Interval vs. Point Temporal Logic Model Checking

Angelo Montanari
Dept. of Mathematics, Computer Science, and Physics
University of Udine, Italy

(On leave at LaBRI - October/December 2017)

INFINI seminar
Cachan, France
November 30, 2017

Model checking

Model checking: the desired properties of a system are checked against a model of it

- the model is usually a (finite) state-transition system
- system properties are specified by a temporal logic (LTL, CTL, CTL^* and the like)

Distinctive features of model checking:

- exaustive check of all the possible behaviours
- fully automatic process
- a counterexample is produced for a violated property

Point-based vs. interval-based model checking

Model checking is usually point-based:

- properties express requirements over points (snapshots) of a computation (states of the state-transition system)
- they are specified by means of point-based temporal logics such as LTL, CTL, and CTL*

Interval properties express conditions on computation stretches instead of on computation states

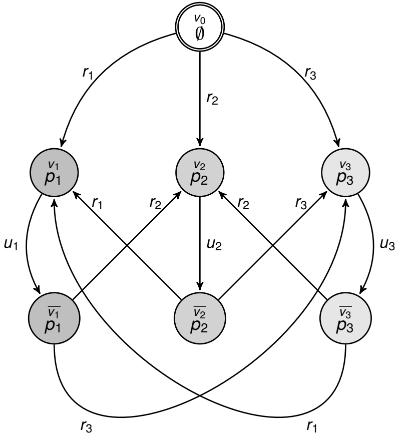
A lot of work has been done on interval temporal logic (ITL) satisfiability checking (an up-to-date survey can be found at: *https://users.dimi.uniud.it/~angelo.montanari/Movep2016-partI.pdf*).

ITL model checking entered the research agenda only recently (Bozzelli, Lomuscio, Michaliszyn, Molinari, Montanari, Murano, Perelli, Peron, Sala)

Outline of the talk

- The model checking problem for interval temporal logics
- Complexity results: the general picture
- Interval vs. point temporal logic model checking: an expressiveness comparison
- Ongoing work and future developments

The modeling of the system: Kripke structures



An example of Kripke structure

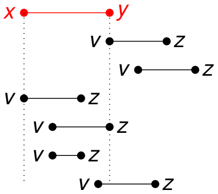
- HS formulas are interpreted over (finite) state-transition systems, whose states are labeled with sets of proposition letters (Kripke structures)
- An interval is a trace (finite path) in a Kripke structure

HS: the modal logic of Allen's interval relations

Allen's interval relations: the 13 binary ordering relations between 2 intervals on a linear order. They give rise to corresponding unary modalities over frames where intervals are primitive entities:

- HS features a modality for any Allen ordering relation between pairs of intervals (except for equality)

Allen rel.	HS	Definition	Example
meets	$\langle A \rangle$	$[x,y]\mathcal{R}_A[v,z] \iff y = v$	
before	$\langle L \rangle$	$[x,y]\mathcal{R}_L[v,z] \iff y < v$	
started-by	$\langle B \rangle$	$[x,y]\mathcal{R}_B[v,z] \iff x = v \wedge z < y$	
finished-by	$\langle E \rangle$	$[x,y]\mathcal{R}_E[v,z] \iff y = z \wedge x < v$	
contains	$\langle D \rangle$	$[x,y]\mathcal{R}_D[v,z] \iff x < v \wedge z < y$	
overlaps	$\langle O \rangle$	$[x,y]\mathcal{R}_O[v,z] \iff x < v < y < z$	



All modalities can be expressed by means of $\langle A \rangle$, $\langle B \rangle$, $\langle E \rangle$, and their transposed modalities only (if point intervals are admitted, $\langle B \rangle$, $\langle E \rangle$, and their transposed modalities suffice)

HS semantics and model checking

Truth of a formula ψ over a trace ρ of a Kripke structure $\mathcal{K} = (\mathcal{AP}, W, \delta, \mu, w_0)$ defined by induction on the complexity of ψ:

- $\mathcal{K}, \rho \models p$ iff $p \in \bigcap_{w \in \text{states}(\rho)} \mu(w)$, for any letter $p \in \mathcal{AP}$ (homogeneity assumption);
- clauses for negation, disjunction, and conjunction are standard;
- $\mathcal{K}, \rho \models \langle \text{A} \rangle\, \psi$ iff there is a trace ρ' s.t. $\text{lst}(\rho) = \text{fst}(\rho')$ and $\mathcal{K}, \rho' \models \psi$;
- $\mathcal{K}, \rho \models \langle \text{B} \rangle\, \psi$ iff there is a proper prefix ρ' of ρ s.t. $\mathcal{K}, \rho' \models \psi$;
- $\mathcal{K}, \rho \models \langle \text{E} \rangle\, \psi$ iff there is a proper suffix ρ' of ρ s.t. $\mathcal{K}, \rho' \models \psi$;
- the semantic clauses for $\langle \overline{\text{A}} \rangle$, $\langle \overline{\text{B}} \rangle$, and $\langle \overline{\text{E}} \rangle$ are similar

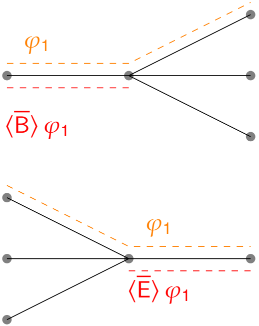
Model Checking

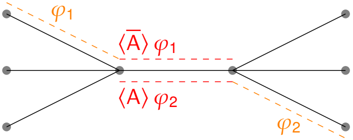
$\mathcal{K} \models \psi \iff$ for all *initial* traces ρ of $\mathcal{K}$, it holds that $\mathcal{K}, \rho \models \psi$

Possibly infinitely many traces!

