Complexity of timeline-based planning over dense temporal domains: exploring the middle ground

GandALF 2018

L. Bozzelli, A. Molinari, A. Montanari and A. Peron

University of Udine, IT

Department of Mathematics, Computer Science, and Physics (DMIF)

September 27, 2018

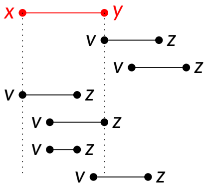
Point-based vs. interval-based MC

- Model checking (MC) 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-based MC:
  - Interval-based properties express conditions on computation stretches
  - they are specified by means of interval temporal logics, which feature intervals as their basic ontological entities (e.g., HS)
    - ability to express: actions with duration, accomplishments, temporal aggregations
    - applied to computational linguistics, artificial intelligence, temporal databases, formal verification

The logic HS

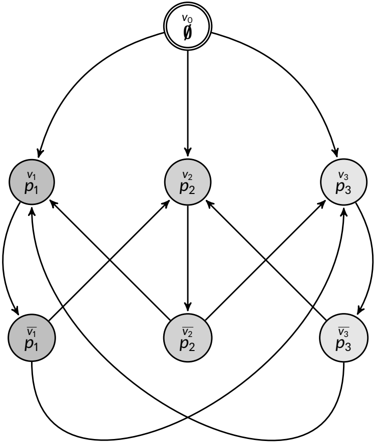
HS features a modality for each of the 13 Allen's ordering relations between pairs of intervals (except for equality)

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	$\langle O\rangle$	$[x,y]\mathcal{R}_O[v,z] \iff x < v < y < z$	



$$\boxed{\psi ::= p \mid \neg\psi \mid \psi \vee \psi \mid \langle X\rangle\psi \mid \langle\overline{X}\rangle\psi} \qquad X \in \{A, L, B, E, D, O\}.$$

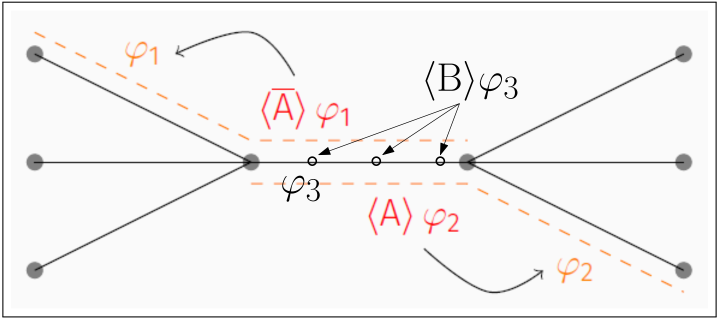
Kripke structures



- HS formulas are interpreted over (finite) state-transition systems whose states are labeled with sets of proposition letters (Kripke structures)

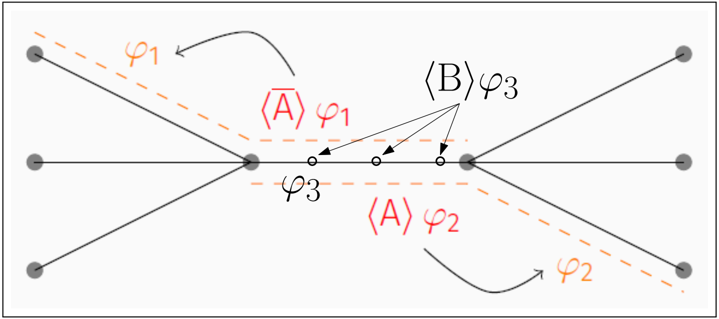
- An interval is a trace (finite path) in a Kripke structure

HS (state-based) semantics



- Branching semantics of past/future operators

HS (state-based) semantics



- Branching semantics of past/future operators

MC

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

Possibly infinitely many traces!

Decidability of HS MC

Theorem

The MC problem for full HS *over Kripke structures is decidable (with a non-elementary algorithm)*

Reference

A. Molinari, A. Montanari, A. Murano, G. Perelli, and A. Peron. Checking interval properties of computations.
Acta Informatica, pages 587–619, 2016

Decidability of HS MC

Theorem

The MC problem for full HS over Kripke structures is decidable (with a non-elementary algorithm)

Reference

A. Molinari, A. Montanari, A. Murano, G. Perelli, and A. Peron. Checking interval properties of computations.
Acta Informatica, pages 587–619, 2016

Theorem

The MC problem for BE over Kripke structures, under homogeneity, is **EXPSPACE***-hard*

Reference

L. Bozzelli, A. Molinari, A. Montanari, A. Peron, and P. Sala. Interval Temporal Logic Model Checking: the Border Between Good and Bad HS Fragments.
In *IJCAR*, pages 389–405, 2016

Many other HS fragments studied (**PSPACE** $\leftrightsquigarrow$ **NP**).

