TCTL model checking lower/upper-bound parametric timed automata without invariants

FORMATS 2018

Étienne André *, Didier Lime **& **Mathias Ramparison***

*LIPN, Université Paris 13
**LS2N, École Centrale de Nantes

September 5th, 2018

Outline

Introduction

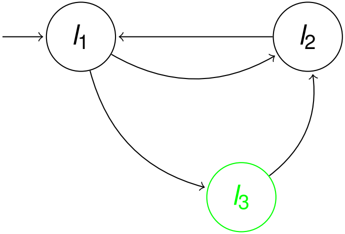
- Discovering a bug during a test of a system can be very expensive
- Can have dramatical consequences in critical embedded system: autonomous car, in aeronautics...

Introduction

- Discovering a bug during a test of a system can be very expensive
- Can have dramatical consequences in critical embedded system: autonomous car, in aeronautics...
- Need for formal verification to ensure ahead the good behavior of a system

Model checking

- Model of a system:

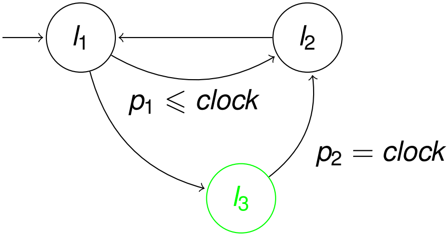


- A property of the system: 

is reachable
- Check whether the system satisfies the property
- Timed Automata [AD94] is a powerful formalism when all timing constants are known

Model checking with unknown constants

- *What if all constants are not specified ahead?*
- Model of a system with parameters:

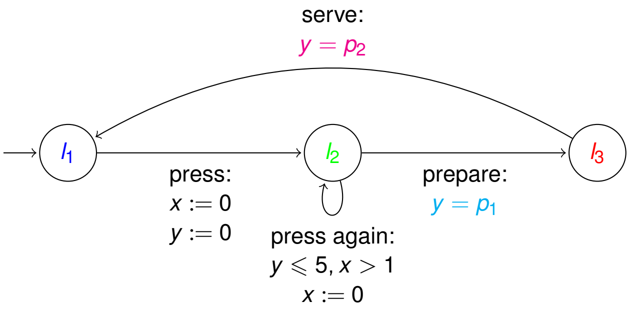


- A property of the system: 

 is reachable
- Compute the values of p_1, p_2 such that the system satisfies the property

Example of parametric timed automaton

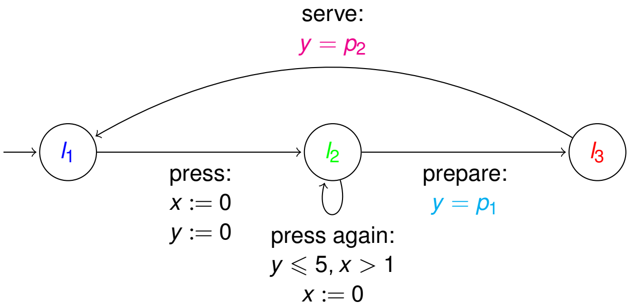
A **parametric timed automaton** [AHV93] which models a parametric coffee machine



- Locations : $\{l_1, l_2, l_3\}$, clocks : $\{x, y\}$, action : {press, press again, prepare, serve}
- $Guard(\text{press again}) = \{y \leqslant 5 \wedge x \geqslant 0\}$, $Guard(\text{prepare}) = \{y = p_1\}$, $Guard(\text{serve}) = \{y = p_2\}$
- $Reset(\text{press}) = \{x, y := 0\}$, $Reset(\text{press again}) = \{x := 0\}$

Example of parametric timed automaton

A **parametric timed automaton** [AHV93] which models a parametric coffee machine



- A possible run if $p_1 = 2, p_2 = 3$: $\big(l_1, (0,0)\big) \xrightarrow[2]{\text{press}} \big(l_2, (0,0)\big) \xrightarrow[1.1]{\text{press again}} \big(l_2, (0, 1.1)\big) \xrightarrow[.9]{\text{prepare}} \big(l_3, (.9, 2)\big) \xrightarrow[1]{\text{serve}} \big(l_1, (1.9, 3)\big)$
- The same run is impossible if $p_1 = 5, p_2 = 2$.

