Real-Time Higher-Order Recursion Schemes

Eric Alsmann[1] and Florian Bruse[1,2]

[1] University of Kassel, Germany
[2] Technical University of Munich, Germany
TUM School of Computation, Information and Technology

31st Int. Symp. on Temporal Representation and Reasoning, TIME'24
28/10 - 30/10/2024
Montpellier, France

Higher-Order Recursion Schemes

Higher-Order Recursion Scheme (HORS): higher-order grammar that generates trees:

$$
\begin{aligned}
S &\mapsto A\ c \\
A\ x &\mapsto a\ x\ (A\ (b\ x))
\end{aligned}
$$

Higher-Order Recursion Schemes

Higher-Order Recursion Scheme (HORS): higher-order grammar that generates trees:

$$S \mapsto A\ c$$
$$A\ x \mapsto a\ x\ (A\ (b\ x))$$

more complex expressions generated by

- expansion of grammar nonterminals
- β-reduction

Higher-Order Recursion Schemes

Higher-Order Recursion Scheme (HORS): higher-order grammar that generates trees:

$$S \mapsto A\ c$$
$$A\ x \mapsto a\ x\ (A\ (b\ x))$$

$A\ c$

more complex expressions generated by

- expansion of grammar nonterminals
- β-reduction

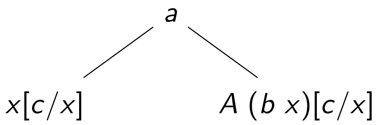
limit of expressions exists, defines tree

$$S \Rightarrow A\ c$$

Higher-Order Recursion Schemes

Higher-Order Recursion Scheme (HORS): higher-order grammar that generates trees:

$$S \mapsto A\ c$$
$$A\ x \mapsto a\ x\ (A\ (b\ x))$$



more complex expressions generated by

- expansion of grammar nonterminals
- β-reduction

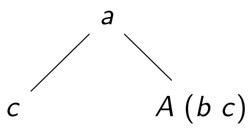
limit of expressions exists, defines tree

$$S \Rightarrow A\ c$$
$$\Rightarrow (a\ x\ (A\ b\ x))[c/x]$$

Higher-Order Recursion Schemes

Higher-Order Recursion Scheme (HORS): higher-order grammar that generates trees:

$$S \mapsto A\ c$$
$$A\ x \mapsto a\ x\ (A\ (b\ x))$$



more complex expressions generated by

- expansion of grammar nonterminals
- β-reduction

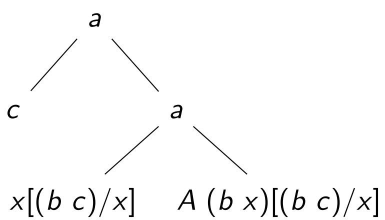
limit of expressions exists, defines tree

$$\begin{aligned} S &\Rightarrow A\ c \\ &\Rightarrow (a\ x\ (A\ b\ x))[c/x] \\ &= a\ c\ (A\ (b\ c)) \end{aligned}$$

Higher-Order Recursion Schemes

Higher-Order Recursion Scheme (HORS): higher-order grammar that generates trees:

$$S \mapsto A\ c$$
$$A\ x \mapsto a\ x\ (A\ (b\ x))$$



more complex expressions generated by

- expansion of grammar nonterminals
- β-reduction

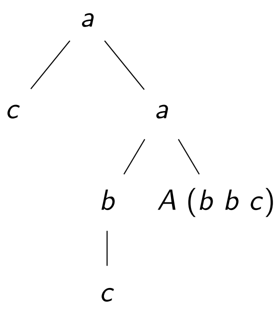
limit of expressions exists, defines tree

$$\begin{aligned} S &\Rightarrow A\ c \\ &\Rightarrow (a\ x\ (A\ b\ x))[c/x] \\ &= a\ c\ (A\ (b\ c)) \\ &\Rightarrow a\ c\ (a\ x\ (A\ (b\ x)))[(b\ c/x] \end{aligned}$$

Higher-Order Recursion Schemes

Higher-Order Recursion Scheme (HORS): higher-order grammar that generates trees:

$$S \mapsto A\ c$$
$$A\ x \mapsto a\ x\ (A\ (b\ x))$$



more complex expressions generated by

- expansion of grammar nonterminals
- β-reduction

limit of expressions exists, defines tree

$$\begin{aligned} S &\Rightarrow A\ c \\ &\Rightarrow (a\ x\ (A\ b\ x))[c/x] \\ &= a\ c\ (A\ (b\ c)) \\ &\Rightarrow a\ c\ (a\ x\ (A\ (b\ x)))[(b\ c/x] \\ &= a\ c\ (a\ (b\ c)\ (A\ (b\ b\ c))) \\ &\quad \ldots \end{aligned}$$

Higher-Order Recursion Schemes

Higher-Order Recursion Scheme (HORS): higher-order grammar that generates trees:

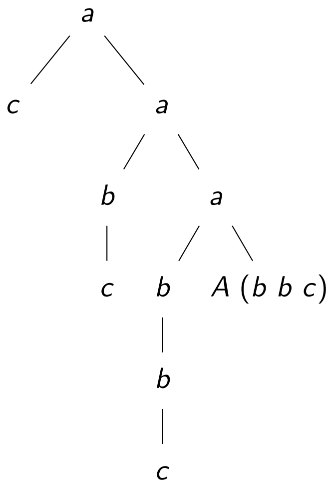
$$S \mapsto A\ c$$
$$A\ x \mapsto a\ x\ (A\ (b\ x))$$

more complex expressions generated by

- expansion of grammar nonterminals
- β-reduction

limit of expressions exists, defines tree

$$\begin{aligned} S &\Rightarrow A\ c \\ &\Rightarrow (a\ x\ (A\ b\ x))[c/x] \\ &= a\ c\ (A\ (b\ c)) \\ &\Rightarrow a\ c\ (a\ x\ (A\ (b\ x)))[(b\ c/x] \\ &= a\ c\ (a\ (b\ c)\ (A\ (b\ b\ c))) \\ &\quad \ldots \end{aligned}$$



Model-Checking for HORS

parity tree automata (APT):

- acceptance explained as two-player game between verifier and spoiler
- play of game defines labeling of path in tree by states of APT

Model-Checking for HORS

parity tree automata (APT):

- acceptance explained as two-player game between verifier and spoiler
- play of game defines labeling of path in tree by states of APT
- infinite plays decided by parity condition

Model-Checking for HORS

parity tree automata (APT):

- acceptance explained as two-player game between verifier and spoiler
- play of game defines labeling of path in tree by states of APT
- infinite plays decided by parity condition

model-checking problem for HORS: given a HORS $\mathcal{G}$ and an APT $\mathcal{A}$, does $\mathcal{A}$ accept the tree generated by $\mathcal{G}$?

Model-Checking for HORS

parity tree automata (APT):

- acceptance explained as two-player game between verifier and spoiler
- play of game defines labeling of path in tree by states of APT
- infinite plays decided by parity condition

model-checking problem for HORS: given a HORS $\mathcal{G}$ and an APT $\mathcal{A}$, does $\mathcal{A}$ accept the tree generated by $\mathcal{G}$?

