FTSCS 2018

16th Novemer 2018
Gold Coast, Australia

A benchmarks library for parametric timed model checking

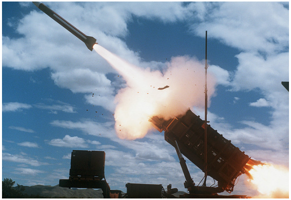
Étienne André

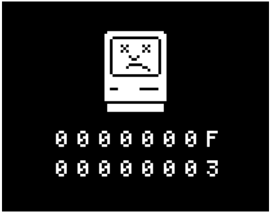
LIPN, Université Paris 13, CNRS, France
National Institute of Informatics, Japan
JFLI, UMI CNRS, Tokyo, Japan

Supported by the ANR national research program PACS (ANR-14-CE28-0002) and JST ERATO HASUO Metamathematics for Systems Design Project (No. JPMJER1603).

Context: Verifying complex timed systems

- Real-time systems are everywhere
  - Hard timing constraints and concurrency
  - Criticality: risk for huge damages in case of unexpected behavior (bug)
  - Bugs discovered when final testing: expensive
  - ⇝ Need for a thorough specification and verification phase



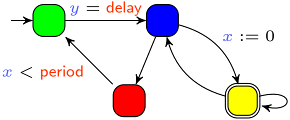


Outline

Outline

Model checking timed concurrent systems

- Use formal methods [Baier and Katoen, 2008]



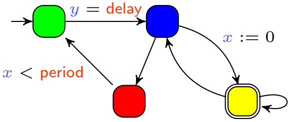
A model of the system

 is unreachable

A property to be satisfied

Model checking timed concurrent systems

- Use formal methods [Baier and Katoen, 2008]



A model of the system

$$\overset{?}{\models}$$

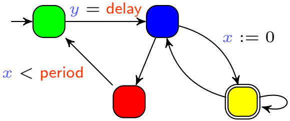
 is unreachable

A property to be satisfied

- Question: does the model of the system satisfy the property?

Model checking timed concurrent systems

- Use formal methods [Baier and Katoen, 2008]



A model of the system

$$\overset{?}{\models}$$

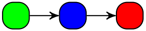
 is unreachable

A property to be satisfied

- Question: does the model of the system satisfy the property?

Yes

No



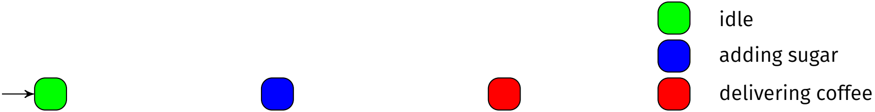
Counterexample

Turing award (2007) to Edmund M. Clarke, Allen Emerson and Joseph Sifakis

Outline

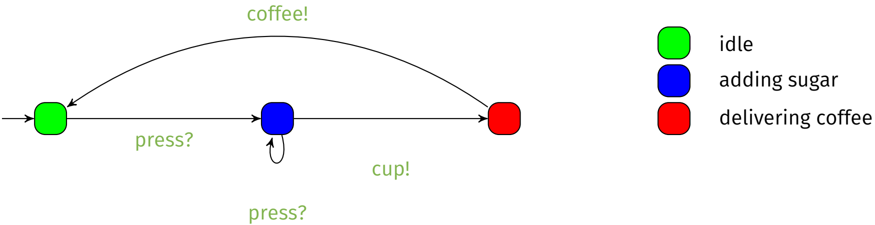
Timed automaton (TA)

- Finite state automaton (sets of locations)



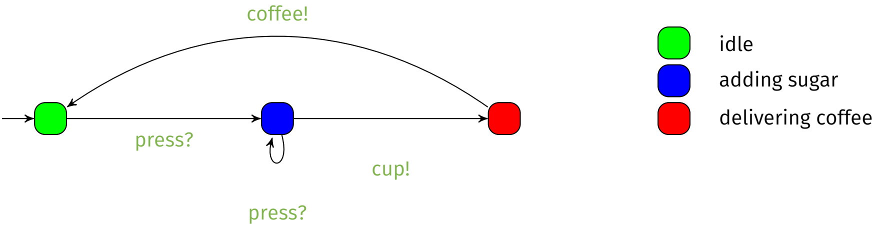
Timed automaton (TA)

- Finite state automaton (sets of locations and actions)



Timed automaton (TA)

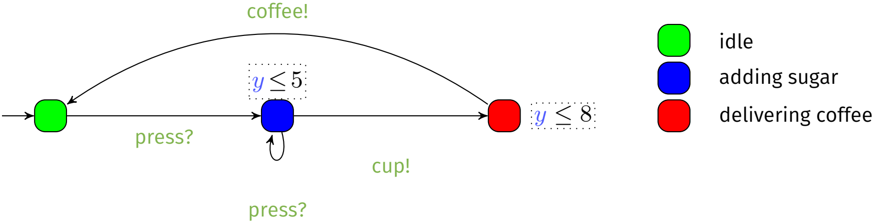
- Finite state automaton (sets of locations and actions) augmented with a set X of clocks [Alur and Dill, 1994]
  - Real-valued variables evolving linearly at the same rate



Timed automaton (TA)

- Finite state automaton (sets of locations and actions) augmented with a set X of clocks [Alur and Dill, 1994]
  - Real-valued variables evolving linearly at the same rate
  - Can be compared to integer constants in invariants

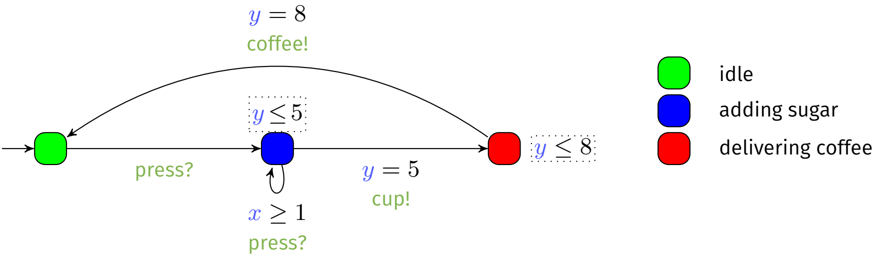
- Features
  - Location invariant: property to be verified to stay at a location