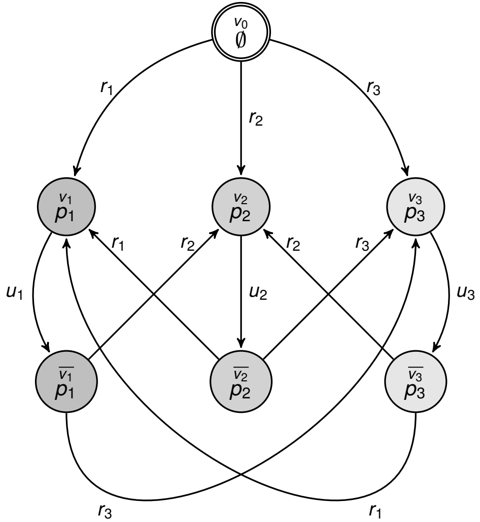
Remark: HS state semantics (HS_{st})

- According to the given semantics, HS modalities allow one to branch both in the past and in the future





The Kripke structure $\mathcal{K}_{Sched}$ for a simple scheduler



A short account of $\mathcal{K}_{Sched}$

$\mathcal{K}_{Sched}$ models the behaviour of a scheduler serving 3 processes which are continuously requesting the use of a common resource (it can be easily generalised to an arbitrary number of processes)

Initial state: v_0 (no process is served in that state)

In v_i and $\overline{v}_i$ the i-th process is served (p_i holds in those states)

The scheduler cannot serve the same process twice in two consecutive rounds:

- process i is served in state v_i, then, after "some time", a transition u_i from v_i to $\overline{v}_i$ is taken; subsequently, process i cannot be served again immediately, as v_i is not directly reachable from $\overline{v}_i$
- a transition r_j, with $j \neq i$, from $\overline{v}_i$ to v_j is then taken and process j is served

Some meaningful properties to be checked over $\mathcal{K}_{Sched}$

Validity of properties over all legal computation intervals can be forced by modality $[E]$ (they are suffixes of at least one initial trace)

Property 1: in any computation interval of length at least 4, at least 2 processes are witnessed (YES/no process can be executed twice in a row)

$$\mathcal{K}_{Sched} \models [E]\big(\langle \mathrm{E} \rangle^3 \top \rightarrow (\chi(p_1, p_2) \vee \chi(p_1, p_3) \vee \chi(p_2, p_3))\big),$$

where $\chi(p, q) = \langle \mathrm{E} \rangle \langle \overline{\mathrm{A}} \rangle p \wedge \langle \mathrm{E} \rangle \langle \overline{\mathrm{A}} \rangle q$

Property 2: in any computation interval of length at least 11, process 3 is executed at least once (NO/the scheduler can postpone the execution of a process ad libitum—starvation)

$$\mathcal{K}_{Sched} \not\models [E](\langle \mathrm{E} \rangle^{10} \top \rightarrow \langle \mathrm{E} \rangle \langle \overline{\mathrm{A}} \rangle p_3)$$

Property 3: in any computation interval of length at least 6, all processes are witnessed (NO/the scheduler should be forced to execute them in a strictly periodic manner, which is not the case)

$$\mathcal{K}_{Sched} \not\models [E](\langle \mathrm{E} \rangle^5 \rightarrow (\langle \mathrm{E} \rangle \langle \overline{\mathrm{A}} \rangle p_1 \wedge \langle \mathrm{E} \rangle \langle \overline{\mathrm{A}} \rangle p_2 \wedge \langle \mathrm{E} \rangle \langle \overline{\mathrm{A}} \rangle p_3))$$

Model checking: the key notion of BE_k-descriptor

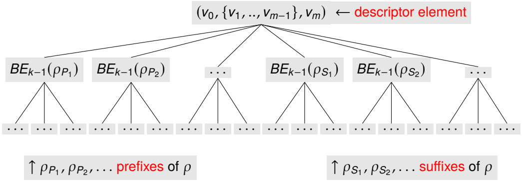
- The BE-nesting depth of an HS formula ψ ($\mathrm{Nest}_{\mathrm{BE}}(\psi)$) is the maximum degree of nesting of modalities B and E in ψ
- Two traces ρ and ρ' of a Kripke structure $\mathcal{K}$ are k-equivalent if and only if $\mathcal{K}, \rho \models \psi$ iff $\mathcal{K}, \rho' \models \psi$ for all HS-formulas ψ with $\mathrm{Nest}_{\mathrm{BE}}(\psi) \leq k$

Model checking: the key notion of BE_k-descriptor

- The BE-nesting depth of an HS formula ψ ($\text{Nest}_{\text{BE}}(\psi)$) is the maximum degree of nesting of modalities B and E in ψ
- Two traces ρ and ρ' of a Kripke structure $\mathcal{K}$ are k-equivalent if and only if $\mathcal{K}, \rho \models \psi$ iff $\mathcal{K}, \rho' \models \psi$ for all HS-formulas ψ with $\text{Nest}_{\text{BE}}(\psi) \leq k$

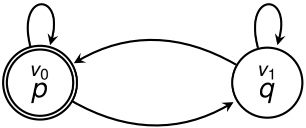
For any given k, we provide a suitable tree representation for a trace, called a BE_k-descriptor

The BE_k-descriptor for a trace $\rho = v_0 v_1 .. v_{m-1}\ v_m$, denoted $BE_k(\rho)$, has the following structure:

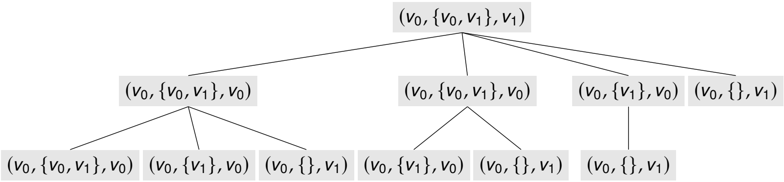


Remark: the descriptor does not feature sibling isomorphic subtrees

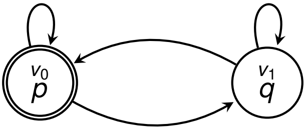
An example of a BE_2-descriptor



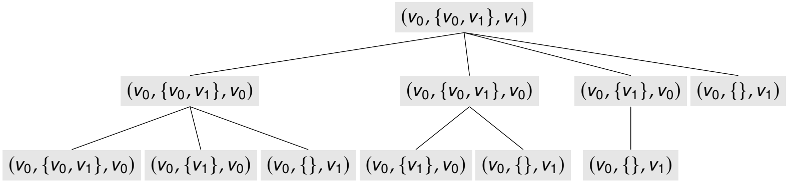
The BE_2-descriptor for the trace $\rho = v_0 v_1 v_0^4 v_1$ (for the sake of readability, only the subtrees for prefixes are displayed and point intervals are excluded)



An example of a BE_2-descriptor



The BE_2-descriptor for the trace $\rho = v_0 v_1 v_0^4 v_1$ (for the sake of readability, only the subtrees for prefixes are displayed and point intervals are excluded)



Remark: the subtree to the left is associated with both prefixes $v_0 v_1 v_0^3$ and $v_0 v_1 v_0^4$ (no sibling isomorphic subtrees in the descriptor)

Decidability of model checking for full HS

FACT 1: For any Kripke structure $\mathcal{K}$ and any BE-nesting depth $k \geq 0$, the number of different BE_k-descriptors is finite (and thus at least one descriptor has to be associated with infinitely many traces)

Decidability of model checking for full HS

FACT 1: For any Kripke structure $\mathcal{K}$ and any BE-nesting depth $k \geq 0$, the number of different BE_k-descriptors is finite (and thus at least one descriptor has to be associated with infinitely many traces)

FACT 2: Two traces ρ and ρ' of a Kripke structure $\mathcal{K}$ described by the same BE_k descriptor are k-equivalent

Decidability of model checking for full HS

FACT 1: For any Kripke structure $\mathcal{K}$ and any BE-nesting depth $k \geq 0$, the number of different BE_k-descriptors is finite (and thus at least one descriptor has to be associated with infinitely many traces)

FACT 2: Two traces ρ and ρ' of a Kripke structure $\mathcal{K}$ described by the same BE_k descriptor are k-equivalent

Theorem

The model checking problem for full HS *on finite Kripke structures is decidable (with a non-elementary algorithm)*

A. Molinari, A. Montanari, A. Murano, G. Perelli, and A. Peron, Checking Interval Properties of Computations, Acta Informatica, Special Issue: Temporal Representation and Reasoning (TIME'14), Vol. 56, n. 6-8, October 2016, pp. 587-619

Decidability of model checking for full HS

FACT 1: For any Kripke structure $\mathcal{K}$ and any BE-nesting depth $k \geq 0$, the number of different BE_k-descriptors is finite (and thus at least one descriptor has to be associated with infinitely many traces)

FACT 2: Two traces ρ and ρ' of a Kripke structure $\mathcal{K}$ described by the same BE_k descriptor are k-equivalent

Theorem

The model checking problem for full HS *on finite Kripke structures is decidable (with a non-elementary algorithm)*

A. Molinari, A. Montanari, A. Murano, G. Perelli, and A. Peron, Checking Interval Properties of Computations, Acta Informatica, Special Issue: Temporal Representation and Reasoning (TIME'14), Vol. 56, n. 6-8, October 2016, pp. 587-619

What about lower bounds?

The logic BE

Theorem

The model checking problem for BE*, over finite Kripke structures, is EXPSPACE-hard*

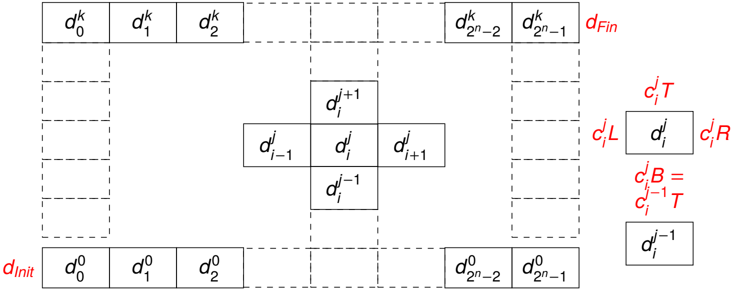
L. Bozzelli, A. Molinari, A. Montanari, A. Peron, and P. Sala, Interval Temporal Logic Model Checking: The Border Between Good and Bad HS Fragments, IJCAR 2016

Proof: a polynomial-time reduction from a domino-tiling problem for grids with rows of single exponential length

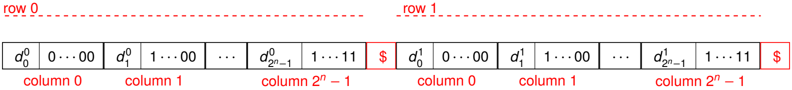
- for an instance $\mathcal{I}$ of the problem, we build a Kripke structure $\mathcal{K}_{\mathcal{I}}$ and a BE formula $\varphi_{\mathcal{I}}$ in polynomial time
- there is an initial trace of $\mathcal{K}_{\mathcal{I}}$ satisfying $\varphi_{\mathcal{I}}$ iff there is a tiling of $\mathcal{I}$
- $\mathcal{K}_{\mathcal{I}} \models \neg\varphi_{\mathcal{I}}$ iff there exists no tiling of $\mathcal{I}$

