Pure-Past Linear Temporal and Dynamic Logic on Finite Traces

Giuseppe De Giacomo[1], Antonio Di Stasio[1], Francesco Fuggitti[1,2], Sasha Rubin[3]

[1]Università degli Studi di Roma "La Sapienza", Roma, Italy
[2]York University, Toronto, ON, Canada
[3]University of Sydney, Sydney, NSW, Australia

{degiacomo, distasio, fuggitti}@diag.uniroma1.it, sasha.rubin@sydney.edu.au

Abstract

We review PLTL_f and PLDL_f, the pure-past versions of the well-known logics on finite traces LTL_f and LDL_f, respectively. PLTL_f and PLDL_f are logics about the past, and so scan the trace backwards from the end towards the beginning. Because of this, we can exploit a foundational result on reverse languages to get an exponential improvement, over $\text{LTL}_f/\text{LDL}_f$, for computing the corresponding DFA. This exponential improvement is reflected in several forms of sequential decision making involving temporal specifications, such as planning and decision problems in non-deterministic and non-Markovian domains. Interestingly, PLTL_f (resp., PLDL_f) has the same expressive power as LTL_f (resp., LDL_f), but transforming a PLTL_f (resp., PLDL_f) formula into its equivalent LTL_f (resp., LDL_f) is quite expensive. Hence, to take advantage of the exponential improvement, properties of interest must be directly expressed in $\text{PLTL}_f/\text{PLDL}_f$.

LTL_f and LDL_f

LTL_f is a variant of Linear-time Temporal Logic (LTL) interpreted on *finite*, instead of infinite, traces [1]. Given a set $\mathcal{P}$ of atomic propositions, LTL_f formulas φ are defined by:

$$\varphi ::= a \mid \neg\varphi \mid \varphi \wedge \varphi \mid \bigcirc\varphi \mid \varphi\,\mathcal{U}\,\varphi$$

where $a \in \mathcal{P}$, $\bigcirc$ is the **next** operator, and $\mathcal{U}$ is the **until** operator. Derived future temporal operators are: **eventually** $\Diamond\varphi \doteq \top\,\mathcal{U}\,\varphi$; **always** $\Box\varphi \doteq \neg\Diamond\neg\varphi$; and **weak next** $\bullet\varphi \doteq \neg\bigcirc\neg\varphi$.

LDL_f is a proper extension of LTL_f that captures regular expressions on finite traces.

$$\varphi ::= tt \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\varrho\rangle\varphi$$
$$\varrho ::= \phi \mid \varphi? \mid \varrho + \varrho \mid \varrho;\varrho \mid \varrho^*$$

where tt denotes the LDL_f true formula, ϕ propositional formulas over $\mathcal{P}$, ϱ path expressions, and $\varphi?$ the test construct. Derived operators are: $[\varrho]\varphi \doteq \neg\langle\varrho\rangle\neg\varphi$, $ff \doteq \neg tt$, and $end = [true]ff$. Intuitively, $\langle\varrho\rangle\varphi$ states that there exists an execution satisfying the RE ϱ such that its last step satisfies φ; whereas $[\rho]\varphi$ states that, from the current step, all executions satisfying the RE ϱ are such that their last step satisfies φ.

PLTL_f and PLDL_f

PLTL_f is the *pure-past* version of LTL_f. PLTL_f formulas are satisfied if they hold in the last instant of the trace. Given the set $\mathcal{P}$, PLTL_f formulas are defined as:

$$\varphi ::= a \mid \neg\varphi \mid \varphi \wedge \varphi \mid \ominus\varphi \mid \varphi\,\mathcal{S}\,\varphi$$

where $a \in \mathcal{P}$, $\ominus$ is the **previous** operator, and $\mathcal{S}$ is the **since** operator. Derived past temporal operators are: **once** $\diamondminus\varphi \doteq \top\,\mathcal{S}\,\varphi$; **historically** $\boxminus\varphi \doteq \neg\diamondminus\neg\varphi$.