Prop.: (Ong'06) For order-k HORS, the HORS model-checking problem is k-EXPTIME-complete.

order of a HORS: order of highest type of argument, e.g.,

- order 1: set of trees,
- order 2: set of trees $\rightarrow$ set of trees,
- ...

The Challenge of Dense Linear Time and Higher-Order

trees generated by HORS yield only qualitative, discrete semantics

no possibility to make fine-grained statements on e.g.,

- the time after which a certain action happens,

The Challenge of Dense Linear Time and Higher-Order

trees generated by HORS yield only qualitative, discrete semantics

no possibility to make fine-grained statements on e.g.,

- the time after which a certain action happens,
- bounded time-frames after which a request must be answered,

The Challenge of Dense Linear Time and Higher-Order

trees generated by HORS yield only qualitative, discrete semantics

no possibility to make fine-grained statements on e.g.,

- the time after which a certain action happens,
- bounded time-frames after which a request must be answered,
- the aggregate length of a sequence of individual actions,
- etc.

The Challenge of Dense Linear Time and Higher-Order

trees generated by HORS yield only qualitative, discrete semantics

no possibility to make fine-grained statements on e.g.,

- the time after which a certain action happens,
- bounded time-frames after which a request must be answered,
- the aggregate length of a sequence of individual actions,
- etc.

potential problems:

- theory of HORS is quite complex

The Challenge of Dense Linear Time and Higher-Order

trees generated by HORS yield only qualitative, discrete semantics

no possibility to make fine-grained statements on e.g.,

- the time after which a certain action happens,
- bounded time-frames after which a request must be answered,
- the aggregate length of a sequence of individual actions,
- etc.

potential problems:

- theory of HORS is quite complex
- HORS are genuinely non-regular $\rightsquigarrow$ must avoid adding non-regularity of verification device for decidability reasons

The Challenge of Dense Linear Time and Higher-Order

trees generated by HORS yield only qualitative, discrete semantics

no possibility to make fine-grained statements on e.g.,

- the time after which a certain action happens,
- bounded time-frames after which a request must be answered,
- the aggregate length of a sequence of individual actions,
- etc.

potential problems:

- theory of HORS is quite complex
- HORS are genuinely non-regular $\rightsquigarrow$ must avoid adding non-regularity of verification device for decidability reasons

timed automata are well-established, rather simple, and regular in character $\rightsquigarrow$ good fit

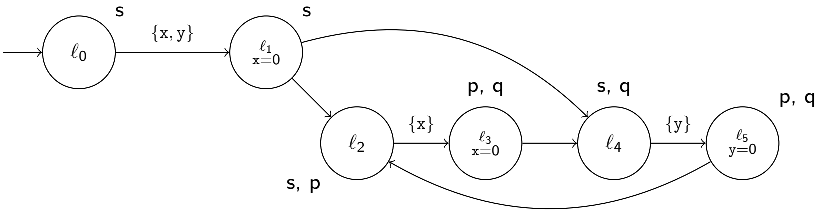
Timed Automata

timed automaton introduced as model of real-time systems

Timed Automata

timed automaton introduced as model of real-time systems

example:



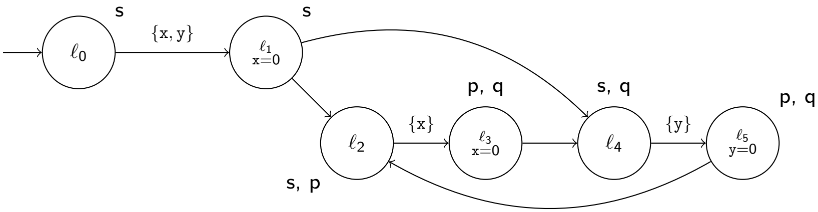
semantics explained via clock evaluations (mappings from clocks $\mathtt{x}, \mathtt{y}, \ldots$ to $\mathbb{R}^{\geq 0}$)

position in TA is pair of location and clock evaluation, e.g., $(\ell_0, (\mathtt{x} \mapsto 1, \mathtt{y} \mapsto 0.3423))$

Timed Automata

timed automaton introduced as model of real-time systems

example:



semantics explained via clock evaluations (mappings from clocks $\mathtt{x}, \mathtt{y}, \ldots$ to $\mathbb{R}^{\geq 0}$)

position in TA is pair of location and clock evaluation, e.g., $(\ell_0, (\mathtt{x} \mapsto 1, \mathtt{y} \mapsto 0.3423))$

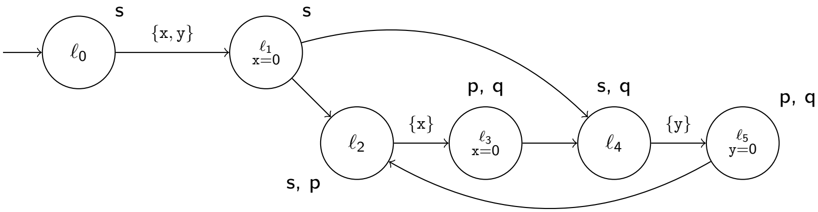
flow of time (by amount t controlled by verifier, she can:

- move alongside transitions,

Timed Automata

timed automaton introduced as model of real-time systems

example:



semantics explained via clock evaluations (mappings from clocks $x, y, \dots$ to $\mathbb{R}^{\geq 0}$)

position in TA is pair of location and clock evaluation, e.g., $(\ell_0, (x \mapsto 1, y \mapsto 0.3423))$

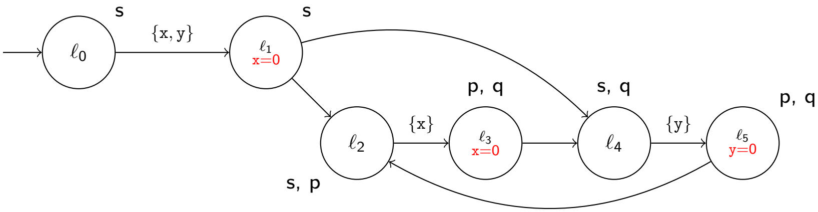
flow of time (by amount t controlled by verifier, she can:

- move alongside transitions,
- let time flow (increases value of all clocks simultaneously), or
- do arbitrary combinations

Timed Automata

timed automaton introduced as model of real-time systems

example:



semantics explained via clock evaluations (mappings from clocks $\mathtt{x}, \mathtt{y}, \ldots$ to $\mathbb{R}^{\geq 0}$)

position in TA is pair of location and clock evaluation, e.g., $(\ell_0, (\mathtt{x} \mapsto 1, \mathtt{y} \mapsto 0.3423))$

flow of time (by amount t controlled by verifier, she can:

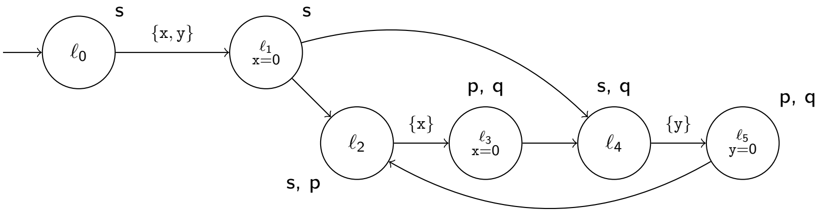
- move alongside transitions,
- let time flow (increases value of all clocks simultaneously), or
- do arbitrary combinations

flow of time must respect location invariants, guards, and resets (and t)