Ongoing work

We are looking for possible replacements of Kripke structures by more expressive system models in interval-based MC:

- interval-based system models, that allow one to directly describe systems on the basis of their interval behavior/properties (e.g., timelines).
- visibly pushdown systems, that can encode recursive programs and infinite state systems;

Timelines

- Timelines have been fruitfully exploited in temporal planning
- Timeline-based planning (TP for short) is a more declarative alternative to the classic action-based planning

Timelines

- Timelines have been fruitfully exploited in temporal planning
- Timeline-based planning (TP for short) is a more declarative alternative to the classic action-based planning
- Temporal domain commonly assumed discrete.
- Gigante et al. showed that TP with bounded temporal relations and token durations, and no temporal horizon, is **EXPSPACE**-complete and expressive enough to capture action-based temporal planning. (**EXPSPACE**-completeness also with unbounded relations)

Timelines

State variable

$$x = (V_x, T_x, D_x)$$

where, e.g.,

- $V_x = \{a, b, c\}$,
- $T_x(a) = \{b, c\}$, $T_x(b) = \{a, b, c\}$, $T_x(c) = \{a, b\}$ and
- $D_x(a) = [5, 8]$, $D_x(b) = [1, 4]$, $D_x(c) = [2, \infty[$

Timelines

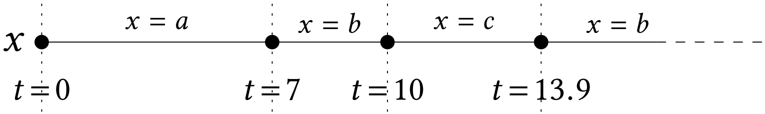
State variable

$$x = (V_x, T_x, D_x)$$

where, e.g.,

- $V_x = \{a, b, c\}$,
- $T_x(a) = \{b, c\}$, $T_x(b) = \{a, b, c\}$, $T_x(c) = \{a, b\}$ and
- $D_x(a) = [5, 8]$, $D_x(b) = [1, 4]$, $D_x(c) = [2, \infty[$

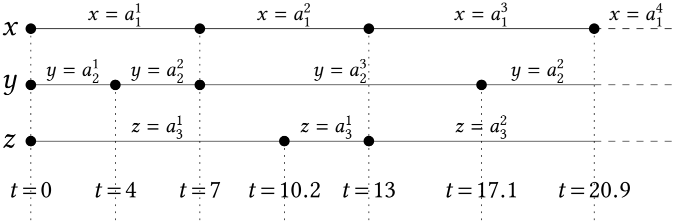
Example of timeline for x:



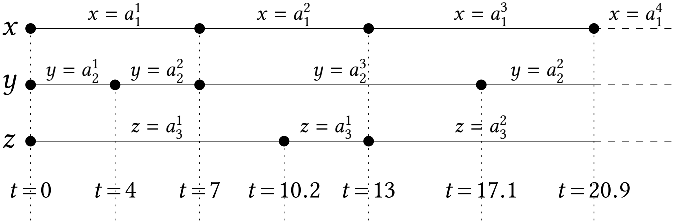
$$(a, 7)(b, 3)(c, 3.9)\cdots$$

Pairs of value/duration are called tokens.

Timelines


x
y
z
$x = a_1^1$
$x = a_1^2$
$x = a_1^3$
$x = a_1^4$
$y = a_2^1$
$y = a_2^2$
$y = a_2^3$
$y = a_2^2$
$z = a_3^1$
$z = a_3^1$
$z = a_3^2$
$t = 0$
$t = 4$
$t = 7$
$t = 10.2$
$t = 13$
$t = 17.1$
$t = 20.9$

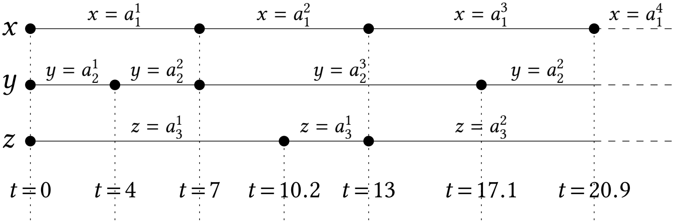
Timelines



Synchronization rules on timelines:

$$\forall o_0[x = a_1^1] \rightarrow \exists o_1[z = a_3^1]\exists o_2[y = a_2^2].\left(o_0 \leq^{e,s}_{[3,4]} o_1 \wedge o_0 \leq^{s,s}_{[5,\infty[} o_2\right)$$

Timelines



Synchronization rules on timelines:

$$\top \rightarrow \exists o_1[z = a_3^1]\exists o_2[y = a_2^2].\left(o_0 \leq^{e,s}_{[3,4]} o_1 \wedge o_0 \leq^{s,s}_{[5,\infty[} o_2\right)$$

Trigger-less rules.