BE hardness: encoding of the domino-tiling problem

Instance of the tiling problem: $(C, \Delta, n, d_{init}, d_{final})$, with C a finite set of colors and $\Delta \subseteq C \times C \times C \times C$ a set of tuples (c_B, c_L, c_T, c_R)

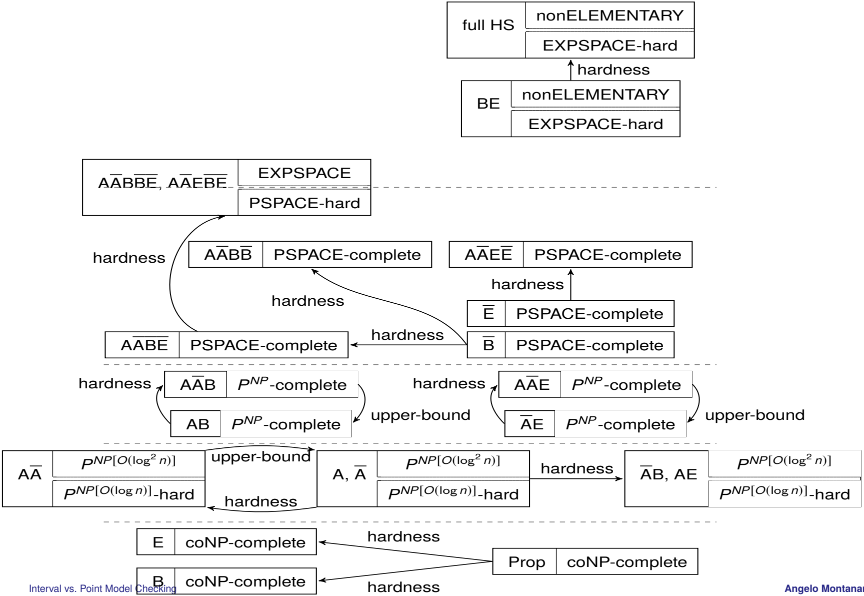


String (interval) encoding of the problem



The complexity picture

skip



Hardness results

- EXPSPACE-hardness of BE via a reduction from a domino-tiling problem
- PSPACE-hardness of $\overline{\text{B}}$ via a reduction from QBF
- P^{NP}-hardness of AB and $\overline{\text{A}}$E via a reduction from SNSAT (a logical problem with nested satisfiability questions)
- $P^{NP[O(\log n)]}$-hardness of A and $\overline{\text{A}}$ via a reduction from Parity-SAT (is the number of satisfiable formulas in a given set odd or even?)
- co-NP-hardness of Prop via a reduction from SAT to the not-model problem

Three main gaps to fill

There are three main gaps to fill:

- full HS and BE are in between nonELEMENTARY and EXPSPACE
- $A\overline{A}B\overline{B}\overline{E}$, $A\overline{A}E\overline{B}\overline{E}$, $AB\overline{B}\overline{E}$, $AE\overline{B}\overline{E}$, $\overline{A}B\overline{B}\overline{E}$, and $\overline{A}E\overline{B}\overline{E}$ are in between EXPSPACE and PSPACE
- A, $\overline{A}$, $A\overline{A}$, $\overline{A}B$, and AE are in between $P^{NP[O(\log^2 n)]}$ and $P^{NP[O(\log n)]}$

Point vs. interval temporal logic model checking

Question: is there any advantage in replacing points by intervals as the primary temporal entities, or is it just a matter of taste?

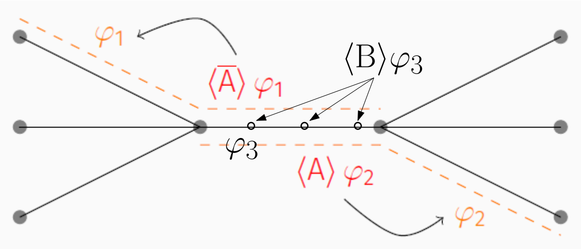
In order to compare the expressiveness of HS in model checking with those of LTL, CTL, and CTL*, we consider three semantic variants of HS:

- HS with state-based semantics (the original one)
- HS with computation-tree-based semantics
- HS with trace-based semantics

These variants are compared with the above-mentioned standard temporal logics and among themselves

L. Bozzelli, A. Molinari, A. Montanari, A. Peron, and P. Sala, Interval vs. Point Temporal Logic Model Checking: an Expressiveness Comparison. Proceedings of the 36th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS), December 2016, pp 26:1-14.

Branching semantic variant of HS

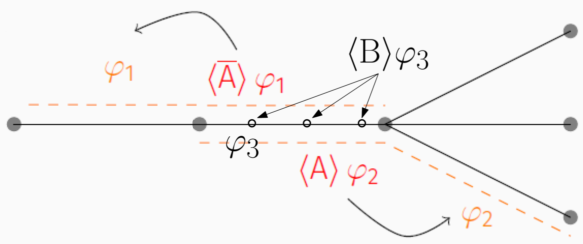


State-based semantics of HS (HS_{st}):

- both the future and the past are branching

A. Molinari, A. Montanari, A. Murano, G. Perelli, and A. Peron, Checking Interval Properties of Computations, Acta Informatica, Special Issue: Temporal Representation and Reasoning (TIME'14), Vol. 56, n. 6-8, October 2016, pp. 587-619