Timed automaton (TA)

- Finite state automaton (sets of locations and actions) augmented with a set X of clocks [Alur and Dill, 1994]
  - Real-valued variables evolving linearly at the same rate
  - Can be compared to integer constants in invariants and guards

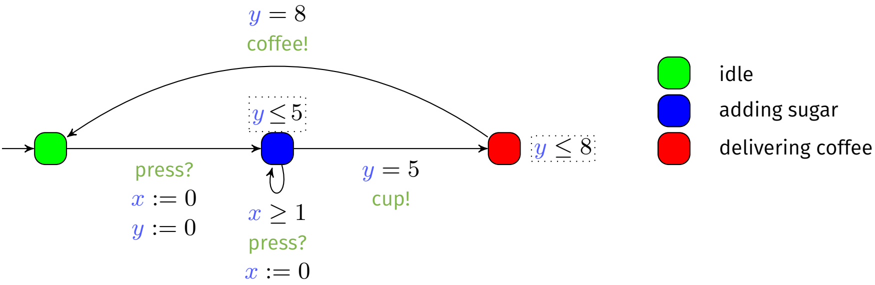
- Features
  - Location invariant: property to be verified to stay at a location
  - Transition guard: property to be verified to enable a transition



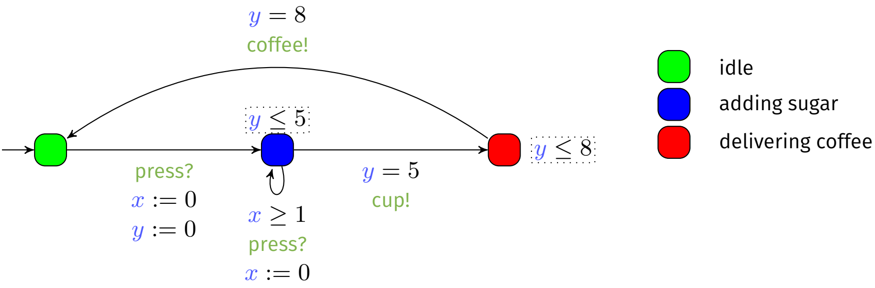
Timed automaton (TA)

- Finite state automaton (sets of locations and actions) augmented with a set X of clocks [Alur and Dill, 1994]
  - Real-valued variables evolving linearly at the same rate
  - Can be compared to integer constants in invariants and guards

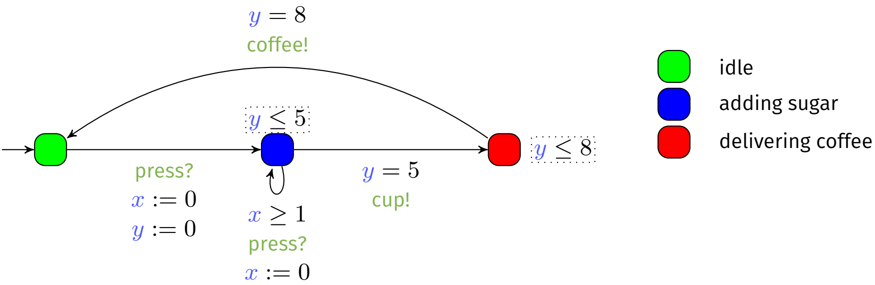
- Features
  - Location invariant: property to be verified to stay at a location
  - Transition guard: property to be verified to enable a transition
  - Clock reset: some of the clocks can be set to 0 along transitions



The most critical system: The coffee machine



The most critical system: The coffee machine

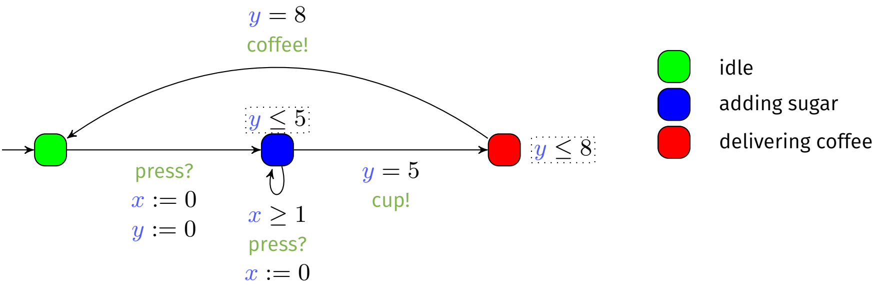


- Example of concrete run for the coffee machine
  - Coffee with 2 doses of sugar

$x =$ 0

$y =$ 0

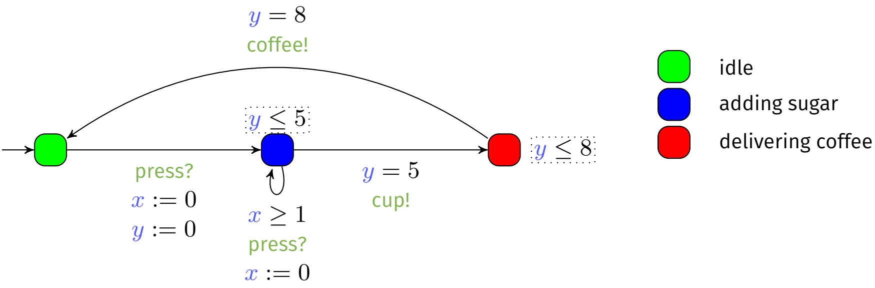
The most critical system: The coffee machine



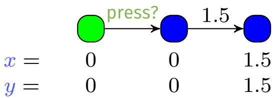
- Example of concrete run for the coffee machine
  - Coffee with 2 doses of sugar