Timed Automata

timed automaton introduced as model of real-time systems

example:



semantics explained via clock evaluations (mappings from clocks $\mathtt{x}, \mathtt{y}, \ldots$ to $\mathbb{R}^{\geq 0}$)

position in TA is pair of location and clock evaluation, e.g., $(\ell_0, (\mathtt{x} \mapsto 1, \mathtt{y} \mapsto 0.3423))$

flow of time (by amount t controlled by verifier, she can:

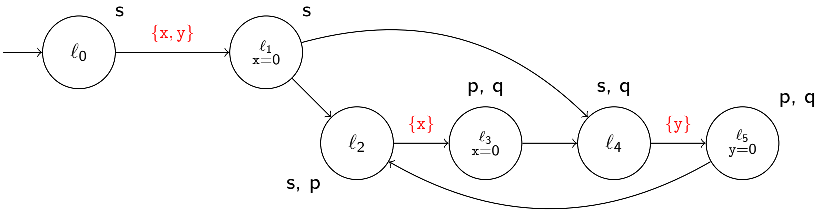
- move alongside transitions,
- let time flow (increases value of all clocks simultaneously), or
- do arbitrary combinations

flow of time must respect location invariants, guards, and resets (and t)

Timed Automata

timed automaton introduced as model of real-time systems

example:



semantics explained via clock evaluations (mappings from clocks $x, y, \ldots$ to $\mathbb{R}^{\geq 0}$)

position in TA is pair of location and clock evaluation, e.g., $(\ell_0, (x \mapsto 1, y \mapsto 0.3423))$

flow of time (by amount t controlled by verifier, she can:

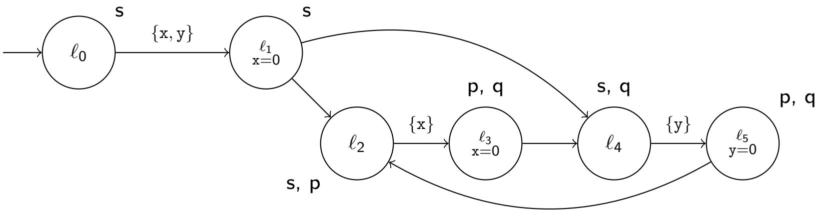
- move alongside transitions,
- let time flow (increases value of all clocks simultaneously), or
- do arbitrary combinations

flow of time must respect location invariants, guards, and resets (and t)

Timed Automata

timed automaton introduced as model of real-time systems

example:



semantics explained via clock evaluations (mappings from clocks $\mathrm{x}, \mathrm{y}, \ldots$ to $\mathbb{R}^{\geq 0}$)

position in TA is pair of location and clock evaluation, e.g., $(\ell_0, (\mathrm{x} \mapsto 1, \mathrm{y} \mapsto 0.3423))$

flow of time (by amount t controlled by verifier, she can:

- move alongside transitions,
- let time flow (increases value of all clocks simultaneously), or
- do arbitrary combinations

flow of time must respect location invariants, guards, and resets (and t)

The Plan

goal: incorporate real-time semantics onto HORS using timed automata ...

... without giving up established theory of HORS model-checking

The Plan

goal: incorporate real-time semantics onto HORS using timed automata . . .

. . . without giving up established theory of HORS model-checking

proposal:

- annotate tree constructors by intervals (intuition: duration of an action)

The Plan

goal: incorporate real-time semantics onto HORS using timed automata ...

... without giving up established theory of HORS model-checking

proposal:

- annotate tree constructors by intervals (intuition: duration of an action)
- add timed automata as extra verification device alongside APT

The Plan

goal: incorporate real-time semantics onto HORS using timed automata . . .

. . . without giving up established theory of HORS model-checking

proposal:

- annotate tree constructors by intervals (intuition: duration of an action)
- add timed automata as extra verification device alongside APT

main challenges:

- timed automata invented as models of systems, not as verification device
- need meaningful interaction between timed automaton and APT for useful specifications

Construction Details

timed HORS:

- tree constructors annotated by intervals
- only finitely many intervals appear due to finiteness of grammar

Construction Details

timed HORS:

- tree constructors annotated by intervals
- only finitely many intervals appear due to finiteness of grammar

timed automaton:

- standard TA model, but semantics controlled by verifier

Construction Details

timed HORS:

- tree constructors annotated by intervals
- only finitely many intervals appear due to finiteness of grammar

timed automaton:

- standard TA model, but semantics controlled by verifier
- propositions same as state set of APT
- when time flows, only locations enabled that are labeled by current APT state

Construction Details

timed HORS:

- tree constructors annotated by intervals
- only finitely many intervals appear due to finiteness of grammar

timed automaton:

- standard TA model, but semantics controlled by verifier
- propositions same as state set of APT
- when time flows, only locations enabled that are labeled by current APT state

timed APT:

- transition function depends on (untimed) tree label, current state, and clock evaluations of TA
- presentation finite due to only finitely many clock constraints allowed

Construction Details

timed HORS:

- tree constructors annotated by intervals
- only finitely many intervals appear due to finiteness of grammar

timed automaton:

- standard TA model, but semantics controlled by verifier
- propositions same as state set of APT
- when time flows, only locations enabled that are labeled by current APT state

timed APT:

- transition function depends on (untimed) tree label, current state, and clock evaluations of TA
- presentation finite due to only finitely many clock constraints allowed

general flow of acceptance game:

- verifier lets time flow according to current tree label
- then APT transitions

Example

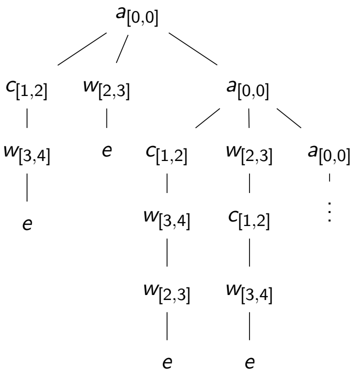
timed HORS:

$$
\begin{aligned}
S &\mapsto T\ (F\ e)\ (G\ e) \\
T\ x\ y &\mapsto a_{[0,0]}\ x\, y\, (T\ (F\ y)(G\ x)) \\
F\ z &\mapsto c_{[1,2]}\ w_{[3,4]}\ z \\
G\ t &\mapsto w_{[2,3]}\ z
\end{aligned}
$$

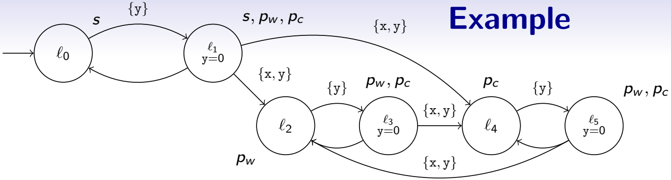
Example

timed HORS:

$$S \mapsto T\ (F\ e)\ (G\ e)$$
$$T\ x\ y \mapsto a_{[0,0]}\ x\, y\, (T\ (F\ y)(G\ x))$$
$$F\ z \mapsto c_{[1,2]}\ w_{[3,4]}\ z$$
$$G\ t \mapsto w_{[2,3]}\ z$$

