HyperLDL over Finite Traces

Giuseppe De Giacomo[1] Paolo Felli[2] Marco Montali[2] Giuseppe Perelli[1]

[1]Sapienza University of Rome [2]Free University of Bozen-Bolzano

Motivation

- The analysis of finite traces is important both in *Artificial Intelligence*, e.g. automated planning, and *Business Process Management*, e.g., process mining.
- LTL_f and LDL_f are temporal logics widely used for the analysis of dynamic systems with *finite traces* [1];
- Often, traces come as logs, that are (possibly infinite) sets of traces, typically generated by a *regular process*;
- When analyzing logs, it becomes of interest to understand the relationships among different traces, i.e., how different traces evolve with respect to one another;
- We propose a formalism based on LTL_f/LDL_f that is able to capture such relationship, namely $\mathrm{HyperLDL}_f$.

Our Contribution

- **$\mathrm{HyperLDL}_f$**: an extension of LDL_f incorporating quantifiers over (finite) traces;
- Decidability and complexity of the **model-checking** problem of $\mathrm{HyperLDL}_f$ over sets of regular languages;
- Algorithm based on classical finite automata, avoiding detour to infinite objects automata.

Syntax

$\varphi := \psi \mid \exists\pi\varphi \mid \forall\pi\varphi$	Trace quantifiers
$\psi := \mathsf{tt} \mid \mathsf{ff} \mid \neg\psi \mid \psi\wedge\psi \mid \psi\vee\psi \mid \langle\rho\rangle\psi \mid [\rho]\psi$	Boolean and temporal modalities
$\rho := \phi \mid \psi? \mid \rho+\rho \mid \rho;\rho \mid \rho^*$	Regular expressions
$\phi := p_\pi \mid \neg p_\pi \mid \phi\wedge\phi \mid \phi\vee\phi$	Boolean propositions

Classic syntactic sugar:

$\mathsf{true}_\pi \doteq p_\pi \vee \neg p_\pi$	propositional true over trace π
$\mathsf{false}_\pi \doteq \neg\mathsf{true}_\pi$	propositional false over trace π
$\mathsf{true}_P \doteq \wedge_{\pi\in P}\mathsf{true}_\pi$	propositional true over set of traces P
$\mathsf{false}_P \doteq \neg\mathsf{true}_P$	propositional false over set of traces P
$\mathtt{X}\psi \doteq \langle\mathsf{true}_{\mathsf{free}(\psi)};\psi?\rangle\mathsf{tt}$	next operator
$\tilde{\mathtt{X}}\psi \doteq \neg\mathtt{X}\neg\psi$	weak next operator
$\psi_1\mathtt{U}\psi_2 \doteq \langle(\psi_1?;\mathsf{true}_{\mathsf{free}(\psi_2)})^*;\psi_2?\rangle\mathsf{tt}$	until operator
$\mathtt{F}\psi \doteq \mathsf{true}_{\mathsf{free}(\psi)}\mathtt{U}\psi$	eventually operator
$\mathtt{G}\psi \doteq \neg\mathtt{F}\neg\psi$	globally operator
$\mathsf{end}_\pi \doteq [\mathsf{true}_\pi]\mathsf{ff}$	trace is ended
$\mathsf{last}_\pi \doteq \langle\mathsf{true}_\pi\rangle\mathsf{end}_\pi$	last event on the trace