PLDL_f is a proper extension of PLTL_f, hence the *pure-past* version of LDL_f.

$$\varphi ::= tt \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\!\langle\varrho\rangle\!\rangle\varphi$$
$$\varrho ::= \phi \mid \varphi? \mid \varrho + \varrho \mid \varrho;\varrho \mid \varrho^*$$

In PLDL_f, the LDL_f diamond operator is replaced by a backward diamond operator. Intuitively, $\langle\!\langle\varrho\rangle\!\rangle\varphi$ states that there exists a point in the past, reachable (going backwards) through the RE ϱ from the current instant, where φ holds. Derived operators are: $[\![\varrho]\!]\varphi \doteq \neg\langle\!\langle\varrho\rangle\!\rangle\neg\varphi$, $ff \doteq \neg tt$, and $start = [\![true]\!]ff$. Intuitively, $[\![\varrho]\!]\varphi$ states that, from the current step, all executions satisfying the RE ϱ (going backwards) are such that their last step in the past satisfies φ.

Examples

"every time you took the bus, you bought a new ticket beforehand"

- PLTL_f: $\boxminus(takeB \supset \ominus(\neg takeB\,\mathcal{S}\,buyT))$
- LTL_f: $(buyT\,\mathcal{R}\,takeB) \wedge \Box(takeB \supset (buyT \vee \bigcirc(buyT\,\mathcal{R}\,\neg takeB)))$

"every time, if the cargo-ship departed (cs), then beforehand there was an alternation of grab and unload (unl) of containers"

- PLDL_f: $[\![true^*]\!](\langle\!\langle cs\rangle\!\rangle tt \supset \langle\!\langle(unl;grab)^*;(unl;grab)\rangle\!\rangle start)$
- LDL_f: $\langle(\neg cs + (grab \wedge \neg cs);(unl;(grab \wedge \neg cs))^*;(cs \wedge unl));\neg cs^*\rangle end$

From φ to Automata

- For $\text{LTL}_f/\text{LDL}_f$ formulas the translation is worst-case 2EXPTIME:

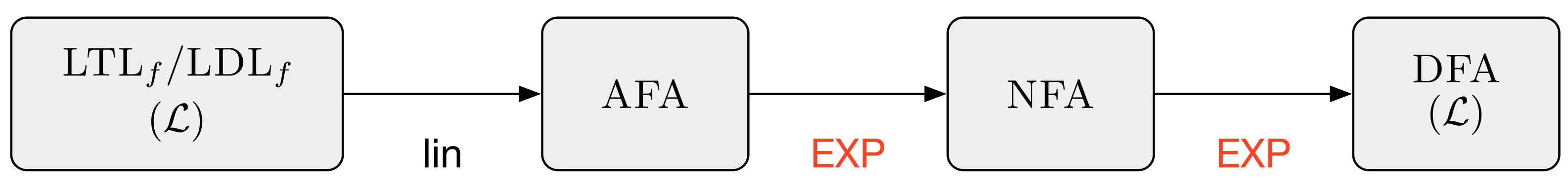


Figure 1: Translation algorithm from $\text{LTL}_f/\text{LDL}_f$ formulas to DFA

- For $\text{PLTL}_f/\text{PLDL}_f$ formulas the translation is worst-case EXPTIME:

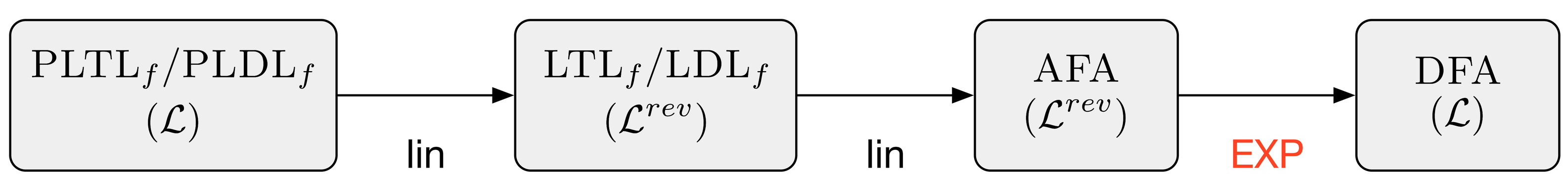
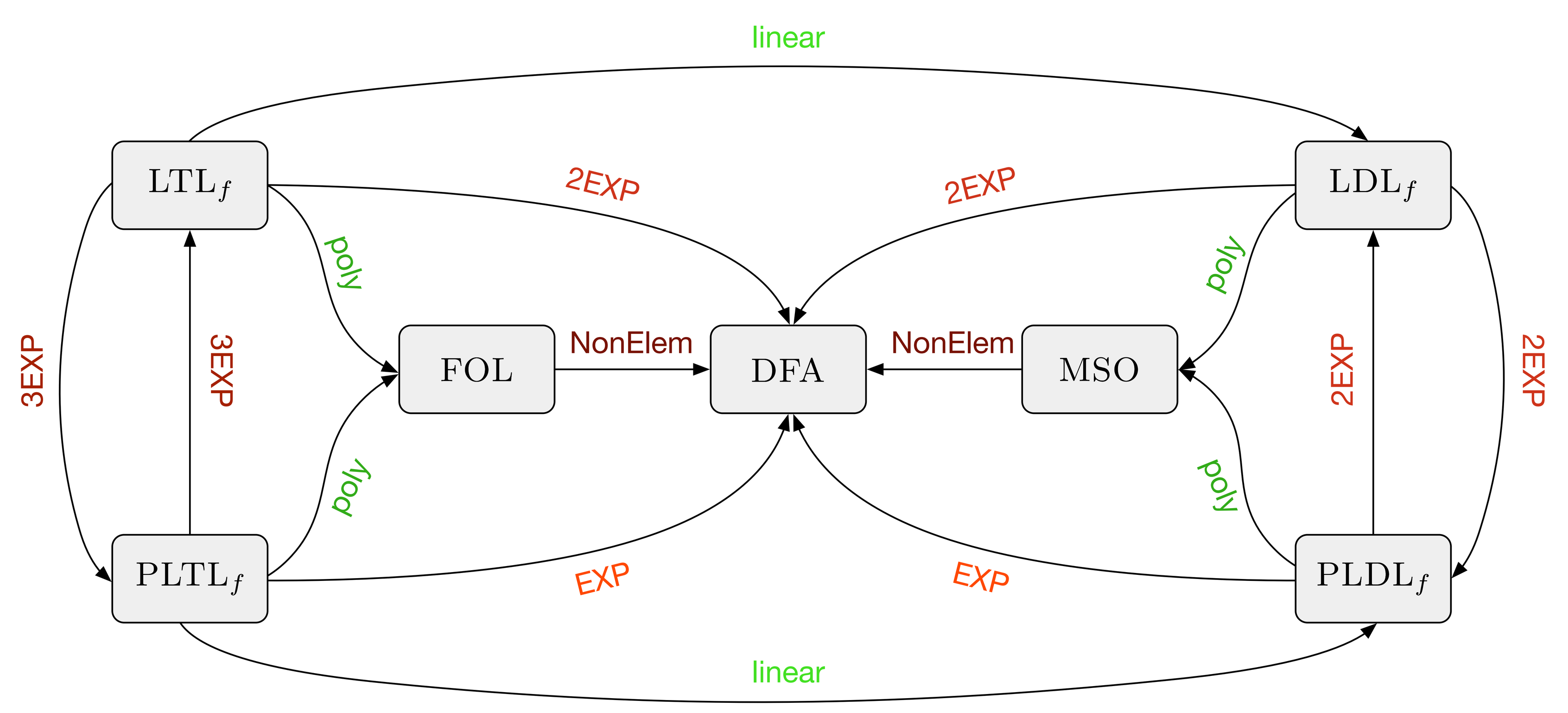


Figure 2: Translation algorithm from $\text{PLTL}_f/\text{PLDL}_f$ formulas to DFA

This is due to a fundamental property of *alternating finite-state automata* (AFA) [2], for which one can obtain directly the DFA of the *reverse* language, namely moving from a past view of the trace to a future one.