Linear-past semantic variant of HS

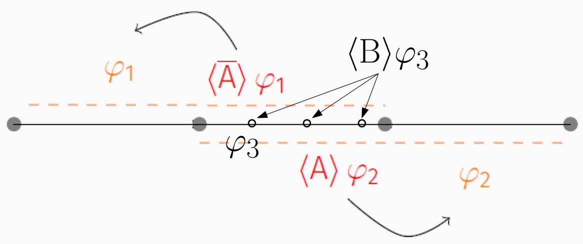


Computation-tree-based semantics of HS (HS_{lp}):

- the future is branching
- the past is linear, finite and cumulative
- similar to CTL* + linear past

A. Lomuscio and J. Michaliszyn, Decidability of model checking multi-agent systems against a class of EHS specifications, Proc. of the 21st European Conference on Artificial Intelligence (ECAI), August 2014, pp. 543–548

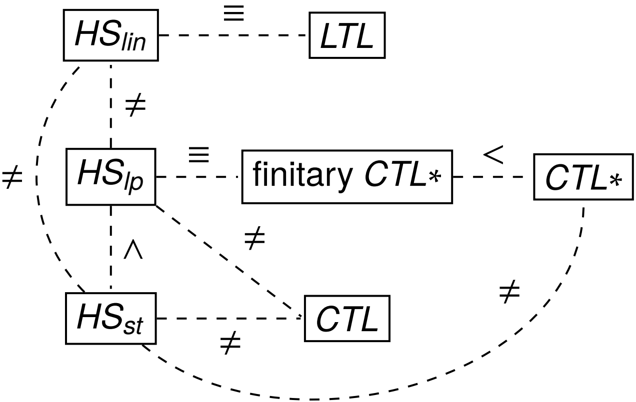
Linear semantic variant of HS



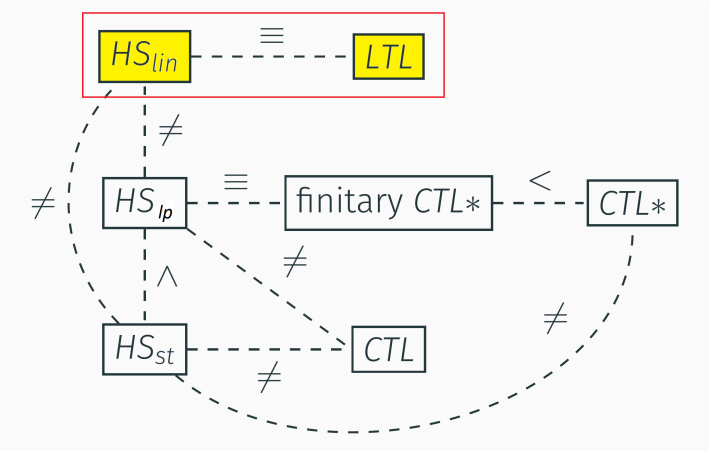
Trace-based semantics of HS (HS_{lin}):

- neither the past nor the future is branching
- similar to LTL + past

The expressiveness picture



Equivalence between LTL and HS_{lin}



Equivalence between LTL and HS_{lin}: LTL and FO

FO formulas φ (first-order fragment of MSO over infinite words):

$$\varphi := \top \mid p \in x \mid x \leq y \mid x < y \mid \neg\, \varphi \mid \varphi \,\wedge\, \varphi \mid \exists x.\varphi$$

- we interpret FO formulas φ over infinite paths π of Kripke structures
- a valuation function g assigns to each variable a position $i \geq 0$
- the satisfaction relation $(\pi, g) \models \varphi$ corresponds to the standard satisfaction relation $(\mu(\pi), g) \models \varphi$, where $\mu(\pi)$ is the infinite word over $2^{\mathcal{AP}}$ given by $\mu(\pi(0))\mu(\pi(1))\cdots$

Equivalence between LTL and HS_{lin}: LTL and FO

FO formulas φ (first-order fragment of MSO over infinite words):

$$\varphi := \top \mid p \in x \mid x \leq y \mid x < y \mid \neg \varphi \mid \varphi \wedge \varphi \mid \exists x . \varphi$$

- we interpret FO formulas φ over infinite paths π of Kripke structures
- a valuation function g assigns to each variable a position $i \geq 0$
- the satisfaction relation $(\pi, g) \models \varphi$ corresponds to the standard satisfaction relation $(\mu(\pi), g) \models \varphi$, where $\mu(\pi)$ is the infinite word over $2^{\mathcal{AP}}$ given by $\mu(\pi(0))\mu(\pi(1))\cdots$

Theorem (Kamp's theorem)

Given a FO sentence φ over $\mathcal{AP}$, one can construct an LTL formula ψ such that for all Kripke structures $\mathcal{K}$ over $\mathcal{AP}$ and infinite paths π,

$$\pi \models \varphi \iff \pi, 0 \models \psi$$

Equivalence between LTL and HS_{lin}: LTL $\geq HS_{lin}$

Given an HS_{lin} formula ψ, one can build an FO sentence ψ_{FO} such that, for all Kripke structures $\mathcal{K}$, it holds that

$\mathcal{K} \models_{lin} \psi$ iff for each initial infinite path π of $\mathcal{K}$, $\mathcal{K}, \pi \models \psi_{FO}$

$$\psi_{FO} = \exists x((\forall z.z \geq x) \wedge \forall y.h(\psi, x, y))$$

$$\begin{aligned}
h(p,x,y) &= \forall z.((z \geq x \wedge z \leq y) \longrightarrow p \in z) \\
h(\langle E \rangle \psi, x, y) &= \exists z.(z > x \wedge z \leq y \wedge h(\psi, z, y)) \\
h(\langle B \rangle \psi, x, y) &= \exists z.(z \geq x \wedge z < y \wedge h(\psi, x, z)) \\
h(\langle \overline{E} \rangle \psi, x, y) &= \exists z.(z < x \wedge h(\psi, z, y)) \\
h(\langle \overline{B} \rangle \psi, x, y) &= \exists z.(z > y \wedge h(\psi, x, z))
\end{aligned}$$