Semantics

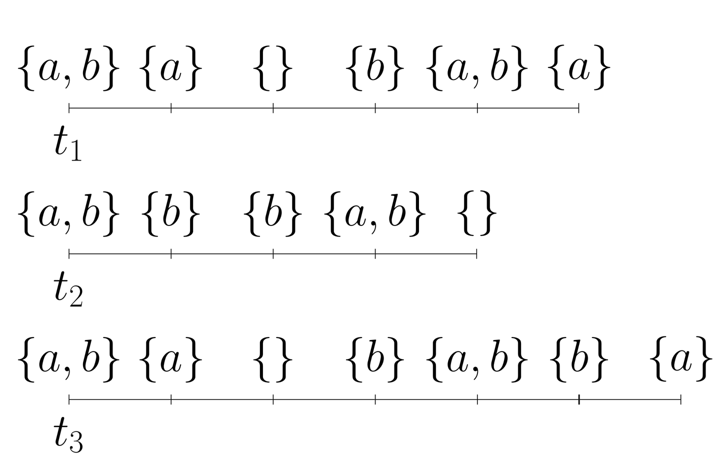
$\mathcal{E} \models \exists\pi\varphi$ if there is $t\in\mathcal{E}$ s.t. $\mathcal{E},[\pi\to t]\models\varphi$

$\mathcal{E} \models \forall\pi\varphi$ if, for each $t\in\mathcal{E}$, $\mathcal{E},[\pi\to t]\models\varphi$

Examples:

$\mathcal{E} \models \exists\pi\langle\mathsf{true}_\pi;a\rangle\mathsf{tt}$

$\mathcal{E} \not\models \forall\pi\langle\mathsf{true}_\pi;a\rangle\mathsf{tt}$



Examples

❶ Security [2]

Noninference $\varphi_{NI} = \forall\pi\exists\pi'(\mathtt{G}\lambda_{\pi'}) \wedge equal_L(\pi_1,\pi_2)$

Low level agents cannot infer any information on the high-level trace.

Observational Determinism $\forall\pi\forall\pi' \wedge_{p\in L_{in}} p_\pi \leftrightarrow p_{\pi'} \to \tilde{\mathtt{X}} \wedge_{p\in L_{out}} p_\pi \leftrightarrow p_{\pi'}$

The low-level user sees deterministic executions even when the executing program is nondeterministic.

❷ Process Mining [3]

Duty separation $\forall\pi_1,\pi_2\,(\wedge_{r\in Res}(\neg\mathtt{F}(\mathsf{open\_env},r)_{\pi_1} \vee \neg\mathtt{F}(\mathsf{record\_check},r)_{\pi_2}))$

Two tasks have to be performed by different resources, within the same instance or across all process instances.

Events in system logs Given an event log $\mathcal{E}$ and a pair of activities a and b appearing in $\mathcal{E}$, $\mathrm{HyperLDL}_f$ can represent and verify the basic ordering relations as defined in [4]:

$a >_{\mathcal{E}} b$	$\mathcal{E} \models \exists\pi\langle(\mathsf{true}_\pi)^*;a_\pi;b_\pi\rangle\mathsf{tt}$
$a \not>_{\mathcal{E}} b$	$\mathcal{E} \models \forall\pi[(\mathsf{true}_\pi)^*;a_\pi;b_\pi]\mathsf{ff}$
$a \to_{\mathcal{E}} b$	$\mathcal{E} \models a >_{\mathcal{E}} b \wedge b \not>_{\mathcal{E}} a$
$a \#_{\mathcal{E}} b$	$\mathcal{E} \models a \not>_{\mathcal{E}} b \wedge b \not>_{\mathcal{E}} a$
$a \Vert_{\mathcal{E}} b$	$\mathcal{E} \models a >_{\mathcal{E}} b \wedge b >_{\mathcal{E}} a$

❸ Instance-Spanning Constraints [5]

Repetition limit $\forall\pi_1,\ldots,\pi_{n+1}[(\mathsf{true}_P)^*;(\vee_{i\in\{1,\ldots,n+1\}} a_{\pi_i};\ldots;(\mathsf{true}_P)^*)^{n+1}]\mathsf{ff}$

Activity a cannot be executed, overall, more than n times.

Activity propagation $\forall\pi_1\forall\pi_2 rel(\pi_1,\pi_2) \to \mathtt{G}(a_{\pi_1} \to \mathtt{XF}\,b_{\pi_2})$

Whenever an activity a occurs in a trace π, then b has to occur later on in all traces *related to* π.