Timelines as system models

- We study timeline-based planning (TP) over dense domains (no recourse to discretization).
    1. Why TP before MC? Timelines will be our system models. TP is a necessary condition for MC (feasibility check of the system description).
    2. Why dense domains? To avoid discreteness in system descriptions ⇒ abstraction at a higher level, neglecting unnecessary details, and paving the way for a more general interval-based MC;
- Both (i) the system model and (ii) the specifications (temporal formulas) can be translated into a common formalism (timed automata)

Undecidability of TP over dense domains

Theorem

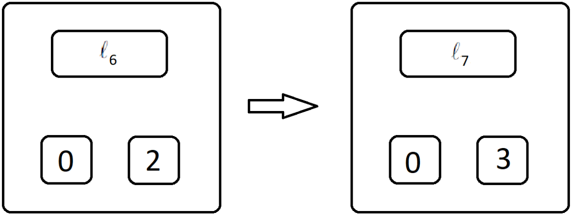
Timeline-based planning (TP) is undecidable over dense temporal domains, even with a single state variable.

Undecidability of TP over dense domains

Theorem

Timeline-based planning (TP) is undecidable over dense temporal domains, even with a single state variable.

- Undecidability proved via a reduction from the halting problem for Minsky 2-counter machines (inspired by SAT of Metric Temporal Logic with past/future on dense time).



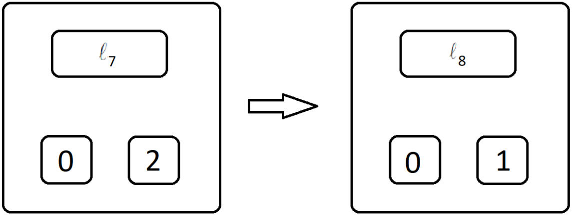
$\ell_6 : c_2 := c_2 + 1; \text{goto } \ell_7$

Undecidability of TP over dense domains

Theorem

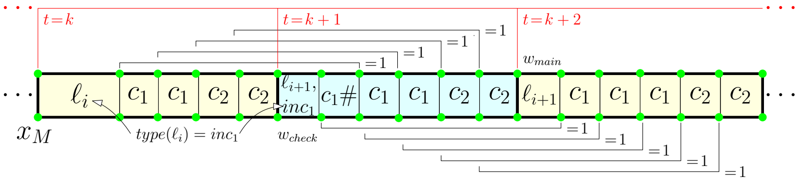
Timeline-based planning (TP) is undecidable over dense temporal domains, even with a single state variable.

- Undecidability proved via a reduction from the halting problem for Minsky 2-counter machines (inspired by SAT of Metric Temporal Logic with past/future on dense time).



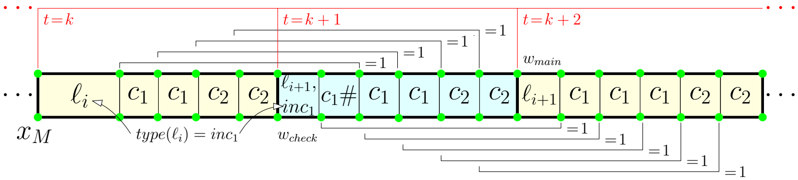
ℓ_7: if $c_2 > 0$ then $c_2 := c_2 - 1$; goto ℓ_8 else goto ℓ_{12}

Undecidability of TP over dense domains



- Exactly one occurrence of $\ell_{\texttt{init}}$ and $\ell_{\texttt{halt}}$ (transition function);

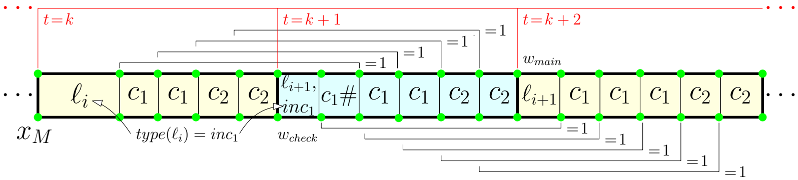
Undecidability of TP over dense domains



- Exactly one occurrence of $\ell_{\texttt{init}}$ and $\ell_{\texttt{halt}}$ (transition function);
- For each $v \in V_{\texttt{Ctrl}} \setminus \{\ell_{\texttt{halt}}\}$,

$$o[x_M = v] \rightarrow \bigvee_{u \in V_{\texttt{Ctrl}}} \exists\, o'[x_M = u].\, o \leq^{s,s}_{[1,1]} o'.$$