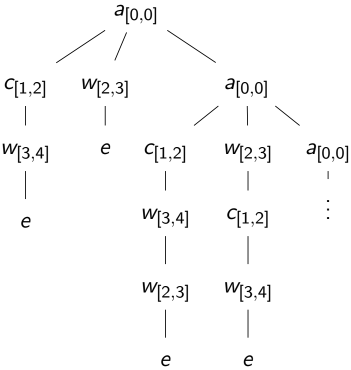


Example

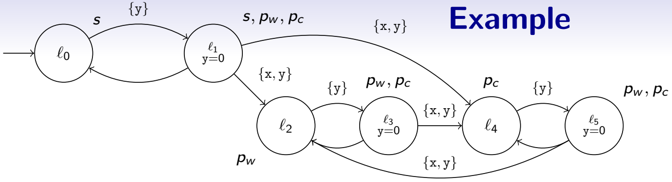


timed HORS:

$$S \mapsto T\ (F\ e)\ (G\ e)$$
$$T\ x\ y \mapsto a_{[0,0]}\ x\, y\, (T\ (F\ y)(G\ x))$$
$$F\ z \mapsto c_{[1,2]}\ w_{[3,4]}\ z$$
$$G\ t \mapsto w_{[2,3]}\ z$$



Example

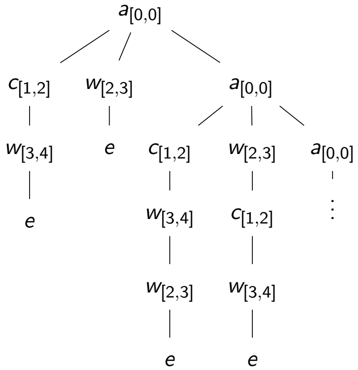


timed HORS:

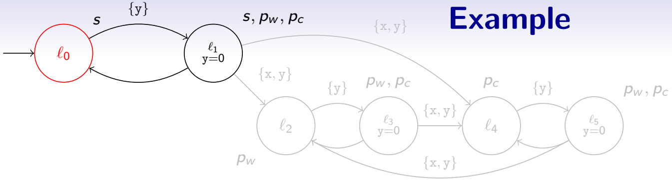
$$S \mapsto T\ (F\ e)\ (G\ e)$$
$$T\ x\ y \mapsto a_{[0,0]}\ x\, y\, (T\ (F\ y)(G\ x))$$
$$F\ z \mapsto c_{[1,2]}\ w_{[3,4]}\ z$$
$$G\ t \mapsto w_{[2,3]}\ z$$

APT:

$$\delta(s, \_, a) = (s, s, s)$$
$$\delta(\_, \mathtt{x} > 4, \_) = \bot \qquad \delta(\_, \mathtt{x} \leq 4, c) = (p_c)$$
$$\delta(\_, \mathtt{x} \leq 4, e) = \top \qquad \delta(\_, \mathtt{x} \leq 4, w) = (p_w)$$



Example

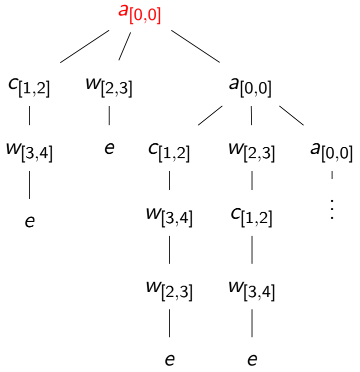


timed HORS:

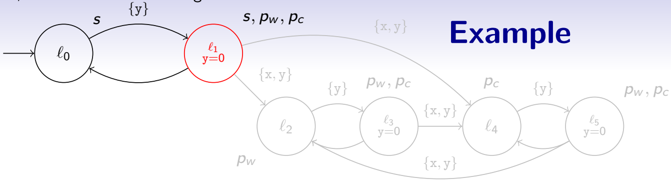
$$S \mapsto T\ (F\ e)\ (G\ e)$$
$$T\ x\ y \mapsto a_{[0,0]}\ x\ y\ (T\ (F\ y)(G\ x))$$
$$F\ z \mapsto c_{[1,2]}\ w_{[3,4]}\ z$$
$$G\ t \mapsto w_{[2,3]}\ z$$

APT:

$$\delta(s, \_, a) = (s, s, s)$$
$$\delta(\_, \mathtt{x} > 4, \_) = \bot \qquad \delta(\_, \mathtt{x} \leq 4, c) = (p_c)$$
$$\delta(\_, \mathtt{x} \leq 4, e) = \top \qquad \delta(\_, \mathtt{x} \leq 4, w) = (p_w)$$



APT state:	TA location:	clock values:	tree address:
s	ℓ_0	$\mathtt{x} \mapsto 0, \mathtt{y} \mapsto 0$	ϵ



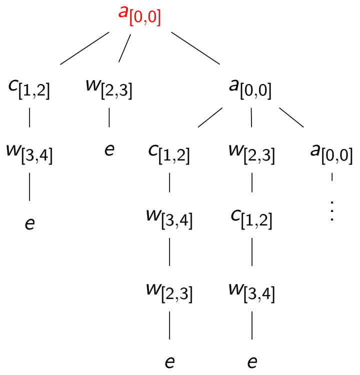
Example

timed HORS:

$$S \mapsto T\ (F\ e)\ (G\ e)$$
$$T\ x\ y \mapsto a_{[0,0]}\ x\ y\ (T\ (F\ y)(G\ x))$$
$$F\ z \mapsto c_{[1,2]}\ w_{[3,4]}\ z$$
$$G\ t \mapsto w_{[2,3]}\ z$$

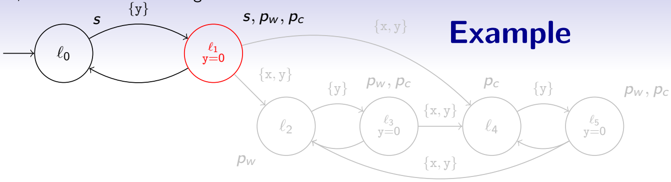
APT:

$$\delta(s, \_, a) = (s, s, s)$$
$$\delta(\_, \mathrm{x} > 4, \_) = \bot \qquad \delta(\_, \mathrm{x} \leq 4, c) = (p_c)$$
$$\delta(\_, \mathrm{x} \leq 4, e) = \top \qquad \delta(\_, \mathrm{x} \leq 4, w) = (p_w)$$



APT state:	TA location:	clock values:	tree address:
s	ℓ_0	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	ϵ
s	ℓ_1	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	ϵ

Example



timed HORS:

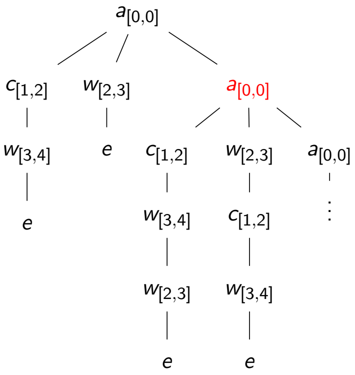
$$S \mapsto T\ (F\ e)\ (G\ e)$$
$$T\ x\ y \mapsto a_{[0,0]}\ x\, y\, (T\ (F\ y)(G\ x))$$
$$F\ z \mapsto c_{[1,2]}\ w_{[3,4]}\ z$$
$$G\ t \mapsto w_{[2,3]}\ z$$