Solution techniques

Automata construction

❶ An LDL_f formula ψ is transformed into an alternating finite automaton $\mathcal{A}_\psi$ [6];

❷ Traces of the logs are represented by a deterministic finite automaton $\mathcal{D}$;

❸ Classic *existential* and *universal* projections correspond to the respective quantifications of a $\mathrm{HyperLDL}_f$ formula $\varphi = \mathtt{Q}n\pi_1\ldots\mathtt{Q}n\pi_n\psi$.

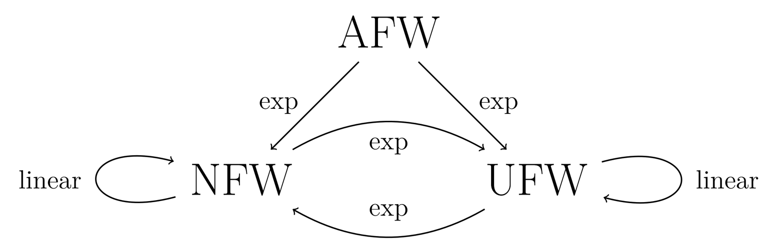
Model checking

Model checking is solved by *emptiness* of the automaton obtained in the procedure described above.

Computational analysis

❶ The alternating automaton $\mathcal{A}_\psi$ is of size linear with respect to the size of ψ.

❷ Every time the formula requires projecting from a nondeterministic to a universal automaton or vice-versa, an exponential blow-up in the size is necessary.



❸ The emptiness problem of a NFA is NLOGSPACE and can be done *on-the-fly* with the last projection. Without loss of generality, we can assume that the last projection is existential. Conversely, we solve the model-checking problem of $\neg\varphi$ and take the opposite answer.

Main Results

Theorem

For a given $\mathrm{HyperLDL}_f$ formula φ with quantifier alternation depth[a] k and a set of traces described by a DFA D, checking whether $\mathcal{D}\models\varphi$ can be solved in k−EXPSPACE in both the size of φ and $\mathcal{D}$, with 0−EXPSPACE = PSPACE.

[a] *How many times the formula switches from a universal quantification to an existential and vice-versa.*

Technique

Manipulates finite automata, avoiding the construction of those over ω-objects.

Publication

[1] Giuseppe De Giacomo and Moshe Y. Vardi. Linear temporal logic and linear dynamic logic on finite traces. In *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*, pages 854–860, 2013.

[2] Michael R. Clarkson, Bernd Finkbeiner, Masoud Koleini, Kristopher K. Micinski, Markus N. Rabe, and César Sánchez. Temporal logics for hyperproperties. In *ETAPS 2014*, pages 265–284, 2014.

[3] Wil M. P. van der Aalst et al. Process mining manifesto. In *Business Process Management Workshops (1)*, volume 99 of *Lecture Notes in Business Information Processing*, pages 169–194. Springer, 2011.

[4] Wil M. P. van der Aalst, Ton Weijters, and Laura Maruster. Workflow mining: Discovering process models from event logs. *IEEE Trans. Knowl. Data Eng.*, 16(9):1128–1142, 2004.

[5] Walid Fdhila, Manuel Gall, Stefanie Rinderle-Ma, Juergen Mangler, and Conrad Indiono. Classification and formalization of instance-spanning constraints in process-driven applications. In *BPM*, volume 9850 of *LNCS*, pages 348–364. Springer, 2016.

[6] Ronen I. Brafman, Giuseppe De Giacomo, and Fabio Patrizi. Ltlf/ldlf non-markovian rewards. In *AAAI 2018*, pages 1771–1778. AAAI Press, 2018.

Acknowledgements

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

ERC Advanced Grant
WhiteMech:
White-box Self Programming Mechanisms