Theorem

LTL $\geq$ HS_{lin}

Equivalence between LTL and HS_{lin}: $HS_{lin} \geq$ LTL

The converse containment holds as well ($HS_{lin} \geq$ LTL)

Theorem

Given an LTL formula φ, we can construct in linear time an AB formula ψ such that φ in LTL is equivalent to ψ in AB_{lin}

$$f(p) = p, \text{ for each proposition letter } p$$

$$f(X\psi) = \langle A \rangle (length_2 \wedge \langle A \rangle (length_1 \wedge f(\psi))),$$

$$f(\psi_1 U \psi_2) = \langle A \rangle \Big(\langle A \rangle (length_1 \wedge f(\psi_2)) \wedge [B](\langle A \rangle (length_1 \wedge f(\psi_1)) \Big)$$

It holds that $\mathcal{K} \models \psi$ iff $\mathcal{K} \models_{lin} length_1 \rightarrow f(\psi)$

Corollary

HS_{lin} and LTL have the same expressive power

What about succinctness?

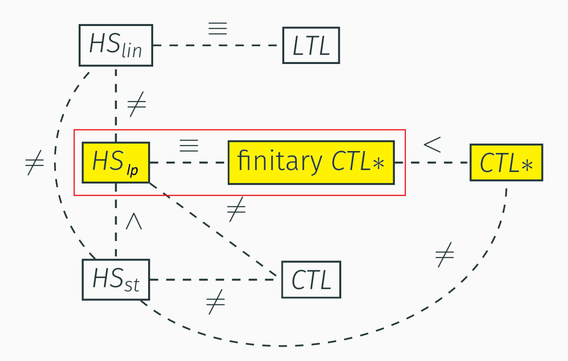
Things change if we consider succinctness: while it is possible to convert any LTL formula into an equivalent HS_{lin} one in linear time, HS_{lin} is at least exponentially more succinct than LTL

To prove it, it suffices to provide an HS_{lin} formula ψ for which there exists no LTL equivalent formula whose size is polynomial in $|\psi|$

We restrict our attention to the fragment BE_{lin}: since modalities $\langle \text{B} \rangle$ and $\langle \text{E} \rangle$ only allow one to 'move' from an interval to its subintervals, BE_{lin} actually coincides with BE_{st}, whose MC is known to be hard for EXPSPACE

A characterization of HS_{lp}

skip



A characterization of HS_{lp}: $HS_{lp} \geq$ finitary CTL^* - 1

We first show that finitary CTL^* is subsumed by HS_{lp} (finitary CTL^* = path quantification ranges over the traces starting from the current state)

A characterization of HS_{lp}: $HS_{lp} \geq$ finitary CTL* - 1

We first show that finitary CTL* is subsumed by HS_{lp} (finitary CTL* = path quantification ranges over the traces starting from the current state)

Preliminary step: when interpreted over finite words, the BE fragment of HS and LTL define the same class of finitary languages

Action-based semantics of BE ($L_{act}(\varphi)$):

- $L_{act}(a) = a^+$ for each $a \in \Sigma$;
- $L_{act}(\neg\varphi) = \Sigma^+ \setminus L_{act}(\varphi)$;
- $L_{act}(\varphi_1 \wedge \varphi_2) = L_{act}(\varphi_1) \cap L_{act}(\varphi_2)$;
- $L_{act}(\langle B \rangle\, \varphi) = \{w \in \Sigma^+ \mid \mathrm{Pref}(w) \cap L_{act}(\varphi) \neq \emptyset\}$;
- $L_{act}(\langle E \rangle\, \varphi) = \{w \in \Sigma^+ \mid \mathrm{Suff}(w) \cap L_{act}(\varphi) \neq \emptyset\}$.

A characterization of HS_{lp}: $HS_{lp} \geq$ finitary CTL* - 1

We first show that finitary CTL* is subsumed by HS_{lp} (finitary CTL* = path quantification ranges over the traces starting from the current state)

Preliminary step: when interpreted over finite words, the BE fragment of HS and LTL define the same class of finitary languages

Action-based semantics of BE ($L_{act}(\varphi)$):

- $L_{act}(a) = a^+$ for each $a \in \Sigma$;
- $L_{act}(\neg\varphi) = \Sigma^+ \setminus L_{act}(\varphi)$;
- $L_{act}(\varphi_1 \wedge \varphi_2) = L_{act}(\varphi_1) \cap L_{act}(\varphi_2)$;
- $L_{act}(\langle B \rangle\, \varphi) = \{w \in \Sigma^+ \mid \mathrm{Pref}(w) \cap L_{act}(\varphi) \neq \emptyset\}$;
- $L_{act}(\langle E \rangle\, \varphi) = \{w \in \Sigma^+ \mid \mathrm{Suff}(w) \cap L_{act}(\varphi) \neq \emptyset\}$.