Undecidability of TP over dense domains



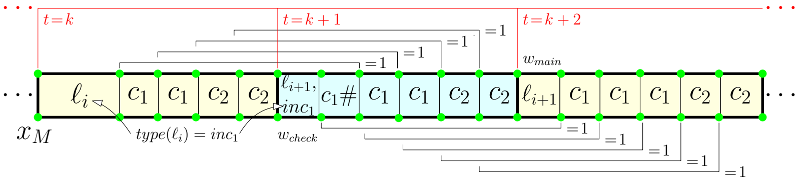
- Exactly one occurrence of $\ell_{\texttt{init}}$ and $\ell_{\texttt{halt}}$ (transition function);
- For each $v \in V_{\texttt{Ctrl}} \setminus \{\ell_{\texttt{halt}}\}$,

$$o[x_M = v] \rightarrow \bigvee_{u \in V_{\texttt{Ctrl}}} \exists\, o'[x_M = u].\, o \leq^{s,s}_{[1,1]} o'.$$

- For each $i = 1, 2$, $v \in (U_{c_i} \cap V_{\mathrm{main}}) \setminus V_{\texttt{halt}}$ (forward):

$$o[x_M = v] \rightarrow \bigvee_{u \in U_{c_i}} \exists\, o'[x_M = u].\, o \leq^{s,s}_{[1,1]} o'.$$

Undecidability of TP over dense domains



- Exactly one occurrence of $\ell_{\mathtt{init}}$ and $\ell_{\mathtt{halt}}$ (transition function);
- For each $v \in V_{\mathtt{Ctrl}} \setminus \{\ell_{\mathtt{halt}}\}$,

$$o[x_M = v] \rightarrow \bigvee_{u \in V_{\mathtt{Ctrl}}} \exists\, o'[x_M = u].\, o \leq^{s,s}_{[1,1]} o'.$$

- For each $i = 1, 2$, $v \in (U_{c_i} \cap V_{\mathrm{main}}) \setminus V_{\mathtt{halt}}$ (forward):

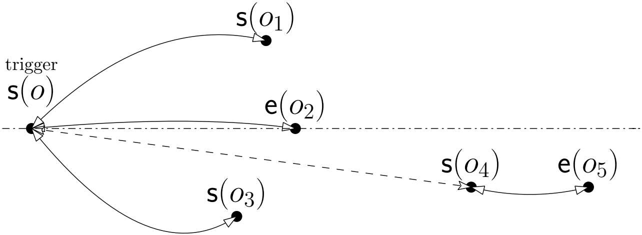
$$o[x_M = v] \rightarrow \bigvee_{u \in U_{c_i}} \exists\, o'[x_M = u].\, o \leq^{s,s}_{[1,1]} o'.$$

- For each $i = 1, 2$, $v \in (U_{c_i} \cap V_{\mathrm{check}}) \setminus V_{\mathtt{init}}$ (backward):

$$o[x_M = v] \rightarrow \bigvee_{u \in U_{c_i}} \exists\, o'[x_M = u].\, o' \leq^{s,s}_{[1,1]} o.$$

What happens if we restrict to future?

- Future: the token triggering a rule can only “refer” to other tokens in the future (i.e., starting after it).



Theorem

Future TP is non-primitive recursive-hard, even with a single state variable.

- Reduction from the halting problem for Gainy counter machines, known to be non-primitive recursive
- Only forward constraint can be expressed by future rules!

Decidability—(1) Translating rules

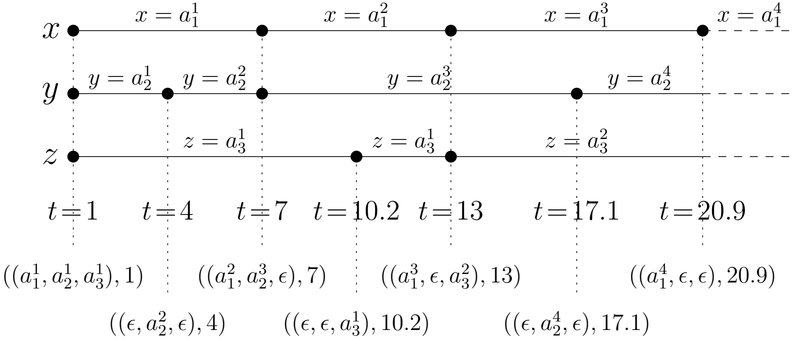
- Decidability of future TP with arbitrary trigger rules is open.
- We restrict to simple trigger rules:
  all existentially quantified tokens (but not the trigger!) occur just once in the rule.
- Decidability can be recovered if rules are simple and future.

Translation into MTL/MITL

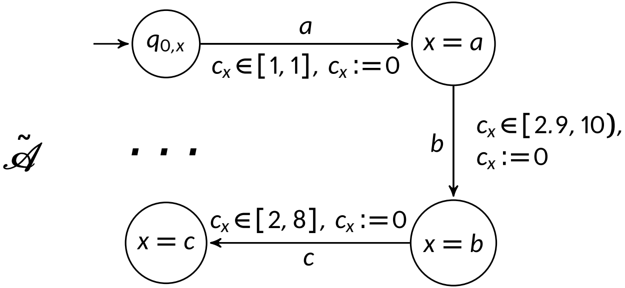
The simple form allows translation into MTL/MITL (future only+finite w!):

$$\varphi ::= \top \mid p \mid \varphi \vee \varphi \mid \neg\varphi \mid \varphi U_I \varphi$$