APT:

$$\delta(s, \_, a) = (s, s, s)$$

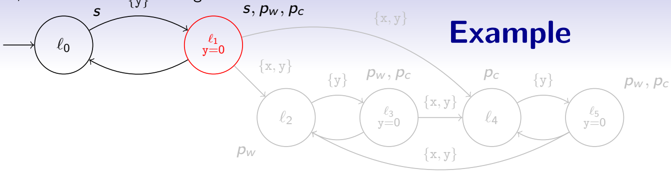
$$\delta(\_, \mathrm{x} > 4, \_) = \bot \qquad \delta(\_, \mathrm{x} \le 4, c) = (p_c)$$

$$\delta(\_, \mathrm{x} \le 4, e) = \top \qquad \delta(\_, \mathrm{x} \le 4, w) = (p_w)$$



APT state:	TA location:	clock values:	tree address:
s	ℓ_0	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	ϵ
s	ℓ_1	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	ϵ
s	ℓ_1	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	2

Example



timed HORS:

$$S \mapsto T\ (F\ e)\ (G\ e)$$
$$T\ x\ y \mapsto a_{[0,0]}\ x\, y\, (T\ (F\ y)(G\ x))$$
$$F\ z \mapsto c_{[1,2]}\ w_{[3,4]}\ z$$
$$G\ t \mapsto w_{[2,3]}\ z$$

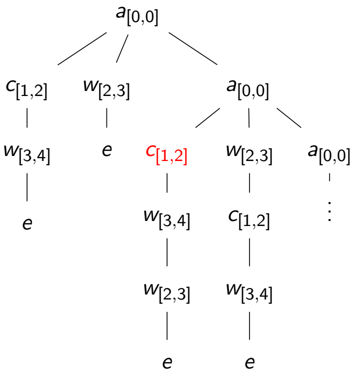
APT:

$$\delta(s, \_, a) = (s, s, s)$$

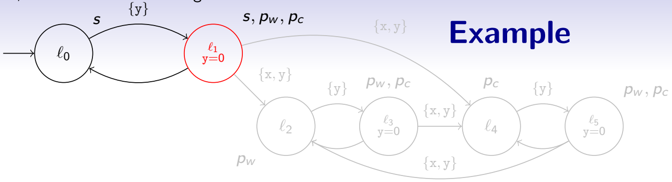
$$\delta(\_, \mathrm{x} > 4, \_) = \bot \qquad \delta(\_, \mathrm{x} \leq 4, c) = (p_c)$$

$$\delta(\_, \mathrm{x} \leq 4, e) = \top \qquad \delta(\_, \mathrm{x} \leq 4, w) = (p_w)$$

APT state:	TA location:	clock values:	tree address:
s	ℓ_0	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	ϵ
s	ℓ_1	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	ϵ
s	ℓ_1	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	2
s	ℓ_1	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	$2 \cdot 0$



Example

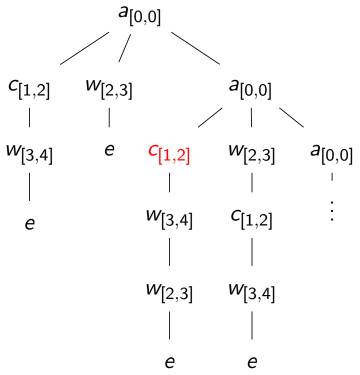


timed HORS:

$$S \mapsto T\ (F\ e)\ (G\ e)$$
$$T\ x\ y \mapsto a_{[0,0]}\ x\ y\ (T\ (F\ y)(G\ x))$$
$$F\ z \mapsto c_{[1,2]}\ w_{[3,4]}\ z$$
$$G\ t \mapsto w_{[2,3]}\ z$$

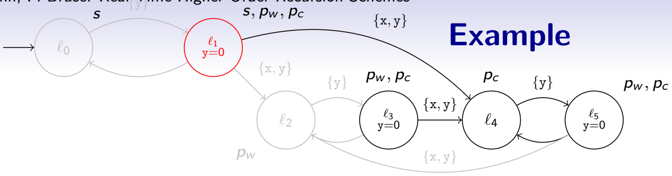
APT:

$$\delta(s, \_, a) = (s, s, s)$$
$$\delta(\_, \mathrm{x} > 4, \_) = \bot \qquad \delta(\_, \mathrm{x} \leq 4, c) = (p_c)$$
$$\delta(\_, \mathrm{x} \leq 4, e) = \top \qquad \delta(\_, \mathrm{x} \leq 4, w) = (p_w)$$



APT state:	TA location:	clock values:	tree address:
s	ℓ_0	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	ϵ
s	ℓ_1	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	2
s	ℓ_1	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	$2 \cdot 0$
s	ℓ_1	$\mathrm{x} \mapsto 1.4, \mathrm{y} \mapsto 0$	$2 \cdot 0$

Example

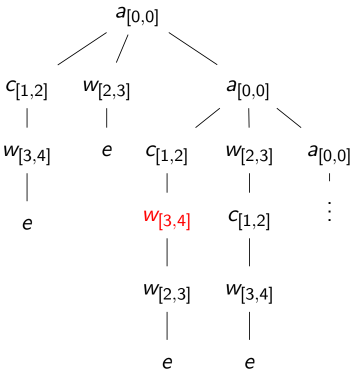


timed HORS:

$$S \mapsto T\ (F\ e)\ (G\ e)$$
$$T\ x\ y \mapsto a_{[0,0]}\ x\, y\, (T\ (F\ y)(G\ x))$$
$$F\ z \mapsto c_{[1,2]}\ w_{[3,4]}\ z$$
$$G\ t \mapsto w_{[2,3]}\ z$$

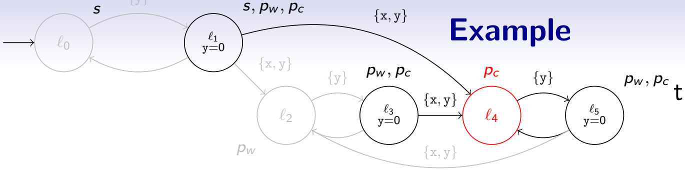
APT:

$$\delta(s, \_, a) = (s, s, s)$$
$$\delta(\_, \mathrm{x} > 4, \_) = \bot \qquad \delta(\_, \mathrm{x} \leq 4, c) = (p_c)$$
$$\delta(\_, \mathrm{x} \leq 4, e) = \top \qquad \delta(\_, \mathrm{x} \leq 4, w) = (p_w)$$



APT state:	TA location:	clock values:	tree address:
s	ℓ_0	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	ϵ
s	ℓ_1	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	2
s	ℓ_1	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	$2 \cdot 0$
s	ℓ_1	$\mathrm{x} \mapsto 1.4, \mathrm{y} \mapsto 0$	$2 \cdot 0$
p_c	ℓ_1	$\mathrm{x} \mapsto 1.4, \mathrm{y} \mapsto 0$	$2 \cdot 0 \cdot 0$