$x =$	0	0
$y =$	0	0

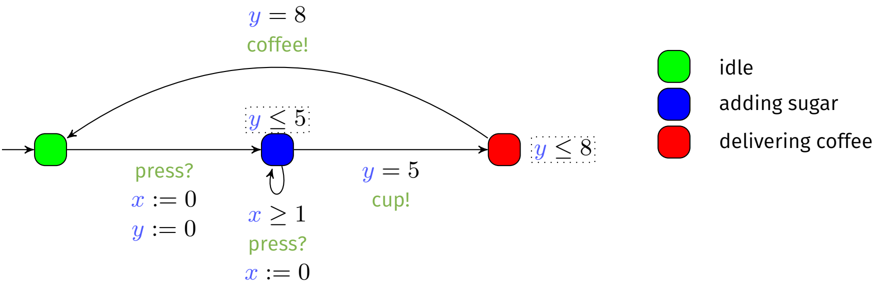
The most critical system: The coffee machine



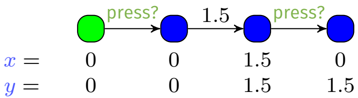
- Example of concrete run for the coffee machine
  - Coffee with 2 doses of sugar



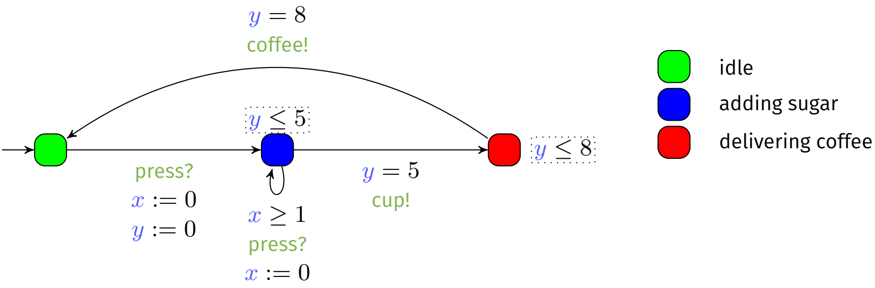
The most critical system: The coffee machine



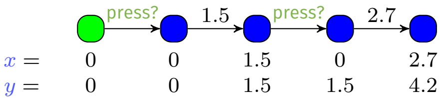
- Example of concrete run for the coffee machine
  - Coffee with 2 doses of sugar



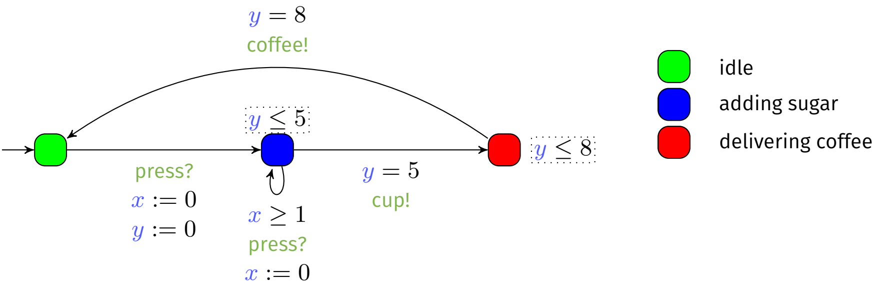
The most critical system: The coffee machine



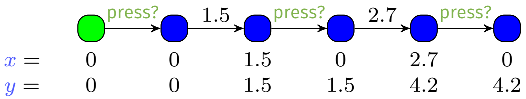
- Example of concrete run for the coffee machine
  - Coffee with 2 doses of sugar



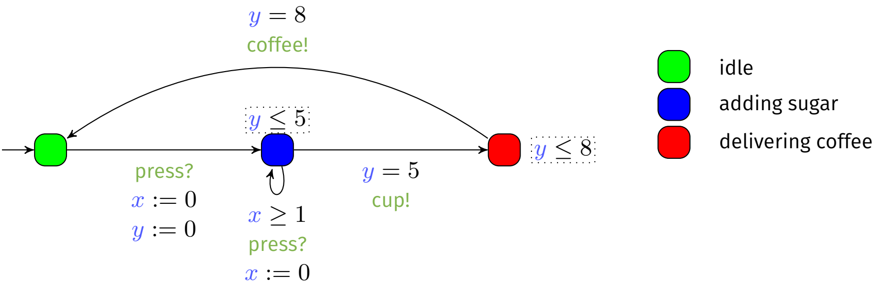
The most critical system: The coffee machine



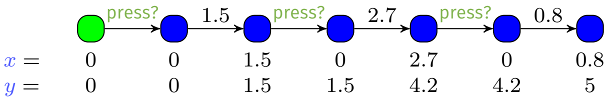
- Example of concrete run for the coffee machine
  - Coffee with 2 doses of sugar



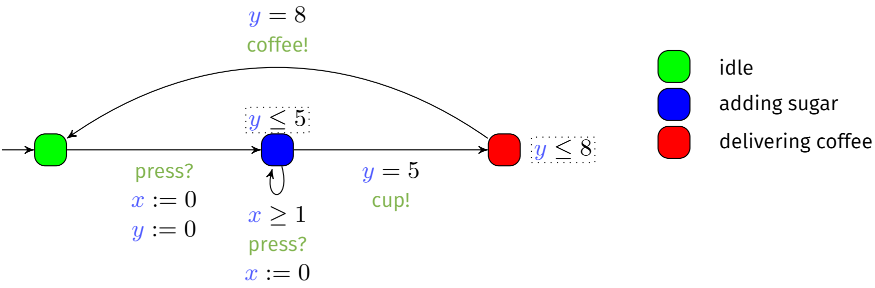
The most critical system: The coffee machine



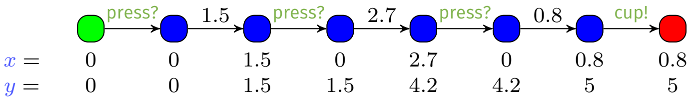
- Example of concrete run for the coffee machine
  - Coffee with 2 doses of sugar