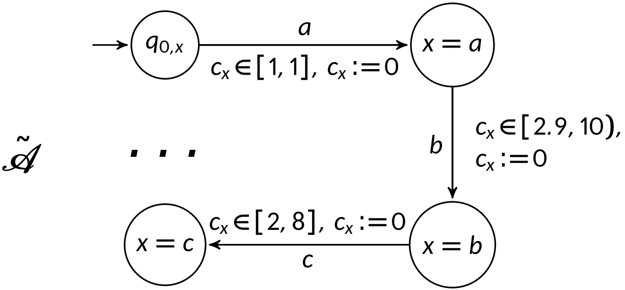
with $p \in \mathcal{AP}$, $I \in \mathtt{Intv}$, U_I is the standard *strict timed until* MTL modality



Decidability—(2) Translating state variables



Decidability—(2) Translating state variables



Theorem

Future TP with simple trigger rules is decidable (in non-primitive recursive time). If the intervals in atoms of the trigger rules are non-singular (resp., belong to $\mathtt{Intv}_{(0,\infty)}$), then it is in ***EXPSPACE*** *(resp., in* ***PSPACE****).*

EXPSPACE-completeness (resp., **PSPACE**-completeness) holds.

Timeline-based planning and MC: results

System model:

$$x_{\text{temp}} = (V_{\text{temp}}, T_{\text{temp}}, D_{\text{temp}})$$

$$x_{\text{proc}} = (V_{\text{proc}}, T_{\text{proc}}, D_{\text{proc}})$$

$$x_{\text{transm}} = (V_{\text{transm}}, T_{\text{transm}}, D_{\text{transm}})$$

$$+$$

Property specification:

Timeline-based planning and MC: results

System model:

$$x_{\text{temp}} = (V_{\text{temp}}, T_{\text{temp}}, D_{\text{temp}})$$

$$x_{\text{proc}} = (V_{\text{proc}}, T_{\text{proc}}, D_{\text{proc}})$$

$$x_{\text{transm}} = (V_{\text{transm}}, T_{\text{transm}}, D_{\text{transm}})$$

$$+$$

$$\forall o[x_{\texttt{proc}} = \texttt{reading}_1] \to$$
$$(\exists o_1[x_{\texttt{proc}} = \texttt{read}_0].o \leq^{e,s}_{[0,1]} o_1) \vee$$
$$(\exists o_2[x_{\texttt{proc}} = \texttt{read}_1]\exists o_3[x_{\texttt{temp}} = \texttt{ready}].o \leq^{e,s}_{[0,1]} o_2 \wedge o_3 \leq^{e,e}_{[0,+\infty[} o).$$

Property specification:

Timeline-based planning and MC: results

System model:

$$x_{\text{temp}} = (V_{\text{temp}}, T_{\text{temp}}, D_{\text{temp}})$$

$$x_{\text{proc}} = (V_{\text{proc}}, T_{\text{proc}}, D_{\text{proc}})$$

$$x_{\text{transm}} = (V_{\text{transm}}, T_{\text{transm}}, D_{\text{transm}})$$

$$+$$

$$\forall o[x_{\mathtt{proc}} = \mathtt{reading}_1] \rightarrow$$
$$(\exists o_1[x_{\mathtt{proc}} = \mathtt{read}_0].o \leq^{e,s}_{[0,1]} o_1) \vee$$
$$(\exists o_2[x_{\mathtt{proc}} = \mathtt{read}_1]\exists o_3[x_{\mathtt{temp}} = \mathtt{ready}].o \leq^{e,s}_{[0,1]} o_2 \wedge o_3 \leq^{e,e}_{[0,+\infty[} o).$$

Property specification:

$$F_{\leq 8}\ \psi(s, \mathtt{read}_1) \qquad F_{\geq 0}\big(\psi(s, \mathtt{ready}) \wedge (\top\, U_{>0}\ \psi(s, \mathtt{ready}))\big)$$

Timeline-based planning and MC: results

Given a system model P (state vars + rules), it is possible to build a TA $\tilde{\mathscr{A}}$ that accepts all and only the (timed words encoding) computations of P.

Definition (Timeline-based model checking)

Given a system model (in the form of) $\tilde{\mathscr{A}}$ and a MITL formula φ, to decide if

$$\mathscr{L}_T(\tilde{\mathscr{A}}) \subseteq \mathscr{L}_T(\varphi).$$

Timeline-based planning and MC: results

Given a system model P (state vars + rules), it is possible to build a TA $\tilde{\mathscr{A}}$ that accepts all and only the (timed words encoding) computations of P.