Easy direction: over finite words, the class of languages defined by BE is subsumed by that defined by LTL

A characterization of HS_{lp}: $HS_{lp} \geq$ finitary CTL^* - 2

Converse direction: we exploit a sufficient condition for the inclusion of the class of LTL-definable languages, called LTL-closure, stating that any LTL-closed class C of finitary languages includes the class of LTL-definable finitary languages (Wilke)

A characterization of HS_{lp}: $HS_{lp} \geq$ finitary CTL* - 2

Converse direction: we exploit a sufficient condition for the inclusion of the class of LTL-definable languages, called LTL-closure, stating that any LTL-closed class C of finitary languages includes the class of LTL-definable finitary languages (Wilke)

Proposition

Let φ be an LTL formula over a finite alphabet Σ. Then, there exists a BE *formula φ_{HS} over Σ such that $L_{act}(\varphi_{HS}) = L_{act}(\varphi)$*

Proof: the class of BE-definable finitary languages is LTL-closed

A characterization of HS_{lp}: $HS_{lp} \geq$ finitary CTL* - 2

Converse direction: we exploit a sufficient condition for the inclusion of the class of LTL-definable languages, called LTL-closure, stating that any LTL-closed class C of finitary languages includes the class of LTL-definable finitary languages (Wilke)

Proposition

Let φ be an LTL formula over a finite alphabet Σ. Then, there exists a BE *formula φ_{HS} over Σ such that $L_{act}(\varphi_{HS}) = L_{act}(\varphi)$*

Proof: the class of BE-definable finitary languages is LTL-closed

Theorem

Let φ be a finitary CTL formula over $\mathcal{AP}$. Then, there is an* ABE *formula φ_{HS} over $\mathcal{AP}$ such that for all Kripke structures $\mathcal{K}$ over $\mathcal{AP}$ and tracks ρ, $\mathcal{K}, \rho, 0 \models \varphi$ iff $\mathcal{K}, \rho \models_{st} \varphi_{HS}$.*

A characterization of HS_{lp}: $\mathsf{HS}_{lp} \geq$ finitary CTL* - 2

Converse direction: we exploit a sufficient condition for the inclusion of the class of LTL-definable languages, called LTL-closure, stating that any LTL-closed class C of finitary languages includes the class of LTL-definable finitary languages (Wilke)

Proposition

Let φ be an LTL formula over a finite alphabet Σ. Then, there exists a BE *formula φ_{HS} over Σ such that $L_{act}(\varphi_{HS}) = L_{act}(\varphi)$*

Proof: the class of BE-definable finitary languages is LTL-closed

Theorem

Let φ be a finitary CTL formula over $\mathcal{AP}$. Then, there is an* ABE *formula φ_{HS} over $\mathcal{AP}$ such that for all Kripke structures $\mathcal{K}$ over $\mathcal{AP}$ and tracks ρ, $\mathcal{K}, \rho, 0 \models \varphi$ iff $\mathcal{K}, \rho \models_{st} \varphi_{HS}$.*

Since for ABE the computation-tree-based and the state-based semantics coincide, the following corollary holds:

both $\mathsf{HS}_{st} \geq$ finitary CTL* and $\mathsf{HS}_{lp} \geq$ finitary CTL*

A characterization of HS_{lp}: (finitary) $CTL^* \geq HS_{lp}$ - 1

Then, we show that HS_{lp} is subsumed by finitary CTL^* and CTL^*

A characterization of HS_{lp}: (finitary) CTL* $\geq$ HS_{lp} - 1

Then, we show that HS_{lp} is subsumed by finitary CTL* and CTL*

Hybrid CTL^*_{lp} (hybrid and linear past extension of CTL*):

$\varphi ::= \top \mid p \mid x \mid \neg\varphi \mid \varphi \vee \varphi \mid \downarrow x.\varphi \mid X\varphi \mid \varphi U \varphi \mid X^{-}\varphi \mid \varphi U^{-} \varphi \mid \exists\varphi$

- $\pi, g, i \models x \Leftrightarrow g(x) = i$
- $\pi, g, i \models \downarrow x.\varphi \Leftrightarrow \pi, g[x \leftarrow i], i \models \varphi$
- path quantification is "memoryful": it ranges over infinite paths that start at the root and visit the current node of the computation tree.

A characterization of HS_{lp}: (finitary) $CTL^* \geq HS_{lp}$ - 1

Then, we show that HS_{lp} is subsumed by finitary CTL^* and CTL^*

Hybrid CTL^*_{lp} (hybrid and linear past extension of CTL^*):

$\varphi ::= \top \mid p \mid x \mid \neg\varphi \mid \varphi \vee \varphi \mid \downarrow x.\varphi \mid X\varphi \mid \varphi U \varphi \mid X^{-}\varphi \mid \varphi U^{-} \varphi \mid \exists\varphi$

- $\pi, g, i \models x \Leftrightarrow g(x) = i$
- $\pi, g, i \models \downarrow x.\varphi \Leftrightarrow \pi, g[x \leftarrow i], i \models \varphi$
- path quantification is "memoryful": it ranges over infinite paths that start at the root and visit the current node of the computation tree.

Well-formed hybrid CTL^*_{lp}:

- each subformula $\exists\psi$ has at most one free variable
- each subformula $\exists\psi(x)$ of φ having x as free variable occurs in φ in the context $(F^{-}x) \wedge \exists\psi(x)$

Intuitively, for each state subformula $\exists\psi$, the unique free variable (if any) refers to ancestors of the current node in the computation tree

A characterization of HS_{lp}: (finitary) $\text{CTL}^* \geq \text{HS}_{\text{lp}}$ - 2

Proposition

Given a HS_{lp} formula φ, one can construct an equivalent well-formed sentence of hybrid CTL^*_{lp} (resp., finitary hybrid CTL^*_{lp})

(Not that difficult)

A characterization of $\mathrm{HS}_{\mathrm{lp}}$: (finitary) $\mathrm{CTL}^* \geq \mathrm{HS}_{\mathrm{lp}}$ - 2

Proposition

Given a $\mathrm{HS}_{\mathrm{lp}}$ formula φ, one can construct an equivalent well-formed sentence of hybrid CTL^*_{lp} (resp., finitary hybrid CTL^*_{lp})

(Not that difficult)

Proposition

Well-formed hybrid CTL^*_{lp} (resp., finitary hybrid CTL^*_{lp}) has the same expressiveness as CTL^* (resp., finitary CTL^*)