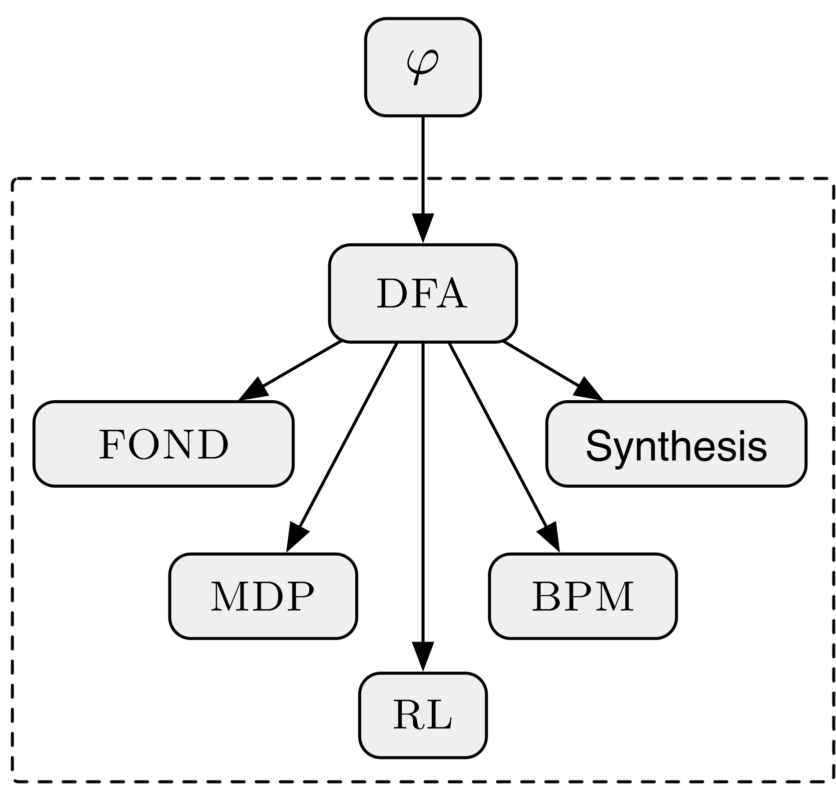
Transformations

Here, we summarize all the transformations and the relationship between $\text{LTL}_f/\text{LDL}_f$ and $\text{PLTL}_f/\text{PLDL}_f$ that we reviewed in the paper.



Implications

The exponential gain in transforming $\text{PLTL}_f/\text{PLDL}_f$ formulas into DFAs, with respect to $\text{LTL}_f/\text{LDL}_f$, is reflected in an exponential gain in solving several forms of sequential decision making problems involving temporal specifications.



Takeaways

1. If you can *naturally* express the specification in $\text{PLTL}_f/\text{PLDL}_f$, then do it to get the computational advantage
2. Converting $\text{LTL}_f/\text{LDL}_f$ to $\text{PLTL}_f/\text{PLDL}_f$ to get the exponential advantage is **not** computationally sensible
3. Complexities are just worst-case, in most AI applications the size of the resulting DFA is actually manageable

Acknowledgements

Work partially supported by the European Research Council under the European Union's Horizon 2020 Programme through the ERC Advanced Grant WhiteMech (No. 834228).

References

[1] G. De Giacomo and M. Vardi.
Linear temporal logic and linear dynamic logic on finite traces.
In *IJCAI*, 2013.

[2] A. Chandra, D. Kozen, and L. Stockmeyer.
Alternation.
J. of the ACM, 28(1), 1981.

SAPIENZA UNIVERSITÀ DI ROMA

YORK UNIVERSITÉ UNIVERSITY

THE UNIVERSITY OF SYDNEY