Definition (Timeline-based model checking)

Given a system model (in the form of) $\tilde{\mathscr{A}}$ and a MITL formula φ, to decide if

$$\mathscr{L}_T(\tilde{\mathscr{A}}) \subseteq \mathscr{L}_T(\varphi).$$

We make a product between $\tilde{\mathscr{A}}$ and $\mathscr{A}_{\neg\varphi}$ and check for emptiness.

Theorem

The MC problem for MITL formulas over timelines, with simple future trigger rules and non-singular intervals, is in **EXPSPACE**.
The MC problem for $\text{MITL}_{(0,\infty)}$ *formulas over timelines, with simple future trigger rules and intervals in* $\mathtt{Intv}_{(0,\infty)}$*, is in* **PSPACE**.

Matching lower bounds derive from TP.

Thanks!

Publications

[1] L. Bozzelli, A. Molinari, A. Montanari, and A. Peron.
An in-depth investigation of interval temporal logic model checking with regular expressions.
In *SEFM*, pages 104–119, 2017.

[2] L. Bozzelli, A. Molinari, A. Montanari, A. Peron, and P. Sala.
Interval Temporal Logic Model Checking: the Border Between Good and Bad HS Fragments.
In *IJCAR*, pages 389–405, 2016.

[3] L. Bozzelli, A. Molinari, A. Montanari, A. Peron, and P. Sala.
Interval vs. Point Temporal Logic Model Checking: an Expressiveness Comparison.
In *FSTTCS*, 2016.

[4] A. Molinari, A. Montanari, A. Murano, G. Perelli, and A. Peron.
Checking interval properties of computations.
Acta Informatica, pages 587–619, 2016.

HS (state-based) semantics and MC

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

- $\mathcal{K}, \rho \models p$ iff p labels all states of $\mathcal{K}$ composing ρ, for any $p \in \mathcal{AP}$ (homogeneity assumption);

HS (state-based) semantics and MC

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

- $\mathcal{K}, \rho \models r$ iff the labeling of ρ is in $\mathscr{L}(r)$ (labeling based on regular expressions);

HS (state-based) semantics and MC

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

- $\mathcal{K}, \rho \models r$ iff the labeling of ρ is in $\mathscr{L}(r)$
  (labeling based on regular expressions);
- negation, disjunction, and conjunction are standard;
- $\mathcal{K}, \rho \models \langle \mathsf{A} \rangle\, \psi \ldots;$
- $\mathcal{K}, \rho \models \langle \mathsf{B} \rangle\, \psi \ldots;$
- $\mathcal{K}, \rho \models \langle \mathsf{E} \rangle\, \psi \ldots;$
- inverse operators $\langle \overline{\mathsf{A}} \rangle$, $\langle \overline{\mathsf{B}} \rangle$, $\langle \overline{\mathsf{E}} \rangle$

HS (state-based) semantics and MC

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

- $\mathcal{K}, \rho \models r$ iff the labeling of ρ is in $\mathscr{L}(r)$ (labeling based on regular expressions);
- negation, disjunction, and conjunction are standard;
- $\mathcal{K}, \rho \models \langle \mathsf{A} \rangle \, \psi \ldots;$
- $\mathcal{K}, \rho \models \langle \mathsf{B} \rangle \, \psi \ldots;$
- $\mathcal{K}, \rho \models \langle \mathsf{E} \rangle \, \psi \ldots;$
- inverse operators $\langle \overline{\mathsf{A}} \rangle$, $\langle \overline{\mathsf{B}} \rangle$, $\langle \overline{\mathsf{E}} \rangle$

MC

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

Possibly infinitely many traces!

HS (state-based) semantics and MC

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

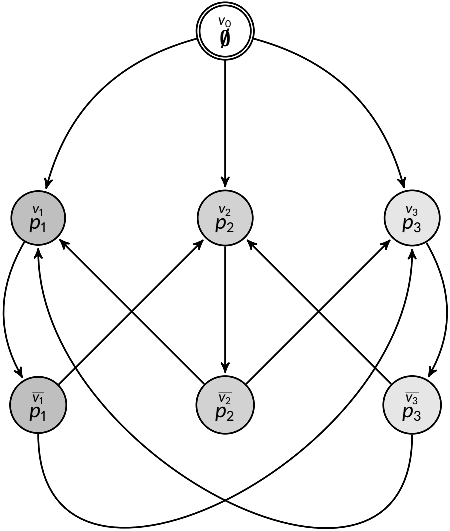
- $\mathcal{K}, \rho \models r$ iff the labeling of ρ is in $\mathcal{L}(r)$ (labeling based on regular expressions);
- negation, disjunction, and conjunction are standard;
- $\mathcal{K}, \rho \models \langle \mathsf{A} \rangle \psi \ldots;$
- $\mathcal{K}, \rho \models \langle \mathsf{B} \rangle \psi \ldots;$
- $\mathcal{K}, \rho \models \langle \mathsf{E} \rangle \psi \ldots;$
- inverse operators $\langle \overline{\mathsf{A}} \rangle$, $\langle \overline{\mathsf{B}} \rangle$, $\langle \overline{\mathsf{E}} \rangle$

MC

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

Possibly infinitely many traces!