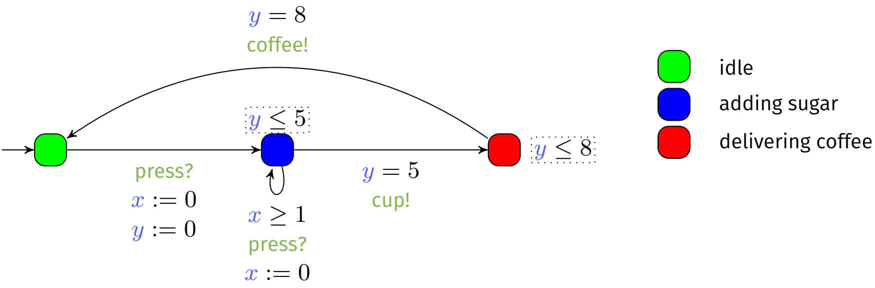
The most critical system: The coffee machine



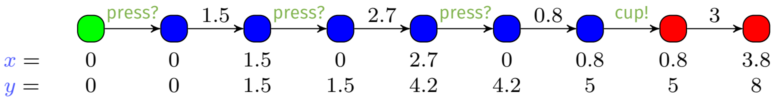
- Example of concrete run for the coffee machine
  - Coffee with 2 doses of sugar



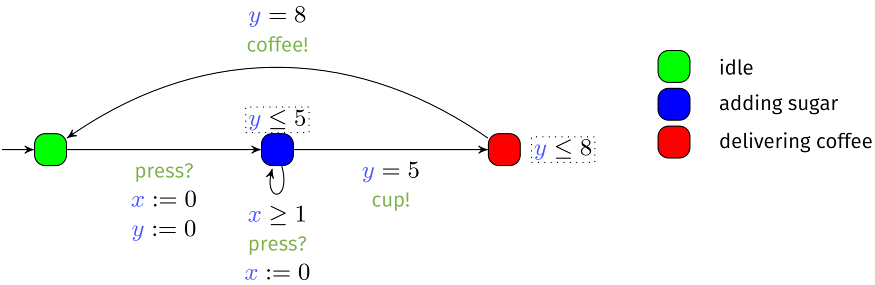
The most critical system: The coffee machine



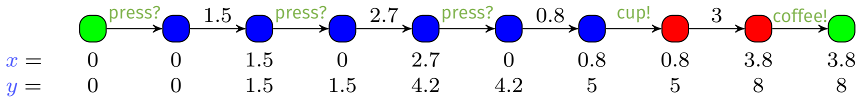
- Example of concrete run for the coffee machine
  - Coffee with 2 doses of sugar



The most critical system: The coffee machine



- Example of concrete run for the coffee machine
  - Coffee with 2 doses of sugar



Timed automata: A success story

- An expressive formalism
  - Dense time
  - Concurrency
- A tractable verification in theory
  - Reachability is PSPACE-complete [Alur and Dill, 1994]
- A very efficient verification in practice
  - Symbolic verification: relatively insensitive to constants
  - Several model checkers, notably UPPAAL [Larsen et al., 1997]
  - Long list of successful case studies

Outline

Beyond timed model checking: parameter synthesis

- Verification for one set of constants does not usually guarantee the correctness for other values

- Challenges
  - Numerous verifications: is the system correct for any value within $[40; 60]$?
  - Optimization: until what value can we increase 10?
  - Robustness [Bouyer et al., 2013]: What happens if 50 is implemented with 49.99?
  - System incompletely specified: Can I verify my system even if I don't know the period value with full certainty?

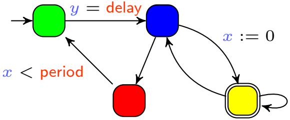
Beyond timed model checking: parameter synthesis

- Verification for one set of constants does not usually guarantee the correctness for other values

- Challenges
  - Numerous verifications: is the system correct for any value within $[40; 60]$?
  - Optimization: until what value can we increase 10?
  - Robustness [Bouyer et al., 2013]: What happens if 50 is implemented with 49.99?
  - System incompletely specified: Can I verify my system even if I don't know the period value with full certainty?

- Parameter synthesis
  - Consider that timing constants are unknown constants (parameters)

timed model checking



$\stackrel{?}{\models}$

 is unreachable

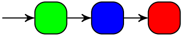
A model of the system

A property to be satisfied

- Question: does the model of the system satisfy the property?

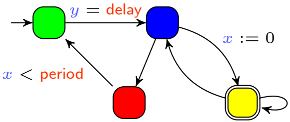
Yes

No



Counterexample

Parametric timed model checking



A model of the system

$\overset{?}{\models}$

 is unreachable

A property to be satisfied

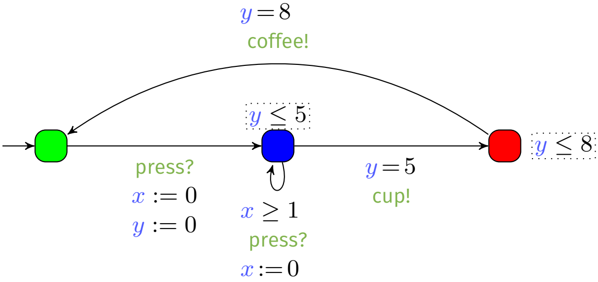
- Question: for what values of the parameters does the model of the system satisfy the property?