Flat (no nesting) TCTL decision problems for PTAs

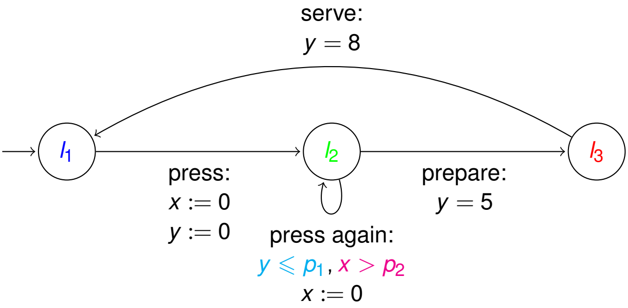
- *EF-emptiness*: is the set of parameter valuations s.t. there exists a run reaching *l* in the instantiated TA empty ?
- *EF-universality*: are all parameter valuations s.t. there exists a run reaching *l* in the instantiated TA
- *EG-emptiness*: is the set of valuations for which one infinite or finite maximal runs always remains in a given set of locations empty?
- *AF-emptiness*: is the set of valuations for which all runs eventually reach a given location empty? (equivalent to *EG*-universality)
- *AG-emptiness*: is the set of valuations for which all infinite or finite maximal run always remain in a given set of locations empty?

Challenges for parametric timed automata

- *EF*-emptiness problem: proved undecidable in general case [AHV93], unbounded integer-valued parameters, (un)bounded rational valued parameters and even with only one bounded parameter [Mil00]
- To recover decidability, we need to add restrictions on parameters, or restrain the PTA syntax

L/U-PTA

Lower/upper bound PTAs (L/U-PTAs) introduced in [HRSV02]. Here is an L/U-PTA without invariant.



Comparison with:

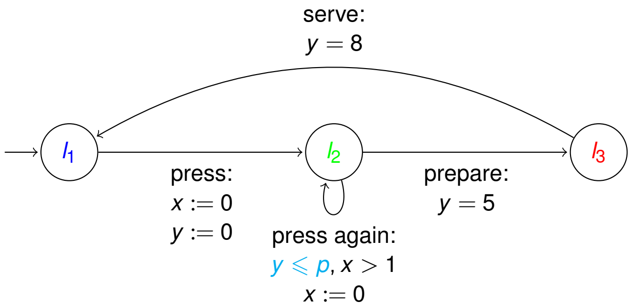
- Upper-bound parameter p_1.
- Lower-bound parameter p_2.

U-PTA

U-PTAs [BL09]: no undecidability result, and almost all decidability results are from L/U-PTAs

- Decidability of EF-emptiness and universality for integer-valued U-PTAs [BL09]
- Decidability language preservation synthesis for one parameter and a deterministic automaton [AM15]

Here is a U-PTA without invariant.



Upper-bound parametric guard: $y \leqslant p$.

Current results and contributions

Class	U-PTAs	integer-valued L/U-PTAs without invariant	L/U-PTAs	PTAs
EF	[HRSV02]	[HRSV02]	[HRSV02]	*[AHV93, Mil00]*
AF	open	open	*[JLR15]*	*[JLR15]*
EG	open	open	*[AL17]*	*[AL17]*
AG	[HRSV02]	[HRSV02]	[HRSV02]	*[ALR16a]*
flat TCTL	open	open	*[JLR15]*	*[AHV93]*
TCTL	open	open	*[JLR15]*	*[AHV93]*

Table: Decidability of the emptiness problems for PTAs and subclasses

Contributions:

- Undecidability of non-flat TCTL (with nesting) for unbounded U-PTA without invariant
- Undecidability of non-flat TCTL for bounded U-PTAs without invariant
- Decidability of EG-emptiness/universality (in PSPACE) for integer-valued L/U-PTAs without invariant

U-PTA

U-PTAs without invariant with rational-valued parameters over dense time.

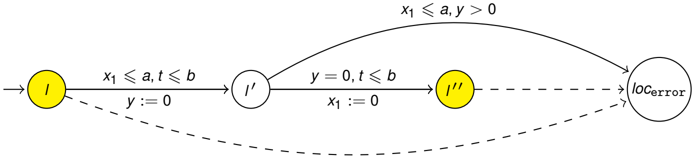
U-PTA

U-PTAs without invariant with rational-valued parameters over dense time.