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
(easily generalizable 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 successive 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 properties to be checked over $\mathcal{K}_{Sched}$

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

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

- 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)

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

- 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)

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

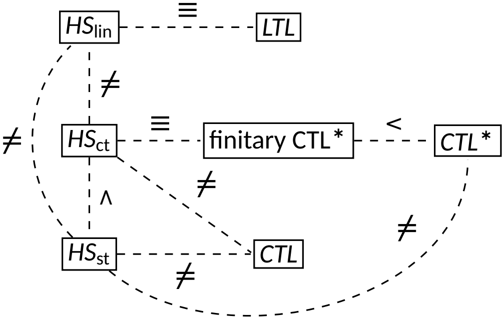
Complexity results

	Homogeneity
Full *HS*, BE	non-elementary **EXPSPACE**-hard
$A\overline{A}B\overline{B}\overline{E}$, $A\overline{A}E\overline{B}\overline{E}$	$\in$ **AEXP**$_{\text{Pol}}$ **PSPACE**-hard
$A\overline{A}\overline{B}\overline{E}$	**PSPACE**-complete
$A\overline{A}B\overline{B}$, $B\overline{B}$, $\overline{B}$, $A\overline{A}E\overline{E}$, $E\overline{E}$, $\overline{E}$	**PSPACE**-complete
$A\overline{A}B$, $A\overline{A}E$, AB, $\overline{A}E$	$\mathbf{P}^{\mathbf{NP}}$-complete
$A\overline{A}$, $\overline{A}B$, AE, A, $\overline{A}$	$\in \mathbf{P}^{\mathbf{NP}[O(\log^2 n)]}$ $\mathbf{P}^{\mathbf{NP}[O(\log n)]}$-hard
Prop, B, E	**co-NP**-complete

Complexity results

	Homogeneity	Regular expressions
Full *HS*, BE	non-elementary **EXPSPACE**-hard	non-elementary **EXPSPACE**-hard
$A\overline{A}B\overline{B}\overline{E}$, $A\overline{A}E\overline{B}\overline{E}$	$\in \mathbf{AEXP}_{\mathrm{Pol}}$ **PSPACE**-hard	$\mathbf{AEXP}_{\mathrm{Pol}}$-complete
$A\overline{A}\overline{B}\overline{E}$	**PSPACE**-complete	$\in \mathbf{AEXP}_{\mathrm{Pol}}$ **PSPACE**-hard
$A\overline{A}B\overline{B}$, $B\overline{B}$, $\overline{B}$, $A\overline{A}E\overline{E}$, $E\overline{E}$, $\overline{E}$	**PSPACE**-complete	**PSPACE**-complete
$A\overline{A}B$, $A\overline{A}E$, AB, $\overline{A}E$	$\mathbf{P}^{\mathbf{NP}}$-complete	**PSPACE**-complete
$A\overline{A}$, $\overline{A}B$, AE, A, $\overline{A}$	$\in \mathbf{P}^{\mathbf{NP}[O(\log^2 n)]}$ $\mathbf{P}^{\mathbf{NP}[O(\log n)]}$-hard	**PSPACE**-complete
Prop, B, E	**co-NP**-complete	**PSPACE**-complete

Expressiveness results (under homogeneity)



Reference

L. Bozzelli, A. Molinari, A. Montanari, A. Peron, and P. Sala. Interval vs. Point Temporal Logic Model Checking: an Expressiveness Comparison. In *FSTTCS*, 2016

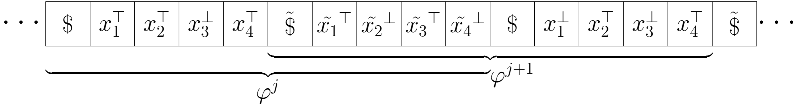
Sketch of **PSPACE**-hardness

- Reduction from the **PSPACE**-complete problem Periodic SAT
- We are given a Boolean formula $\varphi(x_1, \ldots, x_n, x_1^{+1}, \ldots, x_n^{+1})$ in CNF
- φ^j is φ in which we replace each x_i by a fresh x_i^j, and x_i^{+1} by x_i^{j+1}.
- Decide the satisfiability of the infinite-length formula

$$\Phi = \bigwedge_{j \geq 1} \varphi^j$$

(actually equivalent to $\Phi_f = \bigwedge_{j=1}^{2^{2n}+1} \varphi^j$).