Example



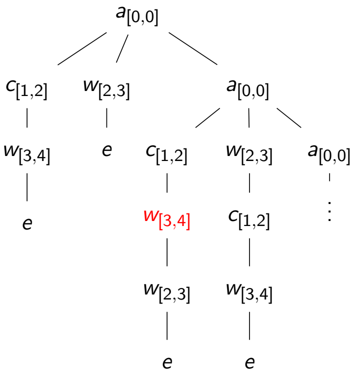
timed HORS:

$$S \mapsto T\ (F\ e)\ (G\ e)$$
$$T\ x\ y \mapsto a_{[0,0]}\ x\, y\, (T\ (F\ y)(G\ x))$$
$$F\ z \mapsto c_{[1,2]}\ w_{[3,4]}\ z$$
$$G\ t \mapsto w_{[2,3]}\ z$$

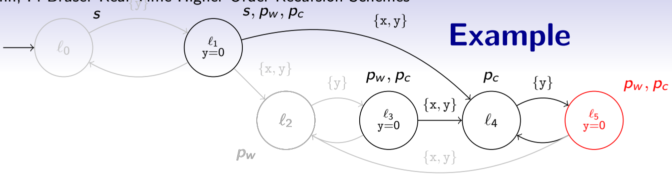
APT:

$$\delta(s, \_, a) = (s, s, s)$$
$$\delta(\_, \mathrm{x} > 4, \_) = \bot \qquad \delta(\_, \mathrm{x} \leq 4, c) = (p_c)$$
$$\delta(\_, \mathrm{x} \leq 4, e) = \top \qquad \delta(\_, \mathrm{x} \leq 4, w) = (p_w)$$

APT state:	TA location:	clock values:	tree address:
s	ℓ_0	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	ϵ
s	ℓ_1	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	2
s	ℓ_1	$\mathrm{x} \mapsto 1.4, \mathrm{y} \mapsto 0$	$2 \cdot 0$
p_c	ℓ_1	$\mathrm{x} \mapsto 1.4, \mathrm{y} \mapsto 0$	$2 \cdot 0 \cdot 0$
p_c	ℓ_4	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	$2 \cdot 0 \cdot 0$



Example



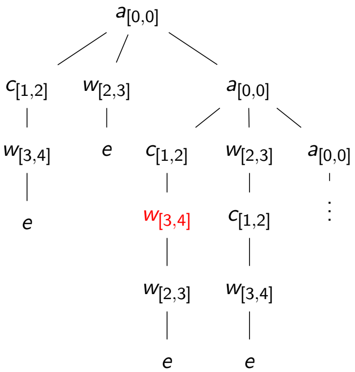
timed HORS:

$$S \mapsto T\ (F\ e)\ (G\ e)$$
$$T\ x\ y \mapsto a_{[0,0]}\ x\, y\, (T\ (F\ y)(G\ x))$$
$$F\ z \mapsto c_{[1,2]}\ w_{[3,4]}\ z$$
$$G\ t \mapsto w_{[2,3]}\ z$$

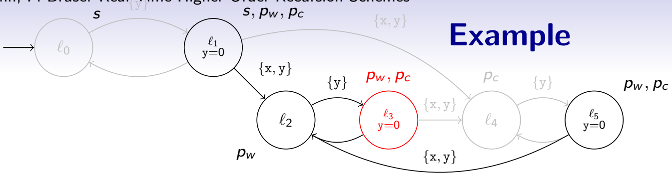
APT:

$$\delta(s, \_, a) = (s, s, s)$$
$$\delta(\_, \mathrm{x} > 4, \_) = \bot \qquad \delta(\_, \mathrm{x} \leq 4, c) = (p_c)$$
$$\delta(\_, \mathrm{x} \leq 4, e) = \top \qquad \delta(\_, \mathrm{x} \leq 4, w) = (p_w)$$

APT state:	TA location:	clock values:	tree address:
s	ℓ_0	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	ϵ
s	ℓ_1	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	2
s	ℓ_1	$\mathrm{x} \mapsto 1.4, \mathrm{y} \mapsto 0$	$2 \cdot 0$
p_c	ℓ_1	$\mathrm{x} \mapsto 1.4, \mathrm{y} \mapsto 0$	$2 \cdot 0 \cdot 0$
p_c	ℓ_4	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	$2 \cdot 0 \cdot 0$
p_c	ℓ_5	$\mathrm{x} \mapsto 3.2, \mathrm{y} \mapsto 0$	$2 \cdot 0 \cdot 0$



Example



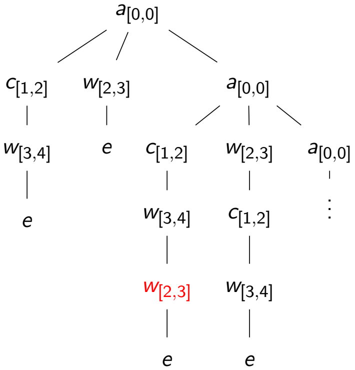
timed HORS:

$$S \mapsto T\ (F\ e)\ (G\ e)$$
$$T\ x\ y \mapsto a_{[0,0]}\ x\, y\, (T\ (F\ y)(G\ x))$$
$$F\ z \mapsto c_{[1,2]}\ w_{[3,4]}\ z$$
$$G\ t \mapsto w_{[2,3]}\ z$$

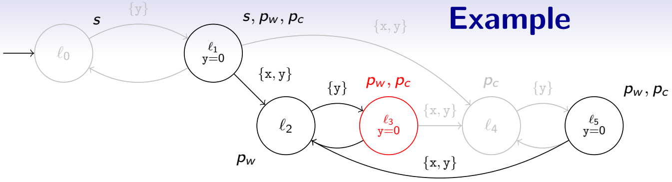
APT:

$$\delta(s, \_, a) = (s, s, s)$$
$$\delta(\_, \mathrm{x} > 4, \_) = \bot \qquad \delta(\_, \mathrm{x} \leq 4, c) = (p_c)$$
$$\delta(\_, \mathrm{x} \leq 4, e) = \top \qquad \delta(\_, \mathrm{x} \leq 4, w) = (p_w)$$

APT state:	TA location:	clock values:	tree address:
s	ℓ_0	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	ϵ
s	ℓ_1	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	2
s	ℓ_1	$\mathrm{x} \mapsto 1.4, \mathrm{y} \mapsto 0$	$2 \cdot 0$
p_c	ℓ_1	$\mathrm{x} \mapsto 1.4, \mathrm{y} \mapsto 0$	$2 \cdot 0 \cdot 0$
p_c	ℓ_4	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	$2 \cdot 0 \cdot 0$
p_c	ℓ_5	$\mathrm{x} \mapsto 3.2, \mathrm{y} \mapsto 0$	$2 \cdot 0 \cdot 0$
p_w	ℓ_5	$\mathrm{x} \mapsto 3.2, \mathrm{y} \mapsto 0$	$2 \cdot 0 \cdot 0 \cdot 0$



Example

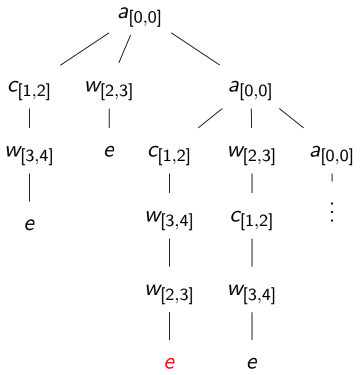


timed HORS:

$$S \mapsto T\ (F\ e)\ (G\ e)$$
$$T\ x\ y \mapsto a_{[0,0]}\ x\, y\, (T\ (F\ y)(G\ x))$$
$$F\ z \mapsto c_{[1,2]}\ w_{[3,4]}\ z$$
$$G\ t \mapsto w_{[2,3]}\ z$$