Yes if...

$$2\text{delay} > \text{period}$$
$$\wedge\ \text{period} < 20.46$$

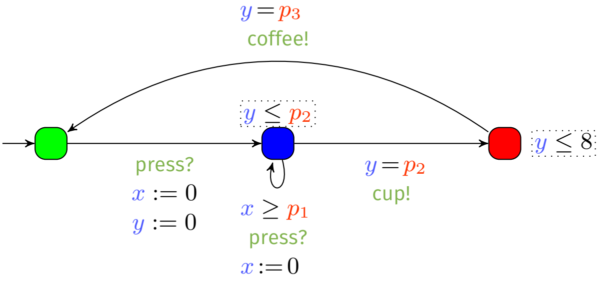
Parametric Timed Automaton (PTA)

- Timed automaton (sets of locations, actions and clocks)



Parametric Timed Automaton (PTA)

- Timed automaton (sets of locations, actions and clocks) augmented with a set P of parameters [Alur et al., 1993]
  - Unknown constants compared to a clock in guards and invariants



1990s: undecidability

- EF-emptiness problem
  "Is the set of parameter valuations for which a given location l is reachable empty?"
  undecidable [Alur et al., 1993, Miller, 2000, Doyen, 2007, Beneš et al., 2015]

1990s: undecidability

- EF-emptiness problem
  "Is the set of parameter valuations for which a given location l is reachable empty?"
  undecidable [Alur et al., 1993, Miller, 2000, Doyen, 2007, Beneš et al., 2015]
- AF-emptiness problem
  "Is the set of parameter valuations for which all runs eventually reach a given location l empty?"
  undecidable [Jovanović et al., 2015]

1990s: undecidability

- EF-emptiness problem
  "Is the set of parameter valuations for which a given location l is reachable empty?"
  undecidable [Alur et al., 1993, Miller, 2000, Doyen, 2007, Beneš et al., 2015]
- AF-emptiness problem
  "Is the set of parameter valuations for which all runs eventually reach a given location l empty?"
  undecidable [Jovanović et al., 2015]
- Preservation of the untimed language
  "Given a parameter valuation, does there exist another valuations with the same untimed language?"
  undecidable [André and Markey, 2015]

Bad news

All interesting problems are undecidable for (general) parametric timed automata.

[ÉA, STTT 2018]

2010s: decidability

New decidability results

EF-emptiness problem with a limited number of clocks:

- √ 1 parametric clock and arbitrarily many non-parametric clocks and integer-valued parameters [Beneš et al., 2015]

2010s: decidability

New decidability results

EF-emptiness problem with a limited number of clocks:

- √ 1 parametric clock and arbitrarily many non-parametric clocks and integer-valued parameters [Beneš et al., 2015]
- √ 1 parametric clock and arbitrarily many rational-valued parameters [Miller, 2000]

2010s: decidability

New decidability results

EF-emptiness problem with a limited number of clocks:

- √ 1 parametric clock and arbitrarily many non-parametric clocks and integer-valued parameters [Beneš et al., 2015]
- √ 1 parametric clock and arbitrarily many rational-valued parameters [Miller, 2000]
- √ 2 parametric clocks and 1 integer-valued parameter [Bundala and Ouaknine, 2014]

A subclass with mild decidability results: L/U-PTA

- Partition of parameters into lower-bound and upper-bound parameters
- Positive results [Hune et al., 2002, Bozzelli and La Torre, 2009, André and Lime, 2017]
- Negative results [Jovanović et al., 2015, André and Lime, 2017, André et al., 2018b]

2010s: pragmatism and applications

New algorithms (often independent of the decidability)

- Bounded model checking [Knapik and Penczek, 2012]
- Symbolic integer parameter synthesis [Jovanović et al., 2015]
- Distributed verification [André et al., 2015]
- LTL synthesis [Bezděk et al., 2016, Bezděk et al., 2018]
- Exploration orders [André et al., 2017, Nguyen et al., 2018]
- etc.

Concrete application domains

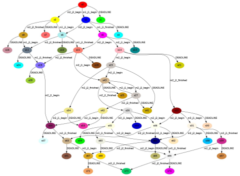
- Scheduling real-time systems
- Hardware verification
- Software product lines
- etc.

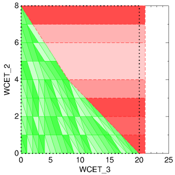
Outline

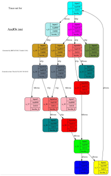
IMITATOR

- A tool for modeling and verifying timed concurrent systems with unknown constants modeled with parametric timed automata
  - Communication through (strong) broadcast synchronization
  - Rational-valued shared discrete variables
  - Stopwatches, to model schedulability problems with preemption

- Synthesis algorithms
  - (non-Zeno) parametric model checking (using a subset of TCTL)
  - Language and trace preservation, and robustness analysis
  - Parametric deadlock-freeness checking







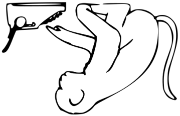
IMITATOR

Under continuous development since 2008 [André et al., FM'12]

A library of benchmarks

- Communication protocols
- Schedulability problems
- Asynchronous circuits
- ...and more

Free and open source software: Available under the GNU-GPL license



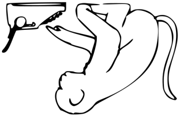
IMITATOR

Under continuous development since 2008 [André et al., FM'12]

A library of benchmarks

- Communication protocols
- Schedulability problems
- Asynchronous circuits
- ...and more

Free and open source software: Available under the GNU-GPL license



Try it!

www.imitator.fr

Some success stories

- Modeled and verified an asynchronous memory circuit by ST-Microelectronics
- Parametric schedulability analysis of a prospective architecture for the flight control system of the next generation of spacecrafts designed at ASTRIUM Space Transportation [Fribourg et al., 2012]
- Verification of software product lines [Luthmann et al., 2017]
- Offline monitoring [ÉA, Hasuo, Waga @ ICECCS'18]
- Formal timing analysis of music scores [Fanchon and Jacquemard, 2013]
- Solution to a challenge related to a distributed video processing system by Thales

Problem

Many recent papers propose new optimizations / new algorithms for parametric timed model checking

- Often in parametric timed automata

Problem

How to ensure a fair evaluation of these new techniques?