Sketch of **PSPACE**-hardness



For the *t*-th conjunct of φ,

$$
\begin{aligned}
o[y = \tilde{\$}] \to \Big(\bigvee_{x_i \in \Gamma \cap L_t^+} \exists o'[y = \tilde{x}_i^{\top}].o \leq^{e,s}_{[0,4n]} o' \Big) \vee \\
\Big(\bigvee_{x_i^{+1} \in \Gamma^{+1} \cap L_t^+} \exists o'[y = x_i^{\top}].o \leq^{e,s}_{[0,4n]} o' \Big) \vee \\
\Big(\bigvee_{x_i \in \Gamma \cap L_t^-} \exists o'[y = \tilde{x}_i^{\bot}].o \leq^{e,s}_{[0,4n]} o' \Big) \vee \\
\Big(\bigvee_{x_i^{+1} \in \Gamma^{+1} \cap L_t^-} \exists o'[y = x_i^{\bot}].o \leq^{e,s}_{[0,4n]} o' \Big) \vee \\
\exists o''[y = stop].o \leq^{e,s}_{[0,2n]} o''.
\end{aligned}
$$

Timelines with trigger-less rules only

- Trigger-less synchronization rules can be directly translated into a timed automaton (no need to translate into MTL)
- Timed automaton of exponential size: it gives us an exponential bound (*) on the number of tokens and on the horizon

Timelines with trigger-less rules only

- Trigger-less synchronization rules can be directly translated into a timed automaton (no need to translate into MTL)
- Timed automaton of exponential size: it gives us an exponential bound (*) on the number of tokens and on the horizon
- We observe that:
  1. timelines for different variables evolve independently, and
  2. each trigger-less rule enforces at most one "synchronization point" among timelines.

Theorem

TP with trigger-less rules only is **NP***-complete.*

Timelines with trigger-less rules only

- Trigger-less synchronization rules can be directly translated into a timed automaton (no need to translate into MTL)
- Timed automaton of exponential size: it gives us an exponential bound (*) on the number of tokens and on the horizon
- We observe that:
  1. timelines for different variables evolve independently, and
  2. each trigger-less rule enforces at most one “synchronization point” among timelines.

Theorem

TP with trigger-less rules only is **NP***-complete.*

How we deal with 1. and 2.:

1. timeline evolutions are enforced by a linear program (where constants are exponential (*)), resting on results on Eulerian multi-graphs (thanks G. Woeginger!)
2. we non-deterministically position tokens (those to which rules refer) along timelines (their start/end times can be generated in polynomial time (*)) and check satisfaction of rules

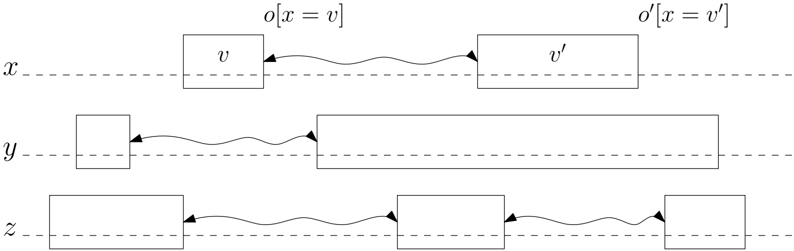
NP-completeness of the trigger-less case

- Timed automata give us (i) an exponential bound on the number of tokens of any plan and (ii) an exponential bound on the horizon.
- We start by reducing to integers all the rational values occurring in the instance.
- For every quantifier $o_i[x_i = v_i]$ in the rules, the algorithm guesses
  1. the integer part of the start and end time of the token for x_i to which o_i is mapped,
  2. an order of all fractional parts of such start/end times.

If we change the start/end time of (some of the) tokens associated with quantifiers, but we leave unchanged (i) all the integer parts, (ii) zeroness/non-zeroness of fractional parts, and (iii) the fractional parts' order, satisfaction of atoms in the rules does not change.

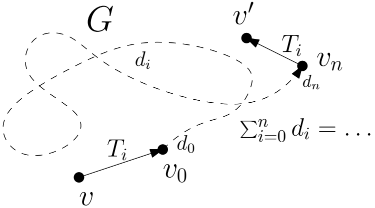
NP-completeness of the trigger-less case

- Now we have to check that there exists a legal timeline evolution “connecting” each pair of adjacent guessed tokens over the same variable



NP-completeness of the trigger-less case

- We interpret each state variable $x_i = (V_i, T_i, D_i)$ as a directed graph $G = (V_i, T_i)$ where D_i associates each $v \in V_i$ with a duration interval.
- For a pair of adjacent guessed tokens (x_i, v, d) and (x_i, v', d'):



- To this aim we guess a set of integers $\{\alpha_{u,v} \mid (u, v) \in T_i\}$ where $\alpha_{u,v}$ is the number of times the path traverses (u, v) and check that they specify a directed Eulerian path (in a multi-graph) $v_0 \rightsquigarrow v_n$.
- To check all this, we solve a linear problem. *(thanks G. Woeginger!)*

Theorem

TP with trigger-less rules is ***NP****-complete.*

NP-hardness: from existence of a Hamiltonian path in directed graph.