Theorem (1)

Non-flat-TCTL is undecidable for rational-valued U-PTAs without invariant.

Proof sketch: we prove that the $EGAF_{=0}$-emptiness problem is *undecidable* for rational-valued U-PTAs without invariant, using a reduction from the halting problem of a two counter machine



U-PTA

U-PTAs without invariant with bounded rational-valued parameters over dense time

Motivation:

- it is impossible to simulate a bounded U-PTA using a U-PTA [ALR16b],
- and EG-emptiness is decidable for bounded L/U-PTAs, but undecidable for L/U-PTAs [AL17].

U-PTA

U-PTAs without invariant with bounded rational-valued parameters over dense time

Motivation:

- it is impossible to simulate a bounded U-PTA using a U-PTA [ALR16b],
- and EG-emptiness is decidable for bounded L/U-PTAs, but undecidable for L/U-PTAs [AL17].

Theorem (2)

Non-flat-TCTL is undecidable for bounded rational-valued U-PTAs without invariant.

Proof sketch: we prove that the $EGAF_{=0}$-emptiness problem is *undecidable* for bounded rational-valued U-PTAs without invariant, using a reduction from the boundedness problem of a two counter machine

Current results

Class	U-PTAs	integer-valued L/U-PTAs without invariant	L/U-PTAs	PTAs
EF	[HRSV02]	[HRSV02]	[HRSV02]	*[AHV93, Mil00]*
AF	open	open	*[JLR15]*	*[JLR15]*
EG	open	open	*[AL17]*	*[AL17]*
AG	[HRSV02]	[HRSV02]	[HRSV02]	*[ALR16a]*
flat TCTL	open	open	*[JLR15]*	*[AHV93]*
TCTL	***Theorem 1***	***Theorem 1***	*[JLR15]*	*[AHV93]*

Table: Decidability of the emptiness problems for PTAs and subclasses

L/U-PTA

L/U-PTAs without invariant with integer-valued parameters over dense time.

L/U-PTA

L/U-PTAs without invariant with integer-valued parameters over dense time.

Theorem (3)

The EG-emptiness/universality problems are PSPACE-complete for integer-valued L/U-PTAs without invariant.

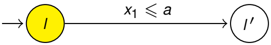
Corollary

Flat-TCTL is decidable for integer-valued L/U-PTAs without invariant (using [HRSV02]).

L/U-PTA

Proof sketch: We reduce this problem to reachability of a location

Is there possibly a deadlock ?



L/U-PTA

Proof sketch: We reduce this problem to reachability of a location

Remove transition and add self loop

And then check whether there is an infinite run

Summary of contributions and conclusion

Class	U-PTAs	integer-valued L/U-PTAs without invariant	L/U-PTAs	PTAs
EF	[HRSV02]	[HRSV02]	[HRSV02]	*[AHV93, Mil00]*
AF	open	**Theorem 3**	*[JLR15]*	*[JLR15]*
EG	open	**Theorem 3**	*[AL17]*	*[AL17]*
AG	[HRSV02]	[HRSV02]	[HRSV02]	*[ALR16a]*
flat TCTL	open	**Theorem 3**	*[JLR15]*	*[AHV93]*
TCTL	***Theorem 1***	***Theorem 1***	*[JLR15]*	*[AHV93]*

Table: Decidability of the emptiness problems for PTAs and subclasses

- Non-flat-TCTL is undecidable for U-PTAs without invariant (bounded or not).
- EG-emptiness and universality (first non trivial subclass of PTAs) is decidable for integer-valued L/U-PTAs without invariant.

Summary of contributions and conclusion

Class	U-PTAs	integer-valued L/U-PTAs without invariant	L/U-PTAs	PTAs
EF	[HRSV02]	[HRSV02]	[HRSV02]	*[AHV93, Mil00]*
AF	open	**Theorem 3**	*[JLR15]*	*[JLR15]*
EG	open	**Theorem 3**	*[AL17]*	*[AL17]*
AG	[HRSV02]	[HRSV02]	[HRSV02]	*[ALR16a]*
flat TCTL	open	**Theorem 3**	*[JLR15]*	*[AHV93]*
TCTL	***Theorem 1***	***Theorem 1***	*[JLR15]*	*[AHV93]*

Table: Decidability of the emptiness problems for PTAs and subclasses

Future work:

- Where exactly the undecidability starts (in particular whether EG and AF are decidable for U-PTAs with invariants or real-valued parameters), which remains open,
- whether our proofs for bounded U-PTAs can be extended over bounded time,
- whether the same results hold for L-PTAs (lower-bound PTAs).