Related problem:

- what is the bottleneck for parametric timed model checking?
- what syntactic features do we actually need?
  - is the L/U subclass useful in practice? [Hune et al., 2002]
  - do we need stopwatches? [Cassez and Larsen, 2000]

Outline

A benchmarks library

Accumulated from various sources

- Classical academic models (Fischer)
- Industrial protocols (CSMA/CD, BRP...)
- Industrial collaborations (ST-Microelectronics, Thales, ArianeGroup...)
- Also: education models (useful for teaching)

A benchmarks library

Accumulated from various sources

- Classical academic models (Fischer)
- Industrial protocols (CSMA/CD, BRP...)
- Industrial collaborations (ST-Microelectronics, Thales, ArianeGroup...)
- Also: education models (useful for teaching)

Domains of applications

- Hardware (asynchronous circuits)
- Communication protocols
- Real-time systems (jobshop, scheduling)
- Monitoring automotive systems
- And some more: Wireless fire alarm, producer-consumer...

Organization

A benchmark can contain different models

- Different scales (e. g., Fischer with 2, 3, 4… processes)
- Varying the number of parameters

A model can contain different properties

- For a given instance of Fischer, one can either synthesize correct valuations, or evaluate the system robustness

Organization

A benchmark can contain different models

- Different scales (e. g., Fischer with 2, 3, 4 . . . processes)
- Varying the number of parameters

A model can contain different properties

- For a given instance of Fischer, one can either synthesize correct valuations, or evaluate the system robustness

In the current version:

- 34 benchmarks
- 80 models
- 122 properties

Classifying benchmarks

Number of variables

- Numbers of clocks, parameters. . .

Syntactic features

- Use of global discrete variables
- Use of stopwatches
- L/U property
- etc.

Type of properties

- Reachability/safety
- Unavoidability
- Robustness
- Optimization
- etc.

An insight in the library (with execution times)

Benchmark	Ref	Domain	Scal.	$\|\mathcal{A}\|$	$\|X\|$	$\|P\|$	$\|\mathbb{V}\|$	L/U	Inv	SW	Prop.	Time
						Academic						
And-Or	[Clarisó and Cortadella, 2005]	Circuit	×	4	4	12	0	—	√	×	Misc.	3.01
CSMA/CD	[Kwiatkowska et al., 2007]	Protocol	√	3	3	3	0	—	√	×	Unavoid.	?
Fischer-AHV93	[Alur et al., 1993]	Protocol	√	3	2	4	0	L/U	×	×	Safety	0.04
Fischer-HRSV02:3	[Hune et al., 2002]	Protocol	√	3	3	4	1	L/U	√	×	Safety	HS
Flip-flop:2	[Clarisó and Cortadella, 2007]	Circuit	×	5	5	2	0	U	√	×	Misc.	0.04
Flip-flop:12	[Clarisó and Cortadella, 2007]	Circuit	×	5	5	12	0	U	√	×	Misc.	23.07
idle-time-sched:3	[Lipari et al., 2014]	RTS	√	8	13	2	3	U	√	√	Safety	1.49
idle-time-sched:5	[Lipari et al., 2014]	RTS	√	12	21	2	0	U	√	√	Safety	14.61
Jobshop:3-4	[Abdeddaïm and Maler, 2001]	Sched.	√	2	3	12	4	—	√	×	Opt. reach.	5.58
Jobshop:4-4	[Abdeddaïm and Maler, 2001]	Sched.	√	4	4	16	4	—	√	×	Opt. reach.	∞
NP-FPS-3tasks:50-0	[Jovanović et al., 2013]	RTS	×	4	6	2	0	—	√	×	Safety	1.03
NP-FPS-3tasks:100-2	[Jovanović et al., 2013]	RTS	×	4	6	2	0	—	√	×	Safety	65.23
SSLAF14-1	[Sun et al., 2013]	RTS	×	7	16	2	2	—	√	√	Safety	0.33
SSLAF14-2	[Wandeler et al., 2006, Sun et al., 2013]	RTS	×	6	14	2	4	—	√	√	Safety	∞
ProdCons:2-3	[Knapik and Penczek, 2012]	Prod.-cons.	√	5	5	6	0	L/U	√	×	Reach.	∞
train-AHV93	[Alur et al., 1993]	TGC	×	3	3	6	0	L/U	×	×	Safety	0.01
WFAS	[Beneš et al., 2015]	Protocol	×	3	4	2	0	—	√	×	Safety	∞
						Industrial						
accel:1	[Hoxha et al., 2014, André et al., 2018a]	PTPM	√	2	2	3	0	—	√	×	PTPM	1.25
accel:10	[Hoxha et al., 2014, André et al., 2018a]	PTPM	√	2	2	3	0	—	√	×	PTPM	12.67
BRP	[D'Argenio et al., 1997]	Protocol	×	6	7	2	12	—	√	×	Safety	248.35
FMTV:1A1	[Sun et al., 2015]	RTS	×	3	3	3	5	—	√	×	Opt. reach.	6.97
FMTV:1A3	[Sun et al., 2015]	RTS	×	3	3	3	7	—	√	×	Opt. reach.	87.39
FMTV:2	[Sun et al., 2015]	RTS	×	6	9	2	0	—	√	√	Opt. reach.	1.61
gear:1	[Hoxha et al., 2014, André et al., 2018a]	PTPM	√	2	2	3	0	—	√	×	PTPM	0.77
gear:10	[Hoxha et al., 2014, André et al., 2018a]	PTPM	√	2	2	3	0	—	√	×	PTPM	7.42
RCP	[Collomb-Annichini and Sighireanu, 2001]	Protocol	×	5	6	5	6	L/U	√	×	Reach.	1.07
SIMOP:3	[André et al., 2009]	Automation	×	5	8	3	0	—	√	×	Reach.	∞
SPSMALL:2	[Chevallier et al., 2009]	Circuit	×	11	11	2	0	—	√	×	Reach.	0.96
SPSMALL:26	[Chevallier et al., 2009]	Circuit	×	11	11	26	0	—	√	×	Reach.	∞

License

The entire library is available under the GNU-GPL license

- required sending some emails to authors of papers written 25 years ago ;-)

Library with models, data and (some) results available at

`www.imitator.fr/library.html`

Outline

What's next?