(Difficult!—It exploits the separation theorem for LTL + past)

A characterization of HS_{lp}: (finitary) $CTL^* \geq HS_{lp}$ - 2

Proposition

Given a HS_{lp} formula φ, one can construct an equivalent well-formed sentence of hybrid CTL^*_{lp} (resp., finitary hybrid CTL^*_{lp})

(Not that difficult)

Proposition

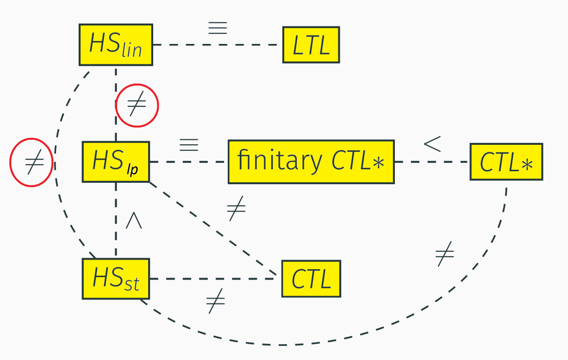
Well-formed hybrid CTL^*_{lp} (resp., finitary hybrid CTL^*_{lp}) has the same expressiveness as CTL^* (resp., finitary CTL^*)

(Difficult!—It exploits the separation theorem for LTL + past)

Theorem

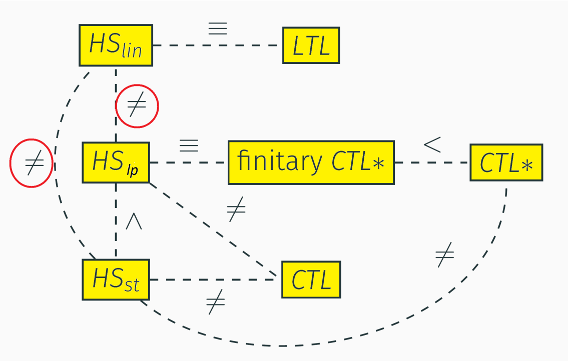
$CTL^* \geq HS_{lp}$. Moreover, HS_{lp} is as expressive as finitary CTL^*

A comparison of HS_{lin}, HS_{lp}, and HS_{st} - 1



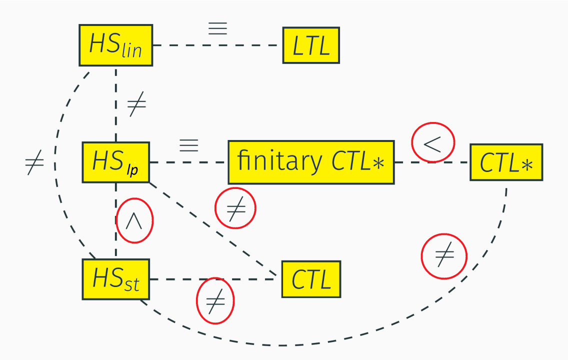
- The reachability condition $\forall G \exists F p$ (from each state reachable from the initial one, it is possible to reach a state where p holds) is not LTL-definable, but it is easily definable in HS_{lp} and HS_{st} by the formula $\langle \overline{B} \rangle \langle E \rangle p$
- The LTL formula Fp cannot be expressed in HS_{lp} or HS_{st} (Not immediate!)

A comparison of HS_{lin}, HS_{lp}, and HS_{st} - 1



- The reachability condition $\forall G \exists F p$ (from each state reachable from the initial one, it is possible to reach a state where p holds) is not LTL-definable, but it is easily definable in HS_{lp} and HS_{st} by the formula $\langle \overline{B} \rangle \langle E \rangle p$
- The LTL formula Fp cannot be expressed in HS_{lp} or HS_{st} (Not immediate!)

A comparison of HS_{lin}, HS_{lp}, and HS_{st} - 2



- We have already proved that $CTL^* \geq HS_{lp}$, $HS_{st} \geq HS_{lp}$
- HS_{lp}, CTL, CTL^* are not sensitive to unwinding, HS_{st} is
- The CTL formula $\forall F p$ cannot be expressed in HS_{lp} or HS_{st}
- The finitary CTL^* formula $\exists\big(((p_1 U p_2) \vee (q_1 U q_2))\, U\, r\big)$ cannot be expressed in CTL

Ongoing work and future developments - 1

Ongoing work: to determine the exact complexity of the satisfiability / model checking problems for BE over finite linear orders, under the homogeneity assumption (the three semantic variants of HS coincide over BE)

We know that the satisfiability/model checking problems for D over finite linear orders, under the homogeneity assumption, are PSPACE-complete (we exploit a spatial encoding of the models for D and a suitable contraction technique)

L. Bozzelli, A. Molinari, A. Montanari, A. Peron, and P. Sala, Satisfiability and Model Checking for the Logic of Sub-Intervals under the Homogeneity Assumption, Proc. of the 44th International Colloquium on Automata, Languages, and Programming(ICALP), LIPIcs 80, July 2017, pp. 120:1–120:14

Ongoing work and future developments - 2

- To fill the expressiveness gap between HS_{lp} and CTL^* by considering abstract interval models, induced by Kripke structures, featuring worlds also for infinite traces/intervals, and extending the semantics of HS modalities to infinite intervals
- To replace of Kripke structures by more expressive models
  - visibly pushdown systems, that can encode recursive programs and infinite state systems
  - inherently interval-based models, that allows one to directly describe systems on the basis of their interval behavior/properties, such as, for instance, those involving actions with duration, accomplishments, or temporal aggregations (no restriction on the evaluation of proposition letters)
- Application: planning as satisfiability checking / model checking in interval temporal logic