Advertisement: Paris summer school 2019

Paris Summer School 2019 in
Specification and Verification of Critical Systems

1st to 19th July 2019

- For Chinese Master students
- Supported by Campus France / French Embassy in Beijing
- Scientific organization: most Paris labs in verification (Sorbonne Université, ENS Paris-Saclay, Le CNAM, Université Paris-Est Créteil Val de Marne, Sorbonne Paris Cité. . .)

More info:

`www.lipn13.fr/paris-2019/`

References

Rajeev Alur and David L. Dill.
A theory of timed automata.
Theoretical Computer Science, 126(2):183–235, April 1994.

Rajeev Alur, Thomas A. Henzinger, and Moshe Y. Vardi.
Parametric real-time reasoning.
In S. Rao Kosaraju, David S. Johnson, and Alok Aggarwal, editors, *STOC*, pages 592–601, New York, NY, USA, 1993. ACM.

Étienne André and Didier Lime.
Liveness in L/U-parametric timed automata.
In Alex Legay and Klaus Schneider, editors, *ACSD*, pages 9–18. IEEE, 2017.

Étienne André, Didier Lime, and Olivier H. Roux.
Decision problems for parametric timed automata.
In Kazuhiro Ogata, Mark Lawford, and Shaoying Liu, editors, *ICFEM*, volume 10009 of *Lecture Notes in Computer Science*, pages 400–416. Springer, 2016.

Étienne André, Didier Lime, and Olivier H. Roux.
On the expressiveness of parametric timed automata.
In Martin Fränzle and Nicolas Markey, editors, *FORMATS*, volume 9984, pages 19–34. Springer, 2016.

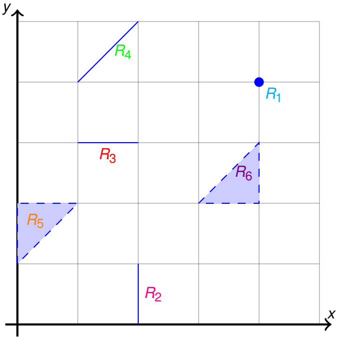
Étienne André and Nicolas Markey.
Language preservation problems in parametric timed automata.
In *FORMATS*, volume 9268, pages 27–43. Springer, 2015.

Laura Bozzelli and Salvatore La Torre.
Decision problems for lower/upper bound parametric timed automata.
Formal Methods in System Design, 35(2):121–151, 2009.

Thomas Hune, Judi Romijn, Mariëlle Stoelinga, and Frits W. Vaandrager.
Linear parametric model checking of timed automata.
Journal of Logic and Algebraic Programming, 52-53:183–220, 2002.

Aleksandra Jovanović, Didier Lime, and Olivier H. Roux.

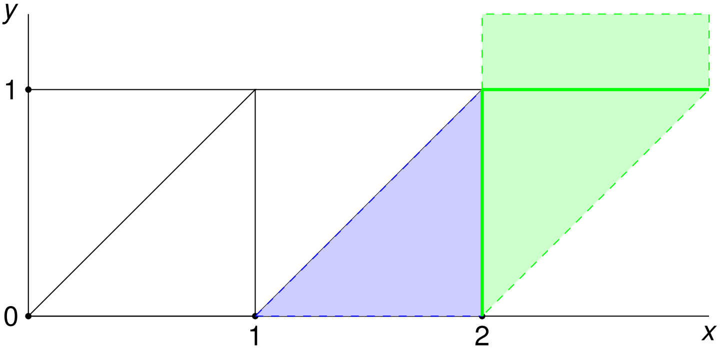
Clock regions



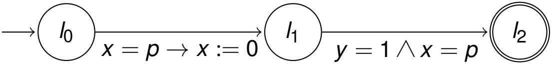
- The corner point: $R_1 = \{(4, 4)\}$
- The vertical line: $R_2 = \{(x, y) \mid x = 2, 0 < y < 1\}$
- The horizontal line: $R_3 = \{(x, y) \mid y = 3, 1 < x < 2\}$
- The diagonal: $R_4 = \{(x, y) \mid x = y - 3, 4 < y < 5\}$
- The upward triangle: $R_5 = \{(x, y) \mid 0 < x < y - 1, 1 < y < 2\}$
- The downward triangle: $R_6 = \{(x, y) \mid y + 1 < x < 4, 2 < y < 3\}$