- Naming benchmarks in a unique manner?
  - `benchmark:model:property` (e.g., `Fischer:2:safety`)

- A versioning system will be important to track future changes and ensure fair comparisons
  - "We evaluated our new technique against the parametric timed model checking library v. 1.2"

- Syntax and translation
  - So far, only IMITATOR format
  - At the very least, UPPAAL translation (for compatible models) would be highly welcome

- Contributions
  - So far, users can propose models to the library on a free basis, but an automated method could be welcome

Bibliography

References I

Abdeddaïm, Y. and Maler, O. (2001).
Job-shop scheduling using timed automata.
In Berry, G., Comon, H., and Finkel, A., editors, *CAV*, volume 2102 of *Lecture Notes in Computer Science*, pages 478–492. Springer.

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

Alur, R., Henzinger, T. A., and Vardi, M. Y. (1993).
Parametric real-time reasoning.
In Kosaraju, S. R., Johnson, D. S., and Aggarwal, A., editors, *STOC*, pages 592–601, New York, NY, USA. ACM.

André, É. (2018).
What's decidable about parametric timed automata?
International Journal on Software Tools for Technology Transfer.
To appear.

André, É., Chatain, Th., De Smet, O., Fribourg, L., and Ruel, S. (2009).
Synthèse de contraintes temporisées pour une architecture d'automatisation en réseau.
In Lime, D. and Roux, O. H., editors, *MSR*, volume 43 of *Journal Européen des Systèmes Automatisés*, pages 1049–1064. Hermès.

References II

André, É., Coti, C., and Nguyen, H. G. (2015).
Enhanced distributed behavioral cartography of parametric timed automata.
In Butler, M., Conchon, S., and Zaïdi, F., editors, *ICFEM*, volume 9407 of *Lecture Notes in Computer Science*, pages 319–335. Springer.

André, É., Fribourg, L., Kühne, U., and Soulat, R. (2012).
IMITATOR 2.5: A tool for analyzing robustness in scheduling problems.
In Giannakopoulou, D. and Méry, D., editors, *FM*, volume 7436 of *Lecture Notes in Computer Science*, pages 33–36. Springer.

André, É., Hasuo, I., and Waga, M. (2018a).
Offline timed pattern matching under uncertainty.
In *ICECCS*. IEEE.
To appear.

André, É. and Lime, D. (2017).
Liveness in L/U-parametric timed automata.
In Legay, A. and Schneider, K., editors, *ACSD*, pages 9–18. IEEE.

André, É., Lime, D., and Ramparison, M. (2018b).
TCTL model checking lower/upper-bound parametric timed automata without invariants.
In Jansen, D. N. and Prabhakar, P., editors, *FORMATS*, volume 11022 of *Lecture Notes in Computer Science*, pages 1–17. Springer.

References III

André, É. and Markey, N. (2015).
Language preservation problems in parametric timed automata.
In Sankaranarayanan, S. and Vicario, E., editors, *FORMATS*, volume 9268 of *Lecture Notes in Computer Science*, pages 27–43. Springer.

André, É., Nguyen, H. G., and Petrucci, L. (2017).
Efficient parameter synthesis using optimized state exploration strategies.
In Hu, Z. and Bai, G., editors, *ICECCS*, pages 1–10. IEEE.

Baier, C. and Katoen, J.-P. (2008).
Principles of Model Checking.
MIT Press.

Beneš, N., Bezděk, P., Larsen, K. G., and Srba, J. (2015).
Language emptiness of continuous-time parametric timed automata.
In Halldórsson, M. M., Iwama, K., Kobayashi, N., and Speckmann, B., editors, *ICALP, Part II*, volume 9135 of *Lecture Notes in Computer Science*, pages 69–81. Springer.

Bezděk, P., Beneš, N., Barnat, J., and Černá, I. (2016).
LTL parameter synthesis of parametric timed automata.
In Nicola, R. D. and eva Kühn, editors, *SEFM*, volume 9763 of *Lecture Notes in Computer Science*, pages 172–187. Springer.

Bezděk, P., Beneš, N., Černá, I., and Barnat, J. (2018).
On clock-aware LTL parameter synthesis of timed automata.
Journal of Logic Programming, 99:114–142.

References IV

Bouyer, P., Markey, N., and Sankur, O. (2013).
Robustness in timed automata.
In Abdulla, P. A. and Potapov, I., editors, *RP*, volume 8169 of *Lecture Notes in Computer Science*, pages 1–18. Springer.
Invited paper.

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

Bundala, D. and Ouaknine, J. (2014).
Advances in parametric real-time reasoning.
In Csuhaj-Varjú, E., Dietzfelbinger, M., and Ésik, Z., editors, *MFCS, Part I*, volume 8634 of *Lecture Notes in Computer Science*, pages 123–134. Springer.

Cassez, F. and Larsen, K. G. (2000).
The impressive power of stopwatches.
In Palamidessi, C., editor, *CONCUR*, volume 1877 of *Lecture Notes in Computer Science*, pages 138–152. Springer.

Chevallier, R., Encrenaz-Tiphène, E., Fribourg, L., and Xu, W. (2009).
Timed verification of the generic architecture of a memory circuit using parametric timed automata.
Formal Methods in System Design, 34(1):59–81.

References V

Clarisó, R. and Cortadella, J. (2005).
Verification of concurrent systems with parametric delays using octahedra.
In *ACSD*, pages 122–131. IEEE Computer Society.

Clarisó, R. and Cortadella, J. (2007).
The octahedron abstract domain.
Science of Computer Programming, 64(1):115–139.

Collomb-Annichini, A. and Sighireanu, M. (2001).
Parameterized reachability analysis of the IEEE 1394 root contention protocol using TReX.
In *RT-TOOLS*.

D'Argenio, P. R., Katoen, J.-P., Ruys, T. C., and Tretmans, J. (1997).
The bounded retransmission protocol must be on time!
In Brinksma, E., editor, *TACAS*, volume 1217 of *Lecture Notes in Computer Science*, pages 416–431. Springer.

Doyen, L. (2007).
Robust parametric reachability for timed automata.
Information Processing Letters, 102(5):208–213.