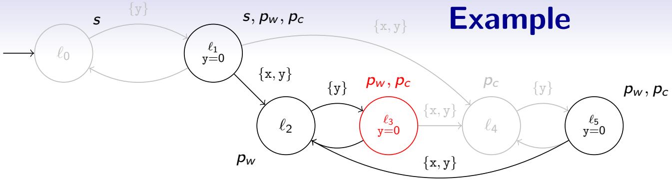
APT:

$$\delta(s, \_, a) = (s, s, s)$$
$$\delta(\_, \mathrm{x} > 4, \_) = \bot \qquad \delta(\_, \mathrm{x} \leq 4, c) = (p_c)$$
$$\delta(\_, \mathrm{x} \leq 4, e) = \top \qquad \delta(\_, \mathrm{x} \leq 4, w) = (p_w)$$



APT state:	TA location:	clock values:	tree address:
s	ℓ_0	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	ϵ
s	ℓ_1	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	2
s	ℓ_1	$\mathrm{x} \mapsto 1.4, \mathrm{y} \mapsto 0$	$2 \cdot 0$
p_c	ℓ_5	$\mathrm{x} \mapsto 3.2, \mathrm{y} \mapsto 0$	$2 \cdot 0 \cdot 0$
p_w	ℓ_5	$\mathrm{x} \mapsto 3.2, \mathrm{y} \mapsto 0$	$2 \cdot 0 \cdot 0 \cdot 0$
p_w	ℓ_3	$\mathrm{x} \mapsto 5.4, \mathrm{y} \mapsto 0$	$2 \cdot 0 \cdot 0 \cdot 0 \cdot 0$

Example



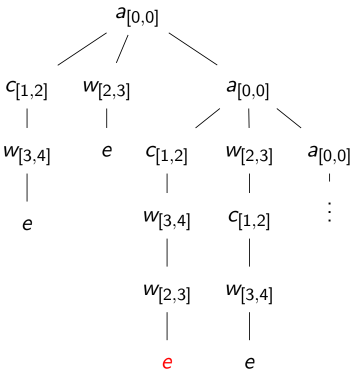
timed HORS:

$$S \mapsto T\ (F\ e)\ (G\ e)$$
$$T\ x\ y \mapsto a_{[0,0]}\ x\, y\, (T\ (F\ y)(G\ x))$$
$$F\ z \mapsto c_{[1,2]}\ w_{[3,4]}\ z$$
$$G\ t \mapsto w_{[2,3]}\ z$$

APT:

$$\delta(s, \_, a) = (s, s, s)$$
$$\delta(\_, \mathrm{x} > 4, \_) = \bot \qquad \delta(\_, \mathrm{x} \leq 4, c) = (p_c)$$
$$\delta(\_, \mathrm{x} \leq 4, e) = \top \qquad \delta(\_, \mathrm{x} \leq 4, w) = (p_w)$$

APT state:	TA location:	clock values:	tree address:
s	ℓ_0	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	ϵ
s	ℓ_1	$\mathrm{x} \mapsto 0, \mathrm{y} \mapsto 0$	2
s	ℓ_1	$\mathrm{x} \mapsto 1.4, \mathrm{y} \mapsto 0$	$2 \cdot 0$
p_c	ℓ_5	$\mathrm{x} \mapsto 3.2, \mathrm{y} \mapsto 0$	$2 \cdot 0 \cdot 0$
p_w	ℓ_5	$\mathrm{x} \mapsto 3.2, \mathrm{y} \mapsto 0$	$2 \cdot 0 \cdot 0 \cdot 0$
p_w	ℓ_3	$\mathrm{x} \mapsto 5.4, \mathrm{y} \mapsto 0$	$2 \cdot 0 \cdot 0 \cdot 0 \cdot 0$
???	ℓ_3	$\mathrm{x} \mapsto 5.4, \mathrm{y} \mapsto 0$	$2 \cdot 0 \cdot 0 \cdot 0 \cdot 0$



Upper Bounds for Timed-HORS Model-Checking

Theorem 1

Model-checking order-k timed HORS is in $(k+1)$-EXPTIME

Upper Bounds for Timed-HORS Model-Checking

Theorem 1

Model-checking order-k timed HORS is in $(k+1)$-EXPTIME

Proof sketch:

- discretize state space generated by timed automaton $\dashv$ via (well-known) region-graph construction (yields graph $R(\mathcal{A})$)
- incorporate resulting graph into APT ($\rightsquigarrow$ exponential blowup)

Upper Bounds for Timed-HORS Model-Checking

Theorem 1

Model-checking order-k timed HORS is in $(k+1)$-EXPTIME

Proof sketch:

- discretize state space generated by timed automaton $\dashv$ via (well-known) region-graph construction (yields graph $R(\mathcal{A})$)
- incorporate resulting graph into APT ($\rightsquigarrow$ exponential blowup)

$$\text{tAPT } \mathcal{P},\ \text{TA } \mathcal{A} \xrightarrow{\text{exp.}} \text{APT } \mathcal{P} \times R(\mathcal{A})$$

$$\text{timed HORS } \mathcal{G} \xrightarrow[\text{lin.}]{} \text{untimed HORS } \mathcal{G}'$$

main difficulty:

- interaction to let time flow cannot be added directly into APT logic

Upper Bounds for Timed-HORS Model-Checking

Theorem 1

Model-checking order-k timed HORS is in $(k+1)$-EXPTIME

Proof sketch:

- discretize state space generated by timed automaton $\dashv$ via (well-known) region-graph construction (yields graph $R(\mathcal{A})$)
- incorporate resulting graph into APT ($\rightsquigarrow$ exponential blowup)

$$
\begin{tikzcd}
\text{tAPT } \mathcal{P}, \text{ TA } \mathcal{A} \arrow[r, "\text{exp.}"] \arrow[dr, "\text{lin.}"] & \text{APT } \mathcal{P} \times R(\mathcal{A}) \\
\text{timed HORS } \mathcal{G} \arrow[r, "\text{lin.}"'] & \text{untimed HORS } \mathcal{G}'
\end{tikzcd}
$$

main difficulty:

- interaction to let time flow cannot be added directly into APT logic
- needs to be emulated on grammar side via gadgets

Lower Bound for $k = 1$

Theorem 2

Model-Checking order-1 timed HORS is 2-EXPTIME-complete.

Lower Bound for $k = 1$

Theorem 2

Model-Checking order-1 timed HORS is 2-EXPTIME-complete.

Prop.: (Chandra, Kozen, Stockmeyer) acceptance of a 2-EXPTIME DTM on input w witnessed by $2^{2^{|w|}} \times 2^{2^{|w|}}$ table listing the configurations in order

#	q_0, i	n	p	u	t	$\square$	$\ldots$	$\square$	#
#	a	n, q_1	p	u	t	$\square$	$\ldots$	$\square$	#
#	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	#
#									#
#	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	#
#	$q_f, \square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	#

Lower Bound for $k = 1$

Theorem 2

Model-Checking order-1 timed HORS is 2-EXPTIME-complete.

