \wedge 2x_{\text{under}} > x_{\text{winners}}) \end{pmatrix}$$

Some early experiments

Results:

- 5 minutes timeout reached at $N = 70$
- **exponential** growth
  - but could be much worse, the problem is **undecidable**!
- liveness property not harder than the safety one
- system with **explicit arithmetics** faster to verify
- everything implemented in **BLACK**



Future directions

Where to go from now?

- find **decidable** $\mathsf{LTL}^{\mathsf{MT}}$ and $\mathsf{LTLf}^{\mathsf{MT}}$ fragments
- find more **efficient** $\mathsf{LTLf}^{\mathsf{MT}}$ fragments (not necessarily decidable)
- reactive **synthesis** for $\mathsf{LTLf}^{\mathsf{MT}}$ objectives
- theoretical properties of $\mathsf{LTLf}^{\mathsf{MT}}$
- **automata modulo theories**

THANK YOU

REFERENCES

[Cim+20] Alessandro Cimatti, Alberto Griggio, Enrico Magnago, Marco Roveri, and Stefano Tonetta. “SMT-based satisfiability of first-order LTL with event freezing functions and metric operators.” In: *Inf. Comput.* 272 (2020), p. 104502. DOI: 10.1016/j.ic.2019.104502.

[GGG22] Luca Geatti, Alessandro Gianola, and Nicola Gigante. “Linear Temporal Logic Modulo Theories over Finite Traces.” In: *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*. ijcai.org, 2022, pp. 2641–2647. DOI: 10.24963/ijcai.2022/366.

[Kon+04] Roman Kontchakov, Carsten Lutz, Frank Wolter, and Michael Zakharyaschev. “Temporalising Tableaux.” In: *Stud Logica* 76.1 (2004), pp. 91–134. DOI: 10.1023/B:STUD.0000027468.28935.6d.