Clock regions

Two clocks x, y, max constants $c_x = 2$, $c_y = 1$.
Time successors of the blue region $\{0 < y < 1, 0 < y < x - 1\}$ different of itself: four regions in green: $\{0 < y < 1, x = 2\}$, $\{0 < y < 1, x > 2\}$, $\{y = 1, x > 2\}$ and $\{y > 1, x > 2\}$

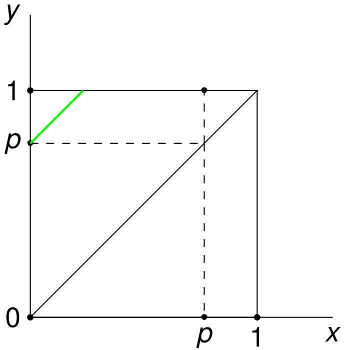


Using regions for parametric timed automata ?

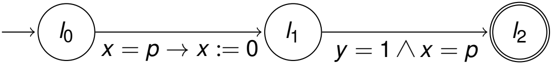


In l_1: $(x, y) = (0, p)$
But after letting some time elapse, depending on the value of $0 < p < 1$ we reach different regions:

- region $y = 1, 0 < x < p$ if $1 > p > \frac{1}{2}$

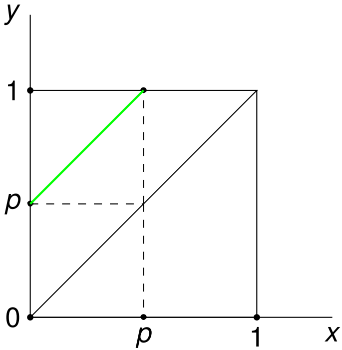


Using regions for parametric timed automata ?

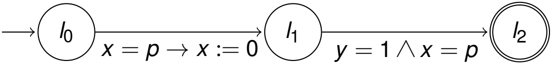


In l_1: $(x, y) = (0, p)$
But after letting some time elapse, depending on the value of $0 < p < 1$ we access different regions:

- region $y = 1, x = p$ if $p = \frac{1}{2}$



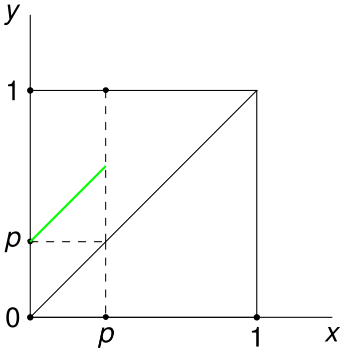
Using regions for parametric timed automata ?



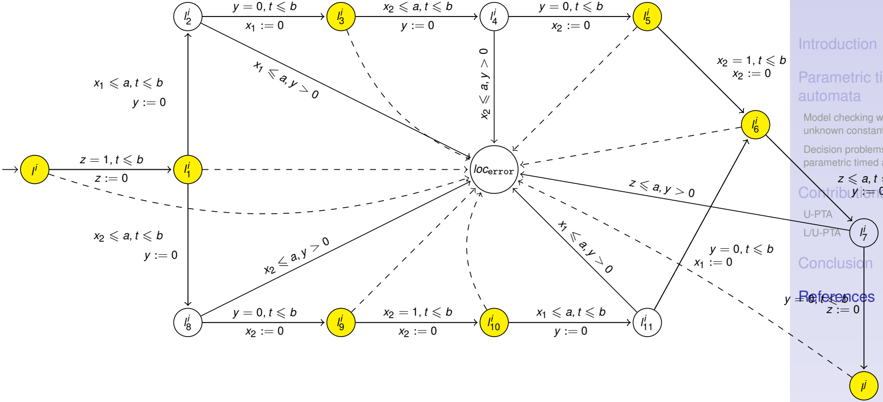
In l_1: $(x, y) = (0, p)$

But after letting some time elapse, depending on the value of $0 < p < 1$ we access different regions:

- region $p < y < 1, x = p$ if $p < \frac{1}{2}$



Proof sketch U-PTAs



Proof sketch bounded U-PTAs