Fanchon, L. and Jacquemard, F. (2013).
Formal timing analysis of mixed music scores.
In *ICMC*. Michigan Publishing.

References VI

Fribourg, L., Lesens, D., Moro, P., and Soulat, R. (2012).
Robustness analysis for scheduling problems using the inverse method.
In Reynolds, M., Terenziani, P., and Moszkowski, B., editors, *TIME*, pages 73–80. IEEE Computer Society Press.

Hoxha, B., Abbas, H., and Fainekos, G. E. (2014).
Benchmarks for temporal logic requirements for automotive systems.
In Frehse, G. and Althoff, M., editors, *ARCH@CPSWeek*, volume 34 of *EPiC Series in Computing*, pages 25–30. EasyChair.

Hune, T., Romijn, J., Stoelinga, M., and Vaandrager, F. W. (2002).
Linear parametric model checking of timed automata.
Journal of Logic and Algebraic Programming, 52-53:183–220.

Jovanović, A., Lime, D., and Roux, O. H. (2013).
Integer parameter synthesis for timed automata.
In Piterman, N. and Smolka, S. A., editors, *TACAS*, volume 7795 of *Lecture Notes in Computer Science*, pages 401–415. Springer.

Jovanović, A., Lime, D., and Roux, O. H. (2015).
Integer parameter synthesis for timed automata.
IEEE Transactions on Software Engineering, 41(5):445–461.

Knapik, M. and Penczek, W. (2012).
Bounded model checking for parametric timed automata.
Transactions on Petri Nets and Other Models of Concurrency V, 6900:141–159.

References VII

Kwiatkowska, M. Z., Norman, G., Sproston, J., and Wang, F. (2007).
Symbolic model checking for probabilistic timed automata.
Information and Computation, 205(7):1027–1077.

Larsen, K. G., Pettersson, P., and Yi, W. (1997).
UPPAAL in a nutshell.
International Journal on Software Tools for Technology Transfer, 1(1-2):134–152.

Lipari, G., Sun, Y., André, É., and Fribourg, L. (2014).
Toward parametric timed interfaces for real-time components.
In Andre, E. and Frehse, G., editors, *SynCoP*, volume 145 of *Electronic Proceedings in Theoretical Computer Science*, pages 49–64.

Luthmann, L., Stephan, A., Bürdek, J., and Lochau, M. (2017).
Modeling and testing product lines with unbounded parametric real-time constraints.
In Cohen, M. B., Acher, M., Fuentes, L., Schall, D., Bosch, J., Capilla, R., Bagheri, E., Xiong, Y., Troya, J., Cortés, A. R., and Benavides, D., editors, *SPLC, Volume A*, pages 104–113. ACM.

Miller, J. S. (2000).
Decidability and complexity results for timed automata and semi-linear hybrid automata.
In Lynch, N. A. and Krogh, B. H., editors, *HSCC*, volume 1790 of *Lecture Notes in Computer Science*, pages 296–309. Springer.

References VIII

Nguyen, H. G., Petrucci, L., and Van de Pol, J. (2018).
Layered and collecting NDFS with subsumption for parametric timed automata.
In Lin, A. W. and Sun, J., editors, *ICECCS*. IEEE.
To appear.

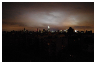
Sun, Y., André, É., and Lipari, G. (2015).
Verification of two real-time systems using parametric timed automata.
In Quinton, S. and Vardanega, T., editors, *WATERS*.

Sun, Y., Soulat, R., Lipari, G., André, É., and Fribourg, L. (2013).
Parametric schedulability analysis of fixed priority real-time distributed systems.
In Artho, C. and Ölveczky, P., editors, *FSTCS*, volume 419 of *Communications in Computer and Information Science*, pages 212–228. Springer.

Wandeler, E., Thiele, L., Verhoef, M., and Lieverse, P. (2006).
System architecture evaluation using modular performance analysis: a case study.
International Journal on Software Tools for Technology Transfer, 8(6):649–667.

Additional explanation

Explanation for the 4 pictures in the beginning



Allusion to the Northeast blackout (USA, 2003)
Computer bug
Consequences: 11 fatalities, huge cost
(Picture actually from the Sandy Hurricane, 2012)

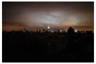
Error screen on the earliest versions of Macintosh

Allusion to the sinking of the Sleipner A offshore platform (Norway, 1991)
No fatalities
Computer bug: inaccurate finite element analysis modeling
(Picture actually from the Deepwater Horizon Offshore Drilling Platform)

Allusion to the MIM-104 Patriot Missile Failure (Iraq, 1991)
28 fatalities, hundreds of injured
Computer bug: software error (clock drift)
(Picture of an actual MIM-104 Patriot Missile, though not the one of 1991)

Licensing

Source of the graphics used I



Title: Hurricane Sandy Blackout New York Skyline
Author: David Shankbone
Source: `https://commons.wikimedia.org/wiki/File:Hurricane_Sandy_Blackout_New_York_Skyline.JPG`
License: CC BY 3.0

Title: Sad mac
Author: Przemub
Source: `https://commons.wikimedia.org/wiki/File:Sad_mac.png`
License: Public domain

Title: Deepwater Horizon Offshore Drilling Platform on Fire
Author: ideum
Source: `https://secure.flickr.com/photos/ideum/4711481781/`
License: CC BY-SA 2.0

Title: DA-SC-88-01663
Author: imcomkorea
Source: `https://secure.flickr.com/photos/imcomkorea/3017886760/`
License: CC BY-NC-ND 2.0

Source of the graphics used II

Title: Smiley green alien big eyes (aaah)
Author: LadyofHats
Source: `https://commons.wikimedia.org/wiki/File:Smiley_green_alien_big_eyes.svg`
License: public domain

Title: Smiley green alien big eyes (cry)
Author: LadyofHats
Source: `https://commons.wikimedia.org/wiki/File:Smiley_green_alien_big_eyes.svg`
License: public domain

License of this document

(LaTeX source available on demand)

Author: Étienne André