Prop.: (Chandra, Kozen, Stockmeyer) acceptance of a 2-EXPTIME DTM on input w witnessed by $2^{2^{|w|}} \times 2^{2^{|w|}}$ table listing the configurations in order

$\#$	q_0, i	n	p	u	t	$\square$	$\dots$	$\square$	$\#$
$\#$	a	n, q_1	p	u	t	$\square$	$\dots$	$\square$	$\#$
$\#$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\#$
$\#$									$\#$
$\#$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\#$
$\#$	$q_f, \square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\#$

important observation: whether a cell of the table contains an entry depends only on

- boundary conditions ($\#$ on sides, first and last row)

Lower Bound for $k = 1$

Theorem 2

Model-Checking order-1 timed HORS is 2-EXPTIME-complete.

Prop.: (Chandra, Kozen, Stockmeyer) acceptance of a 2-EXPTIME DTM on input w witnessed by $2^{2^{|w|}} \times 2^{2^{|w|}}$ table listing the configurations in order

$\#$	q_0, i	n	p	u	t	$\square$	$\dots$	$\square$	$\#$
$\#$	a	n, q_1	p	u	t	$\square$	$\dots$	$\square$	$\#$
$\#$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\#$
$\#$									$\#$
$\#$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\#$
$\#$	$q_f, \square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\#$

important observation: whether a cell of the table contains an entry depends only on

- boundary conditions ($\#$ on sides, first and last row), and
- the cells directly or diagonally above the cell in question

$\rightsquigarrow$ existence of a table can be checked recursively checking for local consistency!

Counting with HORS, APT and TA

- working with an acc. witnessing table requires encoding numbers in $[0, 2^{2^{|w|}} - 1]$
- re-use idea from TIME'22 [B./Lange]: integral clock values in $[0, 2^{|w|} - 1]$ as bits

Counting with HORS, APT and TA

- working with an acc. witnessing table requires encoding numbers in $[0, 2^{2^{|w|}} - 1]$
- re-use idea from TIME'22 [B./Lange]: integral clock values in $[0, 2^{|w|} - 1]$ as bits

Lemma 1: There is an APT that implements Boolean logic over a suitable HORS (including negation!)

Counting with HORS, APT and TA

- working with an acc. witnessing table requires encoding numbers in $[0, 2^{2^{|w|}} - 1]$
- re-use idea from TIME'22 [B./Lange]: integral clock values in $[0, 2^{|w|} - 1]$ as bits

Lemma 1: There is an APT that implements Boolean logic over a suitable HORS (including negation!)

Lemma 2: Quantification over bits ($=$ clock values) is possible using a TA.

Counting with HORS, APT and TA

- working with an acc. witnessing table requires encoding numbers in $[0, 2^{2^{|w|}} - 1]$
- re-use idea from TIME'22 [B./Lange]: integral clock values in $[0, 2^{|w|} - 1]$ as bits

Lemma 1: There is an APT that implements Boolean logic over a suitable HORS (including negation!)

Lemma 2: Quantification over bits (= clock values) is possible using a TA.

Lemma 3: Manipulation of numbers in $[0, 2^{2^{|w|}} - 1]$ is possible using a suitable timed APT, TA and timed HORS.

Counting with HORS, APT and TA

- working with an acc. witnessing table requires encoding numbers in $[0, 2^{2^{|w|}} - 1]$
- re-use idea from TIME'22 [B./Lange]: integral clock values in $[0, 2^{|w|} - 1]$ as bits

Lemma 1: There is an APT that implements Boolean logic over a suitable HORS (including negation!)

Lemma 2: Quantification over bits (= clock values) is possible using a TA.

Lemma 3: Manipulation of numbers in $[0, 2^{2^{|w|}} - 1]$ is possible using a suitable timed APT, TA and timed HORS.

Lemma 4: The logical structure of an acc. witnessing table can be encoded into (the tree generated by) a timed HORS, using Lemma 3 for indexing of cells.

Counting with HORS, APT and TA

- working with an acc. witnessing table requires encoding numbers in $[0, 2^{2^{|w|}} - 1]$
- re-use idea from TIME'22 [B./Lange]: integral clock values in $[0, 2^{|w|} - 1]$ as bits

Lemma 1: There is an APT that implements Boolean logic over a suitable HORS (including negation!)

Lemma 2: Quantification over bits (= clock values) is possible using a TA.

Lemma 3: Manipulation of numbers in $[0, 2^{2^{|w|}} - 1]$ is possible using a suitable timed APT, TA and timed HORS.

Lemma 4: The logical structure of an acc. witnessing table can be encoded into (the tree generated by) a timed HORS, using Lemma 3 for indexing of cells.

Proof of Theorem 2: verify structure of tree from Lemma 4 using an extension if the APT from Lemma 1, and the TA from Lemma 2/3.

Conclusion

results

- introduced real-time HORS
- first combination of higher-order structures and real-time mechanics

Conclusion

results

- introduced real-time HORS
- first combination of higher-order structures and real-time mechanics
- upper bound as expected: adding real time costs one exponential (cf. [B./Lange Time'21,'22]

Conclusion

results

- introduced real-time HORS
- first combination of higher-order structures and real-time mechanics
- upper bound as expected: adding real time costs one exponential (cf. [B./Lange Time'21,'22]
- lower bound for $k = 1$: added exponential is unavoidable

Conclusion

results

- introduced real-time HORS
- first combination of higher-order structures and real-time mechanics
- upper bound as expected: adding real time costs one exponential (cf. [B./Lange Time'21,'22]
- lower bound for $k = 1$: added exponential is unavoidable

future work:

- find specification logic in place of APT and timed automata (likely version of timed μ-calculus)
    - requires TA in universal semantics (i.e., spoiler can also control flow of time)

Conclusion

results

- introduced real-time HORS
- first combination of higher-order structures and real-time mechanics
- upper bound as expected: adding real time costs one exponential (cf. [B./Lange Time'21,'22]
- lower bound for $k = 1$: added exponential is unavoidable

future work:

- find specification logic in place of APT and timed automata (likely version of timed μ-calculus)
    - requires TA in universal semantics (i.e., spoiler can also control flow of time)
- lower bound for all k

Conclusion

results

- introduced real-time HORS
- first combination of higher-order structures and real-time mechanics
- upper bound as expected: adding real time costs one exponential (cf. [B./Lange Time'21,'22]
- lower bound for $k = 1$: added exponential is unavoidable

future work:

- find specification logic in place of APT and timed automata (likely version of timed μ-calculus)
    - requires TA in universal semantics (i.e., spoiler can also control flow of time)
- lower bound for all k
- practical implementation (zone graphs and higher order?)

Questions?

Conclusion

results

- introduced real-time HORS
- first combination of higher-order structures and real-time mechanics
- upper bound as expected: adding real time costs one exponential (cf. [B./Lange Time'21,'22]
- lower bound for $k = 1$: added exponential is unavoidable

future work:

- find specification logic in place of APT and timed automata (likely version of timed μ-calculus)
  - requires TA in universal semantics (i.e., spoiler can also control flow of time)
- lower bound for all k
- practical implementation (zone graphs and higher order?)

Questions? On to the